# Searching for Semantically Correct Postal Addresses on the Croatian Web

**Ivo Ugrina**
Faculty of Science (Department of Mathematics)
University of Zagreb
Bijenička 30, 10000 Zagreb, Croatia
iugrina@math.hr

**Mislav Žigo**
MIREO d.d.
Banjole, Mušoga 3, Medulin, Croatia
mislav.zigo@mireo.hr

**Abstract.** *This article presents a method of extraction and simultaneous verification of postal addresses within web pages written in a highly inflective language (Croatian). The method uses a combined approach of direct city name extraction, string similarity measure (Jaro-Winkler) for street names, an algorithm for treating overlapping addresses and a machine learning classifier (Decision trees) to derive Semantically Correct Postal Addresses. A Semantically Correct Postal Address is defined as one that was meant to be written by an author of the text and is not simply there by a lucky ordering of words. The presented method jointly does geoparsing and geocoding. For the initial search of cities and streets, the method relies on a database containing most of the streets and cities in Croatia. The method was evaluated on a data set consisting of 13,000,000 documents (from 35,000 web domains) and resulted in 4,000,000 addresses found in 2,750,000 documents. The quality of classifiers was tested on a hand annotated set giving $F_1$ scores greater than 0.9.*

**Keywords.** postal addresses; string similarity; machine learning; geographic location; address extraction

## 1 Introduction

Automatic recognition of geographical context (i.e. location information) of web documents can be a difficult task However, it also has countless applications in economics, social sciences and information science. For example, in geospatial search engines ([12]), exploration of spatial context of published news stories ([19], [15]) or other *Geographic information systems (GIS)*.

Unfortunately, the web does not explicitly reveal its location-relation, as this information is hidden somewhere within pages' contents. To exploit such location information, a process to find, extract and geospatially index relevant web pages needs to exist. A method for extracting postal addresses from web pages would be of great importance to this process.

Even though addresses are used worldwide and basically consist of the same components, the order of these components varies among countries (see [18]). The method of extracting postal addresses should be able to deal with (or be able to learn to deal with) those varieties. Therefore, a machine learning approach seems plausible.

Found addresses can be separated into two sets: semantically correct addresses and syntactically correct addresses. Syntactically correct addresses are those that consist of all the necessary parts to uniquely identify some real world address, while semantically correct addresses are those that were meant to be written. An illustrative example can be seen in the following text (Split and Zagreb are cities in Croatia);

> *"... big shops in Gajeva 2, Zagreb, Vukovarska 41, Split, where one could ..."* (1)

Addresses *(Gajeva 2, Zagreb)*, *(Zagreb, Vukovarska 41)*, *(Vukovarska 41, Split)* and *(Gajeva 2, Split)* are all syntactically correct addresses, i.e. they all exist in real world. However, the author of the text almost surely meant for *Vukovarska 41, Split* instead of *Zagreb, Vukovarska 41* and *Gajeva 2, Zagreb* instead of *Gajeva 2, Split*. Hence, *Gajeva 2, Zagreb* and *Vukovarska 41, Split* are the only semantically correct addresses in this example. It should be noticed that every semantically correct address is also a syntactically correct address. Therefore, some kind of preliminary search for a set of possible syntactically correct addresses should be conducted.

The importance of distinguishing between semantically and syntactically correct addresses for applications is high. If the example above is imagined as taken from a web page of a company and those addresses represent offices in different cities, a search engine doing a geospatial search should score results based on semantically correct addresses, thus leading to a better user experience.

The rest of this article is organized as follows: in Section 2, related work is presented; in Section 3, the design of the method for recognition of semantically correct addresses is presented; in Section 4, a binary decision tree classifier is built and appropriate predic-

tors are introduced; in Section 5, some interesting results are presented; in Section 6, conclusions and remarks are given.

# 2 Related Work

The idea to separate addresses to semantically correct and syntactically correct seems to be new, but recognition of syntactically correct addresses in web documents is a well-studied problem, especially in GIS or GIR. To be precise, other authors do not give the solution on how to treat ambiguities like overlapping addresses, but they do extract postal addresses that are syntactically correct and mostly semantically correct.

A short explanation on how to identify and extract address strings from web content is given in Wang et al. [20] where the existence of a postal address in a web document was used as a predictor to infer a content location.

A syntactic approach where almost all possible grammatical patterns of postal addresses are built is presented by Can et al. [8].

Cai et al. [7] employ an ontology-based conceptual information retrieval approach combined with graph matching techniques to automatically identify possible address structures in a web page.

An ontology-based approach that helps recognise, extract and geocode geospatial evidence with local characteristics, such as street names, urban landmarks, telephone area codes and postal addresses, was demonstrated in Borges et al. [5] and Borges et al. [4].

Rule-based, machine learning and hybrid approaches are presented by Yu [22]. These approaches lack sensitivity for mistypes, number of characters/words between address elements and position of street name relative to the appropriate city.

Another pattern-based approach for extraction of addresses from Web pages is presented by Asadi et al. [3]. Both HTML and vision-based segmentations are used to increase the quality of address extraction.

In Loos and Biemann [16], the Natural Language Processing (NLP) approach using Conditional Random Fields (CRFs) to label the address of a target location, and no other addresses that might be contained in the same document, is applied.

CRFs are also used by Chang and Li [9]. The approach is basically the combination of Loos and Biemann [16] and Yu [22].

Chang et al. [10] extend the idea of Chang and Li [9] to Chinese postal addresses extraction and improve the extraction associated information for each address with hierarchical clustering. Unfortunately, their approach relies on different suffixes, like avenue, road, street, etc., and is not easily applicable to Croatian addresses where the use of such suffixes in the general public's writing of addresses is rare.

An excellent introduction to history of addresses along with common addressing concepts is given by Davis and Fonseca [11]. Also, the authors use similar methods to the ones described here for recognition of addresses (defined in a broader sense). They define approximate and exact matches and use approximate string searching algorithms for approximate matching. The definition of a Geocoding Certainty Indicator (GCI) introduces some type of semantical correctness. However, they do not address the problem of overlapping addresses, and no evaluation results are provided.

An approach that is very similar to the one presented here is given by Ahlers and Boll [1]. Authors of the article indicate most of the problems that were noticed during this research and give some very insightful remarks. The problem of overlapping addresses is presented, but no explicit solution is given. Also, only postal addresses with ZIP codes are recognised, leaving those consisting of city name, street and house number for future work.

The use of databases to recognise place (city) names is presented in, for example, Amitay et al. [2].

# 3 Design

The method can basically be broken into two steps: searching for cities and searching for semantically correct postal addresses in those cities. To obtain semantically correct postal addresses, a mid-step of finding all probable syntactically correct addresses is necessary.

First step and mid step are presented in this section, while the second step is the subject of Section 4.

## 3.1 Searching for cities

The anchor point of address matching within any free text document (including HTML) is matching of cities by their name. Although some authors use postal codes as the anchor (like [1]), this is inadequate in the case of the Croatian web, and we believe that the same is true for many other countries and languages.

Borges et al. [4] noted that only around 50% of addresses (*BasicAddress* in their article) had a postal code near them. In our data set, only about 75% of addresses had a postal code near the city name. It was also noticed that postal codes were not accurate in some cases. Still, when a postal code exists, it is a useful secondary tool for treating ambiguities and possible mistypes as well as the alternate forms of city names.

After parsing, a HTML document is stripped of most non-alphanumeric characters. Furthermore, all sequences of white space and line break characters are replaced by a single space, thus converting the document into a list of words. Therefore, the converted document can be regarded as a linearisation of visible web page text.

Only words with capital first letters are candidates for (beginnings of) city names. When such a word is encountered, the first step is to standardise it, which in most cases means de-abbreviation.

The database is then searched for all cities whose name begins with exactly that standardised word, returning the list of candidate cities. Next, subsequent words in their standardised form are added in order to obtain the longest possible sequence of words that still has a fitting candidate. If this longest sequence is exactly the same as the name of a candidate, a full match is obtained. For example, if the document contains the name of the city *Velika Gorica*, a search with the word *Velika* will be made, and the list of candidates *{Velika, Velika Gorica}* (both are cities in Croatia) will be returned. Adding the next word from the document to the query, a single exact match will be obtained. Once the exact match is given, close proximity of the city name is inspected for the existence of a postal code using a simple regular expression. For Croatia, this is a 5-digit number, possibly prefixed by HR.

## 3.2 Searching for full addresses

The idea now is to inspect proximity of each matched city name (several words left and right) and look for possible house numbers and street names.

In doing so, one has to be prepared to deal with the fact that the Croatian language and existing practice support several forms of writing a street name, and all of them are frequently used.

Let us view this through the following example: we have a street whose full name is *Ulica Ljudevita Gaja*. We know that "ulica" means "street" and "Ljudevit Gaj" is a historic person. This street's name can be written as *Ljudevita Gaja*, *Gaja*, *Lj. Gaja* or, even more commonly, *Gajeva*, which is its possessive adjective form (all other forms are genitive case). Three requirements arise from this example alone: subset matching should be supported (*Gaja* should be a match to *Ulica Ljudevita Gaja*), initials and abbreviations (*Lj. Gaja* should be a match to *Ljudevita Gaja* and subsequently to *Ulica Ljudevita Gaja*) and matching of various forms of a single word (*Gajeva* should be a match to *Gaja* and subsequently to all other forms). The first two requirements can be fulfilled by some simple rule-based techniques, but a string similarity measure that tends to assign a relatively high value to pairs of strings that may differ in the last few letters will be necessary for the third one. An application of the Jaro-Winkler string similarity proved to be the most efficient technique to achieve this.

## 3.3 Jaro-Winkler string similarity

Jaro string similarity measure is a result from Mathew A. Jaro published in the article Jaro [14]. The measure was constructed for problems of *record linkeage* and its main purpose was to be used on short sequences representing names. Let us present the definition of the measure.

Let two words be denoted by $s = a_1 a_2 \cdots a_K$ and $t = b_1 b_2 \cdots b_L$. We say that a character $a_i$ in $s$ is matched in $t$ if there exists $b_j = a_i$ in $t$ such that $i - H \leq j \leq i + H$, where $H = min(|s|, |t|)/2$. Let $s' = a'_1 a'_2 \cdots a'_{K'}$ be characters in $s$ matched in $t$ and let $t' = b'_1 b'_2 \cdots b'_{L'}$ be characters in $t$ matched in $s$ (sorted by their order in words $s, t$). Let us now define a transposition between $s'$ and $t'$ as a position $i$ such that $a'_i \neq b'_i$. Let $T_{s',t'}$ be the number of transpositions for $s'$ and $t'$ divided by two. Jaro string similarity between words $s$ and $t$ is given by the following relation

$$Jaro(s,t) = \frac{1}{3}\left(\frac{|s'|}{|s|} + \frac{|t'|}{|t|} + \frac{|s'| - T_{s',t'}}{|s'|}\right) .$$

The result of Jaro string similarity is a real number from segment [0,1]. Greater number represents greater similarity.

Jaro-Winkler string similarity [21] is an extension of Jaro string similiary by the idea that prefixes play important role in words from natural languages. For words $s$ and $t$ the measure is defined by

$$JaroWinkler(s,t) = Jaro(s,t) + prefixLength * PREFIXSCALE * (1.0 - Jaro(s,t)) ,$$

where $prefixLength$ is the length of the common prefix between words $s$ and $t$ while $PREFIXSCALE$ is a constant (weight) for result correction. It is obvious that results from Jaro-Winkler and Jaro string similarities will be equal if words $s$ and $t$ do not contain a common prefix. Jaro-Winkler string similarity returns a real non-negative number where greater number represents greater similarity.

### 3.3.1 Exact Addresses

Exact addresses are defined as those that form a perfect match with one of the database addresses.

The algorithm for extracting such addresses is simple. One of the previously discovered city names is taken, its proximity is inspected for expressions that may denote a street name and a house number, and the database is then checked for such addresses in that city. The obtained address should be a syntactically correct postal address. However, if the process were to stop here, the number of false positives could be high, especially when there is a list of addresses next to each other in the document (possibly overlapping). A certain address may be matched in a wrong city if two cities that are nearby in the document have that same street (see (1)).

Therefore, to obtain semantically correct postal addresses, with each found address the values of some variables describing conditions under which this address is found are collected. These variables are later used in a classifier in order to purify results of the matching process.

### 3.3.2 Approximate Addresses

Approximate addresses are defined as those that form a partial match with one of the database addresses.

A search for approximate addresses is conducted in the proximity of a city after the process of extracting exact addresses is finished. First, expressions that may denote house numbers are found. For each house number candidate, the next word to its left (in the Croatian language, house numbers are always written after street name, though it is different for some other languages) are iteratively added, thus forming a query.

Next, the set of words in the query is scored against the set of words representing street names from the database. For each query word, the corresponding word is found in a street name so that Jaro-Winkler string similarity measure is maximal. This is done until all query words are matched. Repetitions are forbidden: each word on both sides can be used only once. The similarity measure of this match is simply a product of similarity measures of all matched pairs of words.

Finally, the candidate address with highest similarity measure is chosen.

Now, a set of possible syntactically correct addresses is obtained. As with exact addresses, to reduce the candidates to semantically correct addresses, with each matched address a number of variables describing conditions under which an address is matched are collected.

# 4 Evaluation

As was illustrated in example *"... Gajeva 2, Zagreb, Vukovarska 41, Split ..."* in the introduction (see (1)), queries *Gajeva 2* and *Vukovarska 41* can be matched with cities *Split* and *Zagreb* since both cities contain such addresses, giving a set of possible semantically correct addresses. However, only *Gajeva 2, Zagreb* and *Vukovarska 41, Split* are addresses the author referred to.

To decide which of these are semantically correct, some variables describing conditions under which addresses are matched are collected.

After a comprehensive study, predictors for exact and approximate addresses were defined as follows: *LR* is indicating if the address was matched left or right from the city; *WBQAC* is the number of words between query and city (with interpunction); *Dist* is the number of words between query and city (without interpunction); *CBQAC* is the number of characters between query and city (with interpunction); *QLW* is the length of the longest word in the query; *NNA* is the number of numbers in the proximity of the query; *InterQN* denotes if there exists an interpunction character between house number and street name (e.g. string "Vukovarska: 41"); *BaseHouseNo* indicates if the database returned a house number; *ExistsBetter* indicates if there is a "better" address than currently found (thoroughly explained in the next Section); *QueryHouseNoLen* is the number of digits of the house number in the query; *QBDW* is the difference between the number of words in the query and number of words in the street name returned from the database; *QBDC* is the difference between number of characters in query and number of characters in the street name returned from the database; *CandWordsNo* is the number of words sent for extraction of candidates; *CandCapitalNo* is the number of capitalised first letters sent for extraction of candidates; *JW* is the product of Jaro-Winkler values obtained with respect to words from the query and database results; *JWWordsNo* is the number of words used in calculation of JW and *JWCharsNo* is the number of characters used in calculation JW.

Table 1: An example of formating of addresses in HTML documents

```
<table>
  <tr> <td>Street:</td>
    <td>Vukovarska 41</td> </tr>
  <tr> <td>City:</td>
    <td>Split</td> </tr>
  <tr> <td>State:</td>
    <td>Croatia</td> </tr>
</table>
```

| | |
|---|---|
| Street: | Vukovarska 41 |
| City: | Split |
| State: | Croatia |

The reason for differentiating between *WBQAC* and *Dist* (even their usefulness) might seem awkward since it is common practice to write street names and cities together. But, since postal addresses in HTML can be written in different formats, additional characters or words between address parts (city, street, ...) might exist. An example is given in Table 1 where the address is written in HTML as a table element. This was also noticed by Ahlers and Boll [1] and is an important part of their geoparser.

Obviously, some predictors are specific only to approximate addresses. The relation between variables and exact or approximate addresses is presented in Table 2.

## 4.1 The ExistsBetter flag

The problem of overlapping addresses needs thorough consideration. Therefore, a decision was made to include a binary variable (flag) known as *ExistsBetter* to denote the best candidate in the set of overlapping candidates. All other addresses participating in that overlapping will receive a flag denoting that there is (locally) a better address that uses those words.

A dedicated classifier might seem to be a natural choice, but the approach presented here is based on observations and intuition. The observed importance of

Table 2: Affiliation of predictors to type of addresses

| variable (predictor) | exact | approximate |
|---|---|---|
| LR | + | + |
| WBQAC | + | + |
| Dist | + | + |
| CBQAC | + | + |
| QLW | + | + |
| NNA | + | + |
| InterQN | + | + |
| BaseHouseNo | + | + |
| ExistsBetter | + | + |
| QueryHouseNoLen | + | + |
| QBDW | | + |
| QBDC | | + |
| CandWordsNo | | + |
| CandCapitalNo | | + |
| JW | | + |
| JWWordsNo | | + |
| JWCharsNo | | + |

*ExistsBetter* in classifiers for semantically correct addresses (presented later) seems to justify the definition of the flag as described here.

For two addresses, $a$ and $b$, which are competing for the same query words, the $ExistsBetter$ flag is derived from the algorithm described in Table 3.

The *favorite side* is defined for every document in the following way: if 70% of candidates for addresses in that document have queries on a specific side of the cities, then it is defined as that side. Otherwise, it is left undefined. The idea behind the *favorite side* is that the style of the document is almost always constant, i.e. either almost all addresses are written right of the city or almost all are written left of the city. It is hard to justify a mix of these styles in one document. A threshold of 70% was chosen based on empirical observations by the ROC curve analysis.

In short, *ExistsBetter* describes a preference of: exact addresses over approximate, the existence of house numbers, bigger Jaro-Winkler (JW) values, shorter word distance to the city, etc.

## 4.2 Classification with decision trees

The training set consisted of 2659 possible addresses from which 1510 were of type approximate and 1149 of type exact. Of approximate addressees, 46% were marked as correct, and of exact addressees, 73% were marked as correct.

The *R* software environment for statistical computing and graphics (*rpart* package) was used to fit decision trees for exact and approximate addresses. It uses the CART algorithm to fit trees, as described in Breiman et al. [6]. The decision trees were pruned to reduce the effect of over-fitting the training data set. The results are illustrated in Fig. 1 and Fig. 2.

Table 3: Simple algorithm (classifier) for *ExistsBetter* flag

```
a == approx and b == exact:  choose b
a == exact and b == exact:
   b superset of a:  choose b
   b contains house number, a does not:
     choose b
   street and house number equal:
     b has shorter distance to city:
       choose b
     a and b have equal distance to city:
       choose by favorite side
a == approx and b == approx:
   JW(b) > JW(a):  choose b
   JW(b) == JW(a):
     query length for b > query length for a:
       choose b
     query length for a == query length for b:
       b contains house number, a does not:
         choose b
       b has shorter distance to city:
         choose b
       a and b have equal distance to city:
         choose by favorite side
otherwise choose a
```

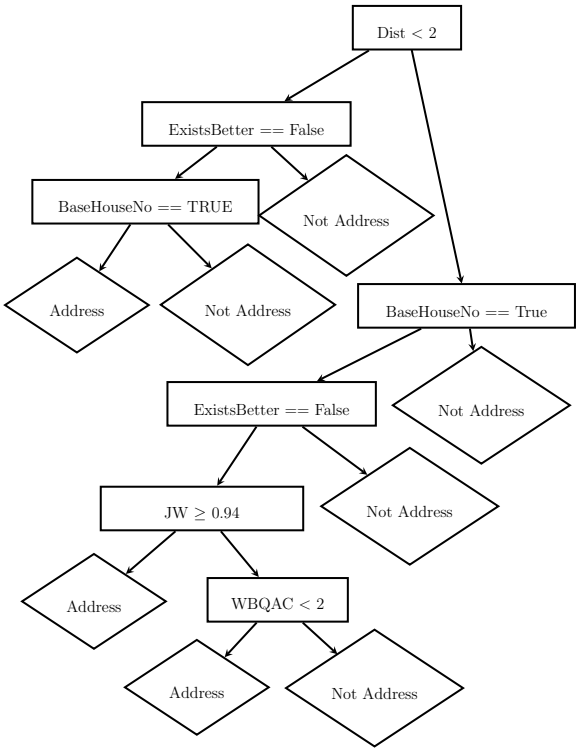


Figure 1: The decision tree for approximate addresses (if a condition is satisfied traverse to the left)

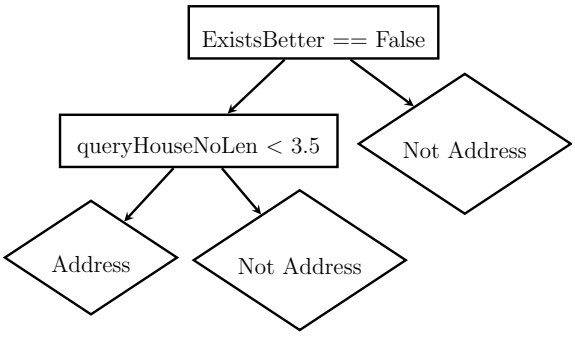


Figure 2: The decision tree for exact addresses (if a condition is satisfied traverse to the left)

Table 4: Precision, Recall and Micro-F1 of decision trees for approximate and exact addresses

| Type | Precision | Recall | Micro-F1 |
|---|---|---|---|
| approximate | 0.918 | 0.947 | 0.933 |
| exact | 0.956 | 0.987 | 0.972 |

Since the *ExistsBetter* variable was tailored to the Croatian language, it is not so surprising that it came out as the most important predictor for exact addresses.

The decision tree for approximate addresses is a bit more complicated. *ExistsBetter* and *JW* look like something one would expect to be useful.

Predictor *BaseHouseNo* is the most interesting one. The ambiguity of possible house numbers and the method of first locating them boost its relevance. The absence of the house number in the database for the specific query usually comes from a query containing bigger numbers than the ones assigned to the candidate, therefore indicating a problem with the query as a semantically correct address.

Precision, Recall and Micro-F1 measures (see [17]) of built classifiers are given in Table 4.

Some might question the choice of decision trees as a classifier. It is known that there exist classifiers (e.g. *SVM*, *Random Forests*) that could give higher precision in certain applications (see [13]). Nevertheless, decision trees were chosen because of their easy interpretability and implementation simplicity (basic if/then).

### 4.3 Application to other countries (languages)

An application to countries with similar languages (similar rules for writing addresses and similar standard language norm), like some Slavic languages, should be possible with small modifications, while the modifications could be more extensive for other countries.

As the first point of the process, an appropriate database containing addresses and cities should be acquired. Of course, ambiguities and various forms of city names may be found more frequently in some other languages, which calls for some disambiguation and fuzzy matching techniques. As stated earlier, postal codes may be used for disambiguation when they exist, along with any other geographic term found in the proximity of city name, or the document itself.

Another option is to use the approach that was presented in Amitay et al. [2].

Having an appropriate training set is of course a necessary condition.

Figure 3: Locations of found semantically correct postal addresses

## 5 Some interesting results

The described method was applied to web documents from the Croatian web space (".hr" extension, *CWeb* in future).

Data set for the analysis was provided by a partial crawl of *CWeb*. At the time of writing this article, around 35,000 domains had been crawled and over 13,000,000 documents had been downloaded. Approximately 4,000,000 addresses were found in 2,750,000 documents.

Around 14,000 domains contained at least one address. In the rest of this section, only these domains will be regarded.

Geographical locations of found addresses are displayed in Fig. 3. It seems that found addresses represent Croatia pretty well, although it is noticeable that some parts of Croatia are a bit sparser than others.

Regarding the appearance of addresses, Table 5 contains top 10 frequencies of the total number of occurrences of an address (the number of all occurrences on the *CWeb* counting one address multiple times in one document). As can be seen, around 29% of addresses appear only once.

Table 5: Top 10 frequencies of total number (#) of occurrences of addresses on the Croatian web

| # | freq. | cumulative sum |
|---|---|---|
| 1 | 0.29262002 | 0.292620202 |
| 2 | 0.16807494 | 0.460695 |
| 3 | 0.13471945 | 0.5954144 |
| 4 | 0.07313523 | 0.6685496 |
| 5 | 0.04057090 | 0.7091205 |
| 6 | 0.03718472 | 0.7463053 |
| 7 | 0.02441533 | 0.7707206 |
| 8 | 0.02363999 | 0.7943606 |
| 9 | 0.01571252 | 0.8100731 |
| 10 | 0.01450995 | 0.824583 |

# 6 Conclusions

The main contribution of this article is in the identification of predictors and the construction of classifiers for ascertaining if a sequence of words represents a postal address. To do so, the difference between semantically correct addresses and syntactically correct addresses is introduced.

The method resulted in precise extraction of semantically correct postal addresses from Croatian web documents.

Since the F1-score in related work was calculated for different languages using different data sets, a comparison with our results might not be indicative. Nevertheless, we report that only Chang et al. [10] presented a higher F1-score on the extraction of Chinese postal addresses.

Although the implementation presented here relies on a commercial database, a free database like OpenStreetMap (`www.openstreetmap.org`) could be used to make the method more approachable to general public.

Application of the method to Web pages and addresses of other countries, an attempt to include unsupervised learning and also an attempt to include the structure of HTML is left for future work.

# Acknowledgements

This article is based on the part of first author's PhD thesis which is written in the partial fulfilment of the requirements for the degree dr.sc. at the University of Zagreb., Faculty of Science under the supervision of prof.dr.sc. Bojan Basrak and prof.dr.sc. Luka Grubišić. The research has been supported by the research grant "Mireo World" from Mireo d.d. (Croatia, `www.mireo.hr`). We gratefully acknowledge the support.

# References

[1] Dirk Ahlers and Susanne Boll. Retrieving address-based locations from the web. In *Proceeding of the 2nd international workshop on Geographic information retrieval*, GIR '08, pages 27–34, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-253-5. doi: http://doi.acm.org/10.1145/1460007.1460015.

[2] Einat Amitay, Nadav Har'El, Ron Sivan, and Aya Soffer. Web-a-where: geotagging web content. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 273–280, New York, NY, USA, 2004. ACM. ISBN 1-58113-881-4. doi: 10.1145/1008992.1009040.

[3] Saeid Asadi, Guowei Yang, Xiaofang Zhou, Yuan Shi, Boxuan Zhai, and Wendy Wen-Rong Jiang. Pattern-based extraction of addresses from web page content. In *Proceedings of the 10th Asia-Pacific web conference on Progress in WWW research and development*, APWeb'08, pages 407–418, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 3-540-78848-4, 978-3-540-78848-5.

[4] Karla A. Borges, Clodoveu A. Davis, Jr, Alberto H. Laender, and Claudia Bauzer Medeiros. Ontology-driven discovery of geospatial evidence in web pages. *Geoinformatica*, 15(4):609–631, October 2011. ISSN 1384-6175. doi: 10.1007/s10707-010-0118-z.

[5] Karla A. V. Borges, Alberto H. F. Laender, Claudia B. Medeiros, and Clodoveu A. Davis, Jr. Discovering geographic locations in web pages using urban addresses. In *Proceedings of the 4th ACM workshop on Geographical information retrieval*, GIR '07, pages 31–36, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-828-2. doi: 10.1145/1316948.1316957.

[6] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

[7] Wentao Cai, Shengrui Wang, and Qingshan Jiang. Address extraction: Extraction of location-based information from the web. In Yanchun Zhang, Katsumi Tanaka, JeffreyXu Yu, Shan Wang, and Minglu Li, editors, *Web Technologies Research and Development - APWeb 2005*, volume 3399 of *Lecture Notes in Computer Science*, pages 925–937. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-25207-8. doi: 10.1007/978-3-540-31849-1_88.

[8] Lin Can, Zhang Qian, Meng Xiaofeng, and Liu Wenyin. Postal address detection from web docu-

ments. In *Proceedings of the International Workshop on Challenges in Web Information Retrieval and Integration*, WIRI '05, pages 40–45, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2414-1.

[9] Chia-Hui Chang and Shu-Ying Li. Mapmarker: Extraction of postal addresses and associated information for general web pages. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '10, pages 105–111, Washington, DC, USA, 2010. IEEE Computer Society. ISBN 978-0-7695-4191-4. doi: 10.1109/WI-IAT.2010.64.

[10] Chia-Hui Chang, Chia-Yi Huang, and Yueng-Sheng Su. On chinese postal address and associated information extraction. In *The 26th Annual Conference of the Japanese Society for Artificial Intelligence*. JSAI, 2012.

[11] Clodoveu A. Davis, Jr and Frederico T. Fonseca. Assessing the certainty of locations produced by an address geocoding system. *Geoinformatica*, 11(1):103–129, March 2007. ISSN 1384-6175. doi: 10.1007/s10707-006-0015-7.

[12] Xianping Ge. Address geocoding, August 2005. U.S. Classification 702/2, 707/E17.018, 702/5; International Classification G06F17/30, G06F19/00, G06Q30/00; Cooperative Classification G06Q30/02, G06F17/30241; European Classification G06Q30/02, G06F17/30L.

[13] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, corrected edition, July 2003. ISBN 0387952845.

[14] Matthew A. Jaro. Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. *Journal of the American Statistical Association*, 84(406): 414–420, 1989. ISSN 0162-1459. doi: 10.1080/01621459.1989.10478785. URL `http://www.tandfonline.com/doi/abs/10.1080/01621459.1989.10478785`.

[15] Jochen L. Leidner. *Toponym Resolution in Text: Annotation, Evaluation and Applications of Spatial Grounding of Place Names*. Universal-Publishers, January 2008. ISBN 9781581123845.

[16] Berenike Loos and Chris Biemann. Supporting web-based address extraction with unsupervised tagging. In Christine Preisach, Hans Burkhardt, Lars Schmidt-Thieme, and Reinhold Decker, editors, *Data Analysis, Machine Learning and Applications*, Studies in Classification, Data Analysis, and Knowledge Organization, pages 577–584. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-78239-1. doi: 10.1007/978-3-540-78246-9_68.

[17] David L. Olson and Dursun Delen. *Advanced Data Mining Techniques*. Springer Publishing Company, Incorporated, 1st edition, 2008. ISBN 3540769161, 9783540769163.

[18] G.R. Rhind. *Global Sourcebook of Address Data Management: A Guide to Address Formats and Data in 194 Countries*. Gower, 1998. ISBN 9780566081095.

[19] Benjamin E. Teitler, Michael D. Lieberman, Daniele Panozzo, Jagan Sankaranarayanan, Hanan Samet, and Jon Sperling. NewsStand: a new view on news. In *Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '08, pages 18:1–18:10, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-323-5. doi: 10.1145/1463434.1463458. URL `http://doi.acm.org/10.1145/1463434.1463458`.

[20] Chuang Wang, Xing Xie, Lee Wang, Yansheng Lu, and Wei-Ying Ma. Detecting geographic locations from web resources. In *Proceedings of the 2005 workshop on Geographic information retrieval*, GIR '05, pages 17–24, New York, NY, USA, 2005. ACM. ISBN 1-59593-165-1. doi: 10.1145/1096985.1096991.

[21] William E. Winkler. String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. *Proceedings of the Section on Survey Research Method*, pages 354–359, January 1990. URL `http://eric.ed.gov/?id=ED325505`.

[22] Zheyuan Yu. High Accuracy Postal Address Extraction From Web Pages. Master's thesis, Dalhousie University, Halifax, Nova Scotia, 2007.