# Correcting ‘Wrong-Column’ Errors in Text Databases

**Caroline Sporleder** C.SPORLEDER@UVT.NL
**Marieke van Erp** M.G.J.VANERP@UVT.NL
**Tijn Porcelijn** M.PORCELIJN@UVT.NL
**Antal van den Bosch** ANTAL.VDNBOSCH@UVT.NL
ILK / Language and Information Science, Tilburg University, P.O. Box 90153, 5000 LE Tilburg, The Netherlands

## Abstract

We present a novel data-driven approach for detecting and correcting errors in text databases. We focus on information that was accidentally entered in an incorrect column. Unlike machine-learning approaches to data cleaning that assume the database cells to contain atomic or numeric content, our method takes into account substrings of textual cells, and treats error detection and correction as a text categorisation task. Errors are detected at points where the classifier disagrees with the data; corrections are the suggestions put forward by the classifier. We demonstrate that the method is suited for high-recall detection of errors in free-text columns of a zoological database, with a high correction accuracy as well.

## 1. Introduction

During the past decades, more and more information has become available in digital form; a major part of this information is textual. Not all textual information is stored in raw or typeset form (i.e., as more or less flat text); rather, a lot of it is semi-structured in databases. Publicly accessible examples of such textual databases are: the Internet Movie Database,[1] the University of St. Andrews Photographic Collection,[2] or the Nederlands Soortenregister.[3] Such databases are designed to be automatically searched and queried, and can be important resources for both laypersons and experts.

However, errors and inconsistencies in such databases can have a negative effect on retrieving information from them reliably. For example, a zoologist interested in finding out about the different biotopes (i.e., habitats) in which a given species was found, might query a zoological specimens database for the content of the BIOTOPE column for all specimens of that species. Database records in which information about the biotope was entered in the wrong column will not be retrieved by such a query. Similarly, if a record erroneously lists the wrong species, it will also not be retrieved.

Usually it is impossible to avoid errors completely, even in well-maintained databases. Errors can arise for a variety of reasons, ranging from technical limitations (e.g., copy-and-paste errors) and negligence or attention slips at data entry time, to different interpretations of what type of information should be entered into different database fields. The latter situation is especially prevalent if the database is maintained by several people. Manual identification and correction of errors is frequently infeasible due to the size of the database. A more realistic approach would be to use automatic means to identify potential errors; these could then be flagged to a human expert, and subsequently corrected manually or semi-automatically.

While there has been a significant amount of previous research on identifying and correcting errors in data sets, most methods are not particularly suitable for textual databases (see Section 2) because they treat database cells as atoms. Textual databases typically contain several “free-text” fields, i.e., fields whose

[1] `http://www.imdb.com/`
[2] `http://special.st-andrews.ac.uk/saspecial/`
[3] `http://www.nederlandsesoorten.nl`

content consists of longer strings of text; these should not be treated as atoms. For example, the St. Andrews photograph database has several fields which contain multi-word text strings, for example, encoding the TITLE of the photograph (e.g., "*Cathedral ruins, West Gate, St Andrews. View of West Entrance and East Gable from north west.*"). Longer text strings also occur in the DESCRIPTION and SPECIAL REMARKS columns. Even the shorter fields tend to contain multi-word text strings, such as LOCATION (e.g., "*Fife, Scotland*"), or IMAGE TYPE (e.g., "*full plate glass negative*").

In this paper, we present an error detection method which is sensitive to cell contents containing textual data, rather than treating them as atomic. We focus on one particular type of error, namely information that was entered in the wrong column. We found that this type of error is quite frequent (see Section 4). It can also lead to entries not being retrieved in column-based searches. For example, if a user is interested in all photographs taken in the Kingdom of Fife in Scotland, they might search the St. Andrews Photographic Collection for records in which the location field contains the string "*Fife*". However, if a photograph was taken in Fife but this information was accidentally added in a wrong column, for example PHOTOGRAPHER instead of LOCATION, the corresponding entry will not be returned by such a search. Hence, wrong-column errors can decrease the recall for column-based searches.

To detect this type of error, we developed a knowledge-lean, data driven method in which the content of a database cell is held against all database columns in order to determine which column fits best. While we utilise supervised machine learning, we only exploit the structure and content of the database itself to obtain training data automatically, i.e., no manual annotation of training examples is necessary.

## 2. Related Work

There is a considerable body of previous work on the generic issue of data cleaning. Much of the research directed specifically at databases focuses on identifying identical records when two databases are merged (Hernández & Stolfo, 1998; Galhardas et al., 1999). This is a non-trivial problem as records of the same objects coming from different sources typically differ in their primary keys. There may also be subtle differences in other database fields. For example, names may be entered in different formats (e.g., *John Smith* vs. *Smith, J.*) or there may be typos which make it difficult to match fields (e.g., *John Smith* vs. *Jon Smith*).[4]

In a wider context, a lot of research has been dedicated to the identification of outliers in datasets. The earliest work uses probability distributions to model the data; all instances which deviate too much from the distributions are flagged as outliers (Hawkins, 1980). This approach is called *distribution-based*. In *clustering-based* methods, a clustering algorithm is applied to the data and instances which cannot be grouped under any cluster, or clusters which only contain very few instances are assumed to be outliers (e.g., Jiang et al., 2001). *Depth-based* methods (e.g., Ruts & Rousseeuw, 1996) use some definition of depth to organise instances in layers in the data space; outliers are assumed to occupy shallow layers. *Distance-based* methods (Knorr & Ng, 1998) utilise a $k$-nearest neighbour approach where outliers are defined, for example, as those instances whose distance to their nearest neighbour exceeds a certain threshold. Finally, Marcus and Maletic (2000) propose a method which learns association rules for the data; records that do not conform to any rules are assumed to be potential outliers.

In principle, techniques developed to detect outliers can be applied to databases as well. However, most methods are not particularly suited for textual databases. Some approaches only work with numeric data (e.g., distribution-based methods), others can deal with categorical data (e.g., distance-based methods) but treat all database fields as atoms. For databases with free text fields it can be fruitful to look at individual tokens within a text string. For instance, units of measurement (*m*, *ft*, etc.) may be very common in one column (such as ALTITUDE) but may indicate an error when they occur in another column (such as AUTHOR).

[4] The problem of whether two proper noun phrases refer to the same entity has also received attention outside the database community (Bagga, 1998).

## 3. Data

We tested our error detection and correction method on a database containing information about animal specimens collected by researchers at Naturalis, the Dutch Natural History Museum.[5] The database contains 16,870 entries and 35 columns. Each entry provides information about one or several specimens, for example, who collected it, where and when it was found, its position in the zoological taxonomy, the publication in which the species was first described and classified, and so on. Some columns contain fairly free text (e.g., SPECIAL REMARKS), others contain textual content[6] of a specific type and in a relatively fixed format, such as proper names (e.g., COLLECTOR or LOCATION), bibliographical information (PUBLICATION), dates (e.g., COLLECTION DATE) or numbers (e.g., REGISTRATION NUMBER).

Some database cells are left unfilled, since many cells can take an optional value but may be intentionally empty. Just under 40% of all cells are filled (i.e., 229,430 cells). There is a relatively large variance in the number of different values in each column, ranging from three for CLASS (i.e., "*Reptilia*", "*Amphibia*", and a remark pointing to a taxonomic inconsistency in the entry) to over 2,000 for SPECIAL REMARKS, which is only filled for a minority of the entries. On the other hand there is also some repetition of cell contents, even for the free text columns, which often contain formulaic expressions. For example, the strings "*no further data available*" or "*(found) dead on road*" occur repeatedly in the special remarks field. A certain amount of repetition is characteristic for many textual databases, and we exploit this in our error detection and correction methods.

While most of the entries are in Dutch or English, the database also contains text strings in several other languages, such as Portuguese and French (and Latin for the taxonomic names). In principle, there is no limit to which languages can occur in the database. For example, the PUBLICATION column often contains texts in languages other than Dutch or English.

## 4. Wrong-Column Errors

Wrong-Column errors, i.e. text strings which were entered in the wrong column of the database (e.g., SPECIAL REMARKS instead of BIOTOPE), can arise for a variety of reasons. They can be accidental, i.e., the person entering the information inadvertently chose the wrong column, but they can also be due to misinterpretation, e.g., the person entering the information may have believed that it fitted the SPECIAL REMARKS column better than the BIOTOPE column, or they may not have known that there is a BIOTOPE column. Some of these errors may also stem from changes in the database structure itself, e.g., maybe the BIOTOPE column was only added after the data was entered.[7]

This type of error should be more frequent in free-text columns, such as BIOTOPE, than in columns with a more fixed content, such as SPECIES or ALTITUDE, as the former provide more room for misinterpretation. It is fairly clear what information should be entered in the SPECIES column, but less clear what should be entered in the BIOTOPE column. To get an idea of how frequent wrong-column errors are in free-text columns, we inspected the contents of the BIOTOPE, SPECIAL REMARKS, PUBLICATION, and LOCATION columns and labelled all cases in which we thought that a text string would be better placed in a different column. Table 1 gives some examples of text strings found in these columns. Note that some of the columns are fairly similar to each other. For example, LOCATION, which gives a general description of the area in which a specimen was found, often overlaps with BIOTOPE. In some cases a text string also contains several pieces of information which would best be located in different columns. For example, the text string "*Dry Dipterocarp forest, 400 m alt*" (first line in Table 1) contains information which belongs in the BIOTOPE column ("*Dry Dipterocarp forest*") as well as information which would be better placed in the ALTITUDE column ("*400 m alt*"). We labelled conservatively, that is we only labelled a text string as an error if the string (or a significant part of it) was

[5] `http://www.naturalis.nl`

[6] We use the term *textual content* in the widest possible sense, i.e., comprising all character strings, including dates and numbers.

[7] Many databases, especially in the cultural heritage domain, are not designed and maintained by database experts. Over time, such database are likely to evolve and change structurally. In our specimens database, for example, several columns were only added at later stages.

*Table 1.* Example Text Strings for Five Free-Text Columns

| Text String | Column |
|---|---|
| Dry Dipterocarp forest, 400 m alt | BIOTOPE |
| in a fresh water lake in the interior of Borneo | BIOTOPE |
| This registration number no longer exists | SPECIAL REMARKS |
| Animal caught by (and died in) a bottle-trap. | SPECIAL REMARKS |
| Brongersma (1934) Zool. Med. 17: 161-251 | PUBLICATION |
| see Schlegel, 1837; Physionomie des Serpens. | PUBLICATION |
| 10 km SE of Antalya | LOCATION |
| dead in little pool of water, near valley of Tonto Creek, 18.05 h. | LOCATION |

clearly in the wrong-column. For instance, the string "*Dry Dipterocarp forest, 400 m alt*", which occurred in the BIOTOPE column, was not labelled as an error, whereas the string "*caught indoors*" in the same column was labelled as an error, since it would be better placed in the SPECIAL REMARKS column.

Table 2 shows the proportion of wrong-column errors we found in each of the five database columns; the number of absolute errors is given in brackets. It can be seen that this type of error seems to be quite frequent for BIOTOPE, SPECIAL REMARKS, and LOCATION. This is despite the fact that we were quite conservative when deciding whether a value represents an error. For PUBLICATION, which is arguably less of a free-text column than the other three, the error rate is lower; probably due to the facts that (i) the column is very dissimilar from other columns in the database and (ii) the column is relatively important, so information is probably entered with more care in this column than in some of the less important columns.

*Table 2.* Wrong-Column Error Rates for Free-Text Columns

| Column | Error Rate | |
|---|---|---|
| BIOTOPE | 3.3% | (64) |
| SP. REM. | 2.6% | (250) |
| PUBLICATION | 0.2% | (4) |
| LOCATION | 4.3% | (67) |

The results in Table 2 show that wrong-column errors occur relatively frequently in free-text columns. However, free-text columns provide a challenge for traditional automatic error detection methods (see Section 2) precisely because they contain relatively long and varied text strings. This distinguishes columns like BIOTOPE and LOCATION from other columns in our database, such as ALTITUDE or GENUS, where errors are relatively easy to detect automatically. For instance, ALTITUDE fields should contain a number and a unit of measurement; anything else is likely to point to an error. For columns such as BIOTOPE it is far more difficult to define formally what constitutes a proper cell content. The fact that some free-text columns can also be quite similar to each other poses another difficulty. In sum, detecting wrong-column errors in free-text columns is an important task; however, it is also a task which poses many challenges, especially for traditional error detection methods. In the following section we present a new error detection method that is more suitable for this problem because it does not treat cell contents as atoms.

## 5. Detecting Wrong-Column Errors

We recast the problem of identifying wrong-column errors as a text classification task: given the content of a cell, i.e., a string of text (in the widest sense: one or more words, including numeric strings), the aim is to determine which column the string most likely belongs to. Text strings which are classified as belonging to a different column than they are currently in, represent a potential error. Recasting error detection as a text classification problem allows the use of supervised machine learning methods, as training data (i.e., text strings labelled with the column they belong to) can easily be obtained from the database.

An important assumption underlying the use of database-internal machine learning for self-improvement is that the machine learner is able to correct outliers, while at the same time it is not affected too much by being trained on exactly these errors.

To obtain a training set, we tokenised the text strings in all database fields[8] and labelled them with the col-

[8] We used a rule-based tokeniser for Dutch developed by Sabine Buchholz. The inclusion of multi-lingual abbreviations in the rule set ensures that this tokeniser is robust enough to also

umn they occurred in. Each string was represented as a vector of 83 features, encoding the (i) string itself and some of its typographical properties (13 features), and (ii) its similarity with each of the 35 columns in terms of weighted unigram and bigram overlap (70 features).

The typographical properties we encoded were: the number of tokens in the string and whether it contained an initial (i.e., an individual capitalised letter), a number, a unit of measurement (e.g., *km*), punctuation, an abbreviation, a word (as opposed to only numbers, punctuation etc.), a capitalised word, a non-capitalised word, a short word ($<$ 4 characters), a long word, or a complex word (e.g., containing a hyphen).

The unigram similarity between a string, consisting of a set $T$ of tokens $t_1 \ldots t_n$, and a column $col_x$ was defined as:

$$sim(T, col_x) = \frac{\sum_{i=1}^{n} t_i \times tfidf_{t_i,col_x}}{|T|}$$

where $tfidf_{t_i col_x}$ is the tfidf weight (*term frequency - inverse document frequency*, cf. (Sparck-Jones, 1972)) of token $t_i$ in column $col_x$. This weight encodes how representative a token is of a column. The term frequency, $tf_{t_i,col_x}$, of a token $t_i$ in column $col_x$ is the number of occurrences of $t_i$ in $col_x$ divided by the number of occurrences of all tokens in $col_x$. The term frequency is 0 if the token does not occur in the column. The inverse document frequency, $idf_{t_i}$, of a token $t_i$ is the number of all columns in the database divided by the number of columns containing $t_i$. Finally, the tfidf weight for a term $t_i$ in column $col_x$ is:

$$tfidf_{t_i,col_x} = tf_{t_i,col_x} \; \log \; idf_{t_i}$$

A high tfidf weight for a given token in a given column means that the token frequently occurs in that column but rarely in other columns, thus the token is a good indicator for that column. Typically tfidf weights are only calculated for content words; we calculated them for all tokens, partly because the use of stop word lists to filter out function words would have jeopardised the language independence of our method and partly because function words and even punctuation can be very useful for distinguishing different columns. For example, prepositions such as "*under*" often indicate BIOTOPE, as in "*under a stone*".

Bigram overlap was defined analogously, however we looked at overlap between bigram sequences rather than unigram sequences.

## 6. Experimental Results

We applied the classifier to the text strings in the four free-text columns discussed above (BIOTOPE, SPECIAL REMARKS, PUBLICATION, and LOCATION). To assign a text string to one of the 35 database columns, we trained a memory-based classifier (TiMBL, Daelemans et al. (2004)) on the feature vectors of all other database cells labelled with the column they belong to.[9] All cells were tested this way, thus constituting a leave-one-out experiment for each of the four columns. Cases in which the predicted column differed from the current column of the string were recorded as potential errors. We then determined whether these potential errors were also identified as errors in the manually labelled test set (see Section 4) and calculated error detection precision (P), recall (R) and F-Score (F). Note that recall is more important than precision if the errors are corrected semi-automatically, i.e., when a human annotator checks the errors flagged by the system. A low precision means more work for the user; a low recall, on the other hand, implies that many errors are not detected at all, undermining our goals. In addition to the error *detection* scores, we also calculated the error *correction* accuracy, defined as the number of errors for which the right column was suggested by the classifier divided by the number of correctly detected errors. If the error correction accuracy is high, the suggested correction could be presented to the human annotator; in most cases the annotator could then simply choose the suggested column, rendering the semi-automatic error correction process easier and faster.

Tables 3 to 5 show the results for using (i) only typographical features (including the text string itself), (ii) typographical features and weighted unigram overlap, and (iii) all features (typographical, unigram and bigram overlap), respectively.

cope with text strings in English and other Western European languages.

[9] We used the default settings (IB1, Weighted Overlap Metric, Information Gain Ratio weighting) and $k$=3.

*Table 3.* Only Typographical Features

| Column | Errors Flagged | Detection P % | R % | F % | Correction Acc. % |
|---|---|---|---|---|---|
| BIO. | 629 | 9.7 | 95.3 | 17.6 | 21.3 |
| SP. REM. | 1148 | 9.1 | 41.6 | 14.9 | 35.6 |
| PUBL. | 139 | 2.9 | 100.0 | 5.6 | 25.0 |
| LOC. | 519 | 11.4 | 88.1 | 20.1 | 54.2 |

*Table 4.* Typographical Features Plus Unigram Overlap

| Column | Errors Flagged | Detection P % | R % | F % | Correction Acc. % |
|---|---|---|---|---|---|
| BIO. | 234 | 24.4 | 89.1 | 38.3 | 91.2 |
| SP. REM. | 298 | 20.1 | 24.0 | 21.9 | 61.7 |
| PUBL. | 58 | 6.9 | 100.0 | 12.9 | 25.0 |
| LOC. | 286 | 18.2 | 77.6 | 29.5 | 51.9 |

*Table 5.* Typographical Features, Unigram and Bigram Overlap

| Column | Errors Flagged | Detection P % | R % | F % | Correction Acc. % |
|---|---|---|---|---|---|
| BIO. | 167 | 36.5 | 95.3 | 52.8 | 86.9 |
| SP. REM. | 278 | 19.4 | 21.6 | 20.5 | 48.2 |
| PUBL. | 55 | 7.3 | 100.0 | 13.6 | 0.0 |
| LOC. | 274 | 18.6 | 76.1 | 29.9 | 54.9 |

It can be seen that using only typographical features already leads to a fairly high recall for all columns, except SPECIAL REMARKS (SP. REM.), where the recall is only 41.6% (Table 3). The precision, however, is relatively low, in most cases below 10%, and for PUBLICATION only 2.9%. The correction accuracy is also fairly low, ranging between 21.3% for BIOTOPE (BIO.) to 54.2% for LOCATION (LOC.). Adding unigram overlap (Table 4) lowers the recall somewhat, but leads to an increase in precision of 50% or more. The correction accuracy also increases dramatically for all columns, except for PUBLICATION (PUBL.), and now lies between 25% for PUBLICATION and 91.2% for BIOTOPE. Adding bigram overlap as well (Table 5) leads to another slight increase in detection precision and for BIOTOPE also to a significant increase in recall. The error correction accuracy, however, drops, except for a slight increase on LOCATION.

Given the difficulties of detecting and correcting errors in free-text columns, the overall results are quite promising. While the error detection precision is not particularly high, the number of flagged errors is small enough for a semi-automatic approach, especially given that checking this type of error can be done relatively quickly. Furthermore, the use of an automatic method for identifying potential errors already decreases the workload significantly compared to fully manual correction. The recall, which is important for semi-automatic error correction, is relatively high, except for SPECIAL REMARKS. That column, however, is very heterogeneous, being a catch-all for information that does not fit anywhere else. Hence, there is no proto-typical "special remark"; the column can contain anything from a remark on an inconsistency in the taxonomic information of a record over comments on the validity of a registration number to information about the circumstances under which a specimen died. Furthermore, many text strings in this column contain several pieces of information, many of which belong to another column (as in "*found with broken neck near Karlobag*", which could be split between the SPECIAL REMARKS and the LOCATION columns). Consequently, errors in this column are particularly difficult to spot.

Table 6 shows some of the automatically corrected errors. Note that the system corrected errors in both English and Dutch text strings without requiring language identification or any language-specific resources.

## 7. System Error Analysis

In the previous section, we showed that our error correction method obtains a reasonably good performance on four free-text columns in the database. In this section, we investigate the origin of the remaining system errors. For each of the columns we looked at the false negatives, i.e., true database errors which were missed, and false positives, i.e., text strings which were flagged as database errors but turned out not to be real errors. We focus particularly on false positives as they were far more frequent.

**Biotope** Three false negatives; 106 false positives. The most frequently predicted columns for false positives were SPECIAL REMARKS (42 cases), PLACE (32) and LOCATION (19). LOCATION, PLACE and BIOTOPE overlap significantly (see Section 4) and are difficult to distinguish, even by a human. Also, most text strings which were misclassified as PLACE do contain a geographical name (e.g., "*on road at night, Kalahari bushveld vegetation*"). As said earlier, the confusions with SPECIAL REMARKS usually stem from the fact that that column is so heterogeneous and sometimes contains elements of BIOTOPE information.

*Table 6.* Examples of Corrected Errors

| Text String | Original Column | Corrected Column |
|---|---|---|
| op boom ongeveer 2,5 m boven grond<br>(*on a tree about 2.5 m above ground*) | SPECIAL REMARKS | BIOTOPE |
| 25 km N.N.W Antalya | SPECIAL REMARKS | LOCATION |
| 1700 M | BIOTOPE | ALTITUDE |
| gestorven in gevangenschap 23 september 1994<br>(*died in captivity 23 September 1994*) | LOCATION | SPECIAL REMARKS |
| roadside bordering secondary forest | LOCATION | BIOTOPE |
| Suriname Exp. 1970<br>(*Surinam Expedition 1970*) | COLLECTION NUMBER | COLLECTOR |

**Special Remarks** 196 false negatives; 224 false positives. It is striking, but not unexpected, that the number of different columns with which SPECIAL REMARKS is confused is much higher than for any of the other three free-text columns. For example, the false negatives belong to eight different columns; for the false positives 19 different columns are predicted. Again, text strings misclassified as PLACE frequently contain geographical names (e.g., "*obtained in exchange from Section Herpetology, South Australian Museum*"). Text strings misclassified as BIOTOPE or LOCATION tend to contain prepositions which are predictive for those two columns, e.g., "*op*" (on) or "*in*": "*op etiket: III 476*", "*in reg. boek als Rana*" (in registration book (classified) as Rana), which were both misclassified as BIOTOPE.

**Publication** No false negatives; 51 false positives. The most frequent misclassifications were: SPECIAL REMARKS (24), PLACE (18), AUTHOR (6). Confusions with SPECIAL REMARKS may again be due to the heterogeneous nature of that column, which does indeed contain some bibliographical references (e.g., "*Zie ook L. D. Brongersma (1966) Zool. Meded. 41 (17) : 243-254.*"). Furthermore, most of these misclassifications involved text strings which were exceptionally long for the PUBLICATION field. PLACE is predicted relatively frequently because many publications do contain a geographical reference (e.g., "*Hoogmoed, M.S., 1973, Notes on the Herpetofauna of Surinam IV*"). Finally, the AUTHOR field overlaps with the PUBLICATION field in some cases, e.g., it sometimes contains book titles in addition to the author's name.

**Location** 16 false negatives; 223 false positives, of which the most frequently predicted columns are: PLACE (112), SPECIAL REMARKS (46), BIOTOPE (34), PROVINCE (17). Again the confusions arise mainly with columns which are similar to LOCATION and with the heterogeneous SPECIAL REMARKS field.

Summarising, it seems that most of the system errors stem from either confusions between very similar columns (e.g., LOCATION and BIOTOPE) or from confusions with the highly heterogeneous SPECIAL REMARKS column. Confusions with similar columns are difficult to deal with, as there is often a real overlap between columns, which, for some text strings, makes it difficult to decide on one of the columns, even for humans. Confusions with the SPECIAL REMARKS column could be dealt with by adding a filter which does not flag a potential error to a user if the predicted column is SPECIAL REMARKS. This would probably increase the precision for most columns, though it might also lead to a small decrease in recall.

We also noticed a few systematic errors. Single capitalised words within brackets tend to be classified as AUTHOR. Text strings in the AUTHOR field are indeed often in this format (e.g., "*(Audouin)*"),[10] however, the system also misclassified the string "*(Kikkervisje)*" (frog fish) as AUTHOR. Similarly, some text strings referring to a location, such as "*N.W. van Meknes*" (North West of Meknes), were classified as COLLECTOR, presumably because the string looks rather like a person name. Named-entity tagging might help in these cases. In a database such as ours, it is often possible to glean a lot of information about named-entities in the domain by looking at entity-specific columns such as COLLECTOR (see Sporleder et al., 2006).

[10] The bracket indicates that the original species classification by this author has been superseded by a more recent reclassification.

## 8. Conclusion

We have presented a novel machine-learning-based error detection and correction method for textual databases. The method aims to identify wrong-column errors, i.e., cell content that was accidentally entered into an incorrect column. We found that this error is fairly frequent, especially in free-text fields. However, free-text fields pose a challenge to traditional outlier detection and error correction methods, which treat cell contents as atomic. Our method overcomes this deficiency by looking at individual words within a text string to decide whether the column is correct, thus recasting the error detection and correction problem as a text categorisation task.

We tested this approach on a zoological database and found that it is suitable for semi-automatic error correction, where potential errors are flagged to a human annotator for manual checking and correction. A significant proportion of errors could be detected at a recall level of up to 100%. While the precision was fairly low (with a maximum of 37%), the number of potential errors flagged was still sufficiently small to check manually, especially when compared to the size of the full database (i.e., several hundreds of corrections vs. a total amount of 229,430 filled cells). Furthermore, the automatically predicted column for an error was often the right one. The remaining false positive system errors seem to be largely due to confusions between very similar and overlapping columns.

While our method utilises supervised machine learning, no manual annotation of training data is required, as the training set is obtained directly from the database itself. Hence, the approach is knowledge-lean and data-driven. It is also language-independent. These properties should make it relatively easy to port the system to new databases and domains.

One improvement we would like to make in future research is to look into ways to split text strings and assign them to different columns. This would be beneficial because individual strings often contain several pieces of information which should be placed in different columns (as in "*found with broken neck near Karlobag*", see Section 4). One way to achieve this, would be by viewing error detection as a sequence labelling task, of which the aim is to determine which substrings of a text string belong to which columns.

**Acknowledgments** The research reported in this paper was funded by NWO (Netherlands Organisation for Scientific Research) and carried out at the Naturalis Research Labs in Leiden. We would like to thank Pim Arntzen and Erik van Nieukerken from Naturalis for helpful discussions. We are also grateful for the comments received from three anonymous reviewers.

## References

Bagga, A. (1998). *Coreference, cross-document coreference, and information extraction methodologies*. Doctoral dissertation, Dept. of Computer Science, Duke University.

Daelemans, W., Zavrel, J., van der Sloot, K., & van den Bosch, A. (2004). *TiMBL: Tilburg memory based learner, version 5.1, reference guide*. ILK Research Group Technical Report Series no. 04-02.

Galhardas, H., Florescu, D., Shasha, D., & Simon, E. (1999). *An extensible framework for data cleaning* (Technical Report RR-3742). INRIA Technical Report.

Hawkins, D. M. (1980). *Identification of outliers*. London: Chapman and Hall.

Hernández, M. A., & Stolfo, S. J. (1998). Real-world data is dirty: Data cleansing and the merge/purge problem. *Journal of Data Mining and Knowledge Discovery*, *2*, 1–31.

Jiang, M.-F., Tseng, S.-S., & Su, C.-M. (2001). Two-phase clustering process for outliers detection. *Pattern Recognition Letters*, *22*, 691–700.

Knorr, E. M., & Ng, R. T. (1998). Algorithms for mining distance-based outliers in large datasets. *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB'98)*.

Marcus, A., & Maletic, J. I. (2000). *Utilizing association rules for identification of possible errors in data sets* (Technical Report TR-CS-00-04). The University of Memphis, Division of Computer Science.

Ruts, I., & Rousseeuw, P. J. (1996). Computing depth contours of bivariate point clouds. *Computational Statistics and Data Analysis*, *23*, 153–168.

Sparck-Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, *28*, 11–21.

Sporleder, C., van Erp, M., Porcelijn, T., van den Bosch, A., & Arntzen, P. (2006). Identifying named entities in text databases from the natural history domain. *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC-06)*. Genoa, Italy.