In *SIGIR'12: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 731–740, Portland, OR, August 2012.

Adaptive Context Features for Toponym Resolution in Streaming News*

Michael D. Lieberman Hanan Samet
Center for Automation Research, Institute for Advanced Computer Studies,
Department of Computer Science, University of Maryland
College Park, MD 20742
{codepoet, hjs}@cs.umd.edu

ABSTRACT

News sources around the world generate constant streams of information, but effective streaming news retrieval requires an intimate understanding of the geographic content of news. This process of understanding, known as *geotagging*, consists of first finding words in article text that correspond to location names (*toponyms*), and second, assigning each toponym its correct lat/long values. The latter step, called *toponym resolution*, can also be considered a classification problem, where each of the possible interpretations for each toponym is classified as correct or incorrect. Hence, techniques from supervised machine learning can be applied to improve accuracy. New classification features to improve toponym resolution, termed *adaptive context features*, are introduced that consider a window of context around each toponym, and use geographic attributes of toponyms in the window to aid in their correct resolution. Adaptive parameters controlling the window's *breadth* and *depth* afford flexibility in managing a tradeoff between feature computation speed and resolution accuracy, allowing the features to potentially apply to a variety of textual domains. Extensive experiments with three large datasets of streaming news demonstrate the new features' effectiveness over two widely-used competing methods.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

General Terms

Algorithms, Design, Performance

Keywords

Toponym resolution, geotagging, streaming news, adaptive context, machine learning

*This work was supported in part by the National Science Foundation under Grants IIS-10-18475, IIS-09-48548, IIS-08-12377, and CCF-08-30618.

SIGIR'12, August 12–16, 2012, Portland, Oregon, USA.

1. INTRODUCTION

Today's increasingly informed and connected society demands ever growing volumes of news and information. Thousands of newspapers, and millions of bloggers and tweeters around the world generate constant streams of data, and the demand for such data is skyrocketing as people strive to stay up-to-date. Also, Internet-enabled mobile devices are increasingly common, which expands the requirement for location-based services and other highly local content—information that is relevant to where users are, or the places in which they are interested. News itself often has a strong geographic component, and newspapers tend to characterize their readership in terms of location, and publish news articles describing events that are relevant to geographic locations of interest to their readers. We wish to collect these articles and make them available for location-based retrieval queries, which requires special techniques.

To enable news retrieval queries with a geographic component, we must first understand the geographic content present in the articles. However, currently, online news sources rarely have articles' geographic content present in machine-readable form. As a result, we must design algorithms to understand and extract the geographic content from the article's text. This process of extraction is called *geotagging* of text, which amounts to identifying locations in natural language text, and assigning lat/long values to them. Put another way, geotagging can be considered as enabling the spatial indexing of unstructured or semistructured text. This spatial indexing provides a way to execute both feature-based queries ("Where is X happening?") and location-based queries ("What is happening at location Y?") [5] where the location argument is specified textually rather than geometrically as in our related systems such as QUILT [34] and the SAND Browser [31]. Geotagging methods have been implemented in many different textual domains, such as Web pages [3, 23, 27], blogs [28], encyclopedia articles [14, 36], tweets [33], spreadsheets [2, 17], the hidden Web [19], and of most relevance for us, news articles [8, 11, 18, 20, 21, 29, 32, 37]. Particular domains such as blogs and tweets may pose additional challenges, such as having few or no formatting or grammatical requirements. The methods in this paper were applied in the NewsStand system [37], which uses a geotagger to assign geographic locations to clusters of news articles based on their content, which allows users to visually explore the news in NewsStand's interactive map interface. Also, several commercial products for geotagging text are available, such as

MetaCarta's Geotagger[1], Thomson Reuters's OpenCalais[2], and Yahoo!'s Placemaker[3], the latter two of which we investigate here.

Geotagging consists of two steps: finding all textual references to geographic locations, known as *toponyms*, and then choosing the correct location interpretation for each toponym (i.e., assigning lat/long values) from a *gazetteer* (database of locations). These two steps are known as *toponym recognition* and *toponym resolution*, the second of which we investigate here, and are difficult due to ambiguities present in natural language. Importantly, both these steps can be considered as *classification* [10] problems: Toponym recognition amounts to classifying each word in the document's text as part of a toponym or not, and toponym resolution amounts to classifying each toponym interpretation as correct or incorrect. With this understanding, and with appropriately annotated datasets, we can leverage techniques from supervised machine learning to create an effective geotagging framework. These techniques take as input sets of values known as *feature vectors*, along with a class label for each feature vector, and learn a function that will predict the class label for a new feature vector. Many such techniques for classification, and other machine learning problems, exist and have been used for geotagging purposes, including SVM [4, 13, 24], Bayesian schemes [9, 12, 41], and expectation maximization [6].

The effectiveness of such techniques for a given problem domain depends greatly on the design of the input *features* that comprise each feature vector. One common feature used for geotagging is the population of each interpretation, since larger places will tend to be mentioned more frequently and are more likely to be correct. However, using population alone or overly relying on it, as many methods do, resulted in greatly reduced accuracy in our experiments, especially for toponym recall. Instead, in this paper, we consider a new class of features to improve the accuracy of toponym resolution, termed *adaptive context features*. These features construct a window of variable size around each toponym t, and use the other toponyms in the window to aid in resolving t correctly by considering the geographic relationships between interpretations l_t of t and those of other toponyms in the window. In particular, we search for interpretations that are geographically *proximate* to l_t, or are *siblings* of l_t in terms of a geographic hierarchy (e.g., cities in the same state). The more such relationships appear in the window, the greater evidence there is that l_t is the correct interpretation of t. These window features are a natural extension of other *context-sensitive* features which depend on other words nearby the toponym, such as *object/container* and *comma group* [20] evidence, as well as pairing notions such as *pair strength* [19].

We call these features *adaptive* because the window's parameters can be varied for different domains, or to achieve different ends. Some relatively small windows can contain a significant number of highly ambiguous toponyms, especially for toponyms at fine scales [26], and considering all possible combinations of interpretations places inhibitive penalties on feature computation speed. For example, consider Figure 1, which is an excerpt from a press release [25] published in the Earth Times newspaper, with toponyms highlighted and the numbers next to each toponym indicating the number of interpretations in our gazetteer for the toponym. If

> ... in and around [*Louisville* 17] and
> [*Lexington* 31], [*Kentucky* 6], [*Nashville* 27] and
> [*Cordova* 55], [*Tennessee* 5], [*Richmond* 69],
> [*Virginia* 42], [*Fort Lauderdale* 1] and
> [*Orlando* 9], [*Florida* 96], [*Indianapolis* 3],
> [*Indiana* 8] and [*Atlanta* 22], [*Georgia* 12].

Figure 1: Excerpt from an Earth Times press release [25] with toponyms and their number of interpretations highlighted, showing the extreme ambiguity of these toponyms and illustrating the need for adaptive context features.

we consider all possible combinations of resolutions for these toponyms, this results in about $3{\cdot}10^{17}$ possibilities, an astonishingly large number for this relatively small portion of text, which is far too many to check in a reasonable time. Instead, we can set parameters which we term the window's *breadth* and *depth*, named analogously to breadth-first and depth-first search, which control the number of toponyms in the window and the number of interpretations examined for each toponym in the window, respectively. The adaptive context features thus afford us flexibility since by varying these parameters, we can control a tradeoff between feature computation time and resolution accuracy. The more toponyms and toponym interpretations we examine, the more likely we are to find the correct interpretation, but the longer resolution will take, and vice versa. Some textual domains such as Twitter, where tweets arrive at a furious rate, demand faster computation times, while in other, offline domains, the time constraint is relaxed and we can afford to spend more time to gain higher accuracy. While window-like features and heuristics have been used in other work related to geotagging (e.g., [15, 16, 24, 30, 35, 42]), these features' adaptive potential has not been explored.

As we pointed out, in this paper our focus is on toponym resolution, while toponym recognition makes use of our previous work [18]. Our work differs from that of others in a number of ways, including our focus on streaming data, and most importantly the size of the domain of the data used in our evaluation. In particular, in our evaluation we make use of articles from a large set of news sources, many of which are from small localities, and the domain of locations is very large, as evidenced by our gazetteer containing 8.1M location interpretations, in contrast to systems such as Web-a-Where [3] which contains only about 30,000 location interpretations.

The rest of this paper is organized as follows: Section 2 introduces the geotagging framework that enables us to test our adaptive context features, and describes our toponym recognition and resolution processes as a whole. We also introduce several other features that complement our adaptive context features and serve as baselines for comparison. Section 3 describes our new adaptive context features, as well as algorithms for their computation. Section 4 details extensive experiments showing our methods' performance benefits over OpenCalais and Placemaker, that also test various feature combinations and parameters. Finally, Section 5 offers potential avenues of future work and concludes.

2. GEOTAGGING FRAMEWORK

In this section, we present the framework that enables testing of our geotagging methods. This framework was originally developed for and is an integral component of the NewsStand [37] and TwitterStand [33] systems. We describe our toponym recognition (Section 2.1) and resolution (Sec-

[1] `http://metacarta.com`
[2] `http://opencalais.com`
[3] `http://developer.yahoo.com/geo/placemaker`

tion 2.2) procedures, as well as a set of baseline features (Section 2.3) that we use in combination with our adaptive context features, which will be presented in Section 3.

2.1 Toponym Recognition

Our toponym recognition procedure is designed as a multifaceted process involving many types of recognition methods, both rule-based and statistics-based. After an initial tokenization step, our method proceeds by performing lookups into various tables of entity names, including location names and abbreviations (e.g., "Maryland", "Md."), business names (e.g., "Apple", "Toyota"), person names (e.g., "Chad", "Victoria"), as well as cue words for the above types of entities (e.g., "X County", "Mr. X", "X Inc."). We also refactor geographic names by shifting particular cue words (e.g., "X Lake" to "Lake X").

In addition to the above rule-based methods, we leverage statistical NLP tools. We use an NER package to recognize toponyms and other entities, and perform extensive postprocessing on its output to ensure higher quality. We also perform part-of-speech (POS) tagging to find phrases of proper nouns, since names of locations (and other types of entities) tend to be composed of proper nouns. The POS tagging also provides a means of recognizing additional grammatical forms that hint at entities' types, including active verbs and noun adjuncts, which we use as signals to adjust entity types. Furthermore, we incorporate evidence from other documents in the document's news cluster. After the above recognition steps, we establish groups of entities to be resolved at the same time, by grouping similar entities together.

This multifaceted recognition procedure is designed to be flexible to capture variations that appear in streaming news, and also to be as inclusive as possible when recognizing toponyms, to maximize toponym recall, which comes at the cost of lower precision. Our recognition procedure's high recall is also corroborated by experimental results in Section 4.3. Since toponym recognition is only the first step in a two-part geotagging process, our toponym resolution methods will serve to restore precision to the entire process.

2.2 Toponym Resolution

As mentioned earlier, geotagging can be understood as a classification problem, and we use methods from supervised machine learning to implement toponym resolution. Specifically, we cast it as a *binary* classification problem, in that we decide for a given toponym/interpretation pair (t, l_t), whether l_t is correct or incorrect. These location interpretations are drawn from a *gazetteer*, which is a database of locations and associated metadata such as population data and hierarchy information. Our gazetteer, which is based on GeoNames[4], is vastly larger than many gazetteers typically used in geotagging methods, which both increases our methods' utility as well as geotagging's difficulty. We characterize our gazetteer further in our experiments in Section 4.1.

For our classifier, we use a decision tree-based ensemble classifier method known as *random forests* [7], which has state-of-the-art performance for many classification tasks. Briefly, given an annotated training dataset, the random forests method constructs many decision trees based on different random subsets of the dataset, sampled with replacement. In addition, each decision tree is constructed using random subsets of features from the training feature vectors. Because the features and subsets are chosen randomly, a variety of trees will be in the forest. Classifying a new feature vector is relatively simple: each tree in the forest votes for the vector's class, and the consensus is taken as the result. Note that individual trees may be excellent or poor class predictors, but as long as some features allow better-than-random classification, the forest taken as a whole will be a strong classifier. Another useful aspect of random forests is that the number of trees that vote for a given class can be used as a confidence score for the classification, and provides a means of tuning the precision/recall balance of the classifier. Assuming the score is an accurate estimate of the method's predictability, accepting classifications with a lower score will result in lower precision but higher recall, and vice versa. For our implementation, we used the fast random forest implementation[5], integrated with the Weka machine learning toolkit[6].

As an alternative to classification, Martins et al. [24] considered the use of SVM *regression* to estimate a distance function based on feature vector values that is intended to capture the distance between a given l_t, and t's ground truth interpretation. They use the resulting distance values to rank the interpretations, essentially using them as confidence scores, and select the one with smallest distance value as the interpretation for t. However, a significant drawback of this technique is that it assumes that all toponyms input to the toponym resolution process are not erroneous, i.e., that the toponym recognition procedure is perfect in identifying toponyms, while in reality, no such procedure is perfect. The distance measures they compute, while useful for ranking, are not necessarily meaningful as confidence scores for deciding whether a given l_t has strong enough evidence to consider it correct. For example, an inferred distance of "10" may indicate strong evidence for a given l_t, but weak evidence for another. On the other hand, our framework using random forests and their confidence scores provide consistent and meaningful scores for deciding classification strength.

2.3 Resolution Features

In addition to the adaptive context features introduced in the next section, we use several baseline toponym resolution features in our methods. To borrow terms from linguistics, these features, which will be computed for each toponym/interpretation pair (t, l_t), can be loosely classed as what we term *context-sensitive* and *context-free* features. Put simply, context-sensitive features depend on t's position in relation to other toponyms in the document, while context-free features do not. Note that our adaptive context features subsume and generalize context-sensitive features, so we will describe them in the next section. On the other hand, the context-free features we use include the following:

I: `interps`. Number of interpretations for t; more interpretations means more opportunities for errors.

P: `population`. The population of l_t, where a larger population indicates that l_t is more well-known.

A: `altnames`. Number of alternate names for l_t in various languages. More names indicates greater renown of l_t.

D: `dateline`. Geographic distance of l_t from an interpretation of a *dateline* toponym, which establishes a general location context for a news article.

L: `locallex`. Geographic distance of l_t from the newspaper's *local lexicon* [21], the expected location of its primary audience, expressed as a lat/long point.

[4] `http://geonames.org`

[5] `http://code.google.com/p/fast-random-forest`

[6] `http://www.cs.waikato.ac.nz/ml/weka`

The `interps`, `population`, and `altnames` features are domain independent, i.e., they can be used in any textual domain, while the `dateline` and `locallex` features are specific to the news domain. In our experiments in Section 4.4, we consider these features alone and in various combinations to understand each feature's relative utility.

3. ADAPTIVE CONTEXT FEATURES

In this section, we present our adaptive context features to aid in the resolution of toponyms. These features reflect two aspects of toponym coocurrence and the evidence that interpretations impart to each other, which include:

1. *Proximate* interpretations, which are both nearby in the text as well as geographically proximate, and
2. *Sibling* interpretations, which are nearby in the text and share containers in a geographic hierarchy.

We capture these interpretation relationships and encode them in features. To compute these features, we examine for each toponym t a window of text around t, and compare interpretations of toponyms in the window with the interpretations of t. That is, a given interpretation l_t of t is promoted if there are other interpretations of toponyms in the window that are geographically proximate to it, or are sibling interpretations. In addition, we vary two parameters of the window termed *window breadth* and *window depth*, which control a tradeoff between computation speed and discriminative utility for the features by changing the number of toponyms in the window, and the number of interpretations per toponym, respectively.

Figure 2 is a schematic representation of the algorithm used to compute our features. Each box represents a toponym, and the lines under the boxes represent location interpretations for each toponym. Different toponyms have different levels of ambiguity, as measured by the number of interpretations for the toponyms. In Figure 2, we are computing adaptive context features for the highlighted toponym and its interpretations in the middle. We compute these features for all toponyms at a document distance of less than the window breadth w_b, and we compare the first few interpretations of each toponym in the window, up to a maximum of w_d interpretations, the window depth.

Note that our proximity and sibling features subsume and generalize other commonly-used features in toponym resolution. In particular, these features generalize *context-sensitive* features, which compute a toponym interpretation's likelihood of correctness based on the other toponyms nearby to it in the document. One example of these context-sensitive features includes the *object/container* pair (e.g., *Paris, Texas*, *Dallas in Texas*), consisting of two toponyms, one of which contains the other. Authors use them when their audiences are not familiar with the location in question, and use the containing toponym to provide a geographic context for the toponym. Object/container pairs are a particularly common type of evidence used in many types of documents, and much research has investigated its utility (e.g., [3, 16, 21, 30, 35]). This type of evidence can be understood as an extreme case of our sibling feature, in the case where the window is restricted to the immediately next or preceding toponym. More general than object/container pairs is the *comma group* [20], which consists of a sequence of toponyms adjacent to each other separated by connector tokens (e.g., "*Paris*, *Dallas*, *San Antonio* and *Houston*") that share geographic characteristics (in our example, all cities in Texas), and hence provide mutual evidence for each others' correct interpretations. These relationships can be captured using our features with a window of appropriate size to contain all the toponyms. Another difference between these sources of evidence and our adaptive context features is that we do not assume any meaning for the specific position of toponyms within the window. For example, we do not consider the grammatical structure involved, or the tokens present between toponyms in the window. This increases the flexibility of our features as compared to, e.g., comma groups, whose recognition depends on specific wording and organization of the toponyms. As noted by Lieberman et al. [20], comma groups in particular can be constructed in various ways that can mislead rule-based heuristics, such as in our original example in Figure 1.

The following sections describe our proximity (Section 3.1) and sibling (Section 3.2) features, and the algorithms we use to compute them (Section 3.3). We also describe a strategy for propagating significant feature values for a toponym to its other instances in the same document (Section 3.4).

3.1 Proximity Features

The *proximity* features we use are based on geographic distance. Because this distance is continuous, appropriate thresholds for what is considered "near" and "far" are not apparent. Thus, it behooves us to defer their definitions to learning algorithms that can learn appropriate and meaningful distance thresholds.

To compute our proximity features for a toponym/interpretation pair (t, l_t), we find for each other toponym o in the window around t the closest interpretation l_o to l_t. Then, we compute the proximity feature for (t, l_t) as the average of the geographic distances to the other interpretations. Thus, a lower feature score indicates a higher level of overall geographic proximity for toponyms within the window. This feature strategy also balances fairness with optimism, in that it allows all toponyms in the window to contribute to (t, l_t)'s feature score, while each toponym in the window contributes its best (i.e., geographically nearest) interpretation to the feature score. It has the additional benefit that no distance thresholds are hard-coded into the feature. Instead, the learning procedure can learn appropriate distance thresholds from its training data.

3.2 Sibling Features

Our second class of adaptive context features are those based on *sibling* relationships between interpretations in a geographic hierarchy. In other words, this feature will capture the relationships between textually proximate toponyms that share the same country, state, or other administrative division. The sibling feature is intended to capture interpretations that are at the same level in the hierarchy (e.g., states in the same country, cities in the same state) as well as interpretations at different levels (e.g., a state and its containing country, a city inside its containing state). The first case captures "true" sibling relationships, while the second case captures containment relationships, which can be considered siblings at a coarse granularity (e.g., *College Park* and *Maryland* are siblings at a state level of granularity).

We compute sibling features in a similar way as the proximity features. For each toponym/interpretation pair (t, l_t), we use as our sibling feature value the number of other toponyms o in the window around t with an interpretation that is a sibling of l_t at a given resolution. We consider three levels of resolution, which correspond to three sibling features for each (t, l_t): country-level, state-level, and county-level.

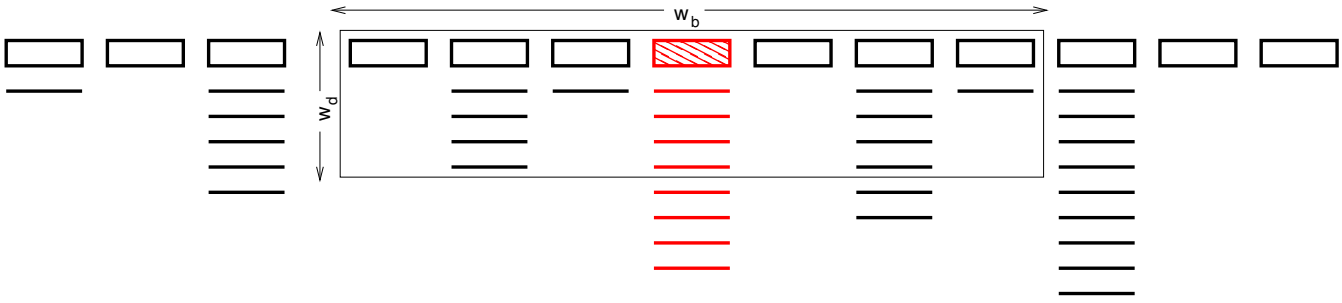


Figure 2: Computing adaptive context features, illustrating the window breadth w_b and window depth w_d.

Given that the sibling features are so related to the proximity features, at first glance, the sibling feature appears to be redundant in that some toponym interpretations that are siblings will tend to be geographically proximate as well (e.g., *Paris, Texas* and *Dallas, Texas*). However, in some cases the sibling feature will prove helpful in distinguishing toponym relationships. For example, cities that are positioned at opposite ends of a given state might be too far to be considered geographically proximate, but would still be considered siblings. Similarly, the notion of geographic distance for area objects such as countries and states depends on their representation. If we represent, e.g., a country by a single point, such as its centroid or the location of its capital city, it might be considered geographically distant from many cities contained in it, while the sibling feature would correctly capture these relationships. Another difference between the proximity and sibling features is that the geographic hierarchy is discrete, while geographic distances are continuous values. As a result, we do not have the same thresholding problem as for the proximity features, as our "thresholds" are effectively the same as the hierarchy levels.

3.3 Feature Computation

As noted earlier, our adaptive context features are based on computing features within a *window* of context around each toponym t. We can consider two variables related to the search for a correct interpretation of t:

1. *Window breadth*, denoted w_b, which corresponds to the size of the window around t to be considered.
2. *Window depth*, denoted w_d, which is the maximum number of interpretations to be considered for each toponym in the window.

The window breadth w_b controls how many toponyms around a given toponym t are to be used in aiding the resolution of t. With a larger w_b, more toponyms will be used to resolve t, thus reducing the resolution algorithm's speed but hopefully increasing its accuracy. Similarly, the window depth w_d controls the number of interpretations to be considered for each toponym in the window. A larger w_d means that more interpretations will be checked, with a resulting decrease in speed, but with more potential for finding corroborating evidence for a correct interpretation of t.

Because the window depth may preclude examination of all interpretations for a given toponym, the order in which the interpretations are examined is important. Ideally, interpretations would be ordered using context-free attributes of each interpretation. In a sense, the ordering is based on an apriori estimate of each interpretation's probability of being mentioned in a given document, though we do not formalize this notion here. We order or rank these interpretations using various factors, which include, in order of importance:

1. Number of alternate names for the location in other languages. GeoNames, being a multilingual gazetteer, contains alternate names and the number of names can indicate the place's renown.
2. Population of the location, where a larger population generally indicates a more well-known place.
3. Geographic distance from a local lexicon location.

These ranking factors can be considered context-free, in that the ordering of interpretations for a given toponym is independent of its position in the document. We could use additional factors such as each interpretation's geographic distance from a dateline toponym interpretation, but because we include these factors as separate features we do not need to include them in the ranking here.

One seeming drawback with regard to the window depth is that it may not seem effective in that most toponyms in our gazetteer have only one or two possible interpretations, as our experiments in Section 4.1 show. However, toponyms that are well-known by virtue of having a well-known interpretation (for example, *Paris*, widely known as the French capital), will tend to be mentioned more frequently in documents, and these will be more ambiguous. This is also reflected in measurements made on the toponyms present in our experimental datasets (Section 4.2).

In addition, rather than using all toponyms in the window around each t, we perform some pre-filtering to remove toponyms that detract from the usefulness of our adaptive context features. For example, we do not use toponyms in the window that have the same name as t, since they will have the same set of interpretations as t, which will impart no useful information. In addition, and more generally, we may not be sure which of the words in the window correspond to toponyms, due to ambiguities in toponym recognition. Our toponym recognition process (described in Section 2.1) is designed for high recall and as a result we will consider many words which are not true toponyms. In other cases, we may not be sure of the appropriate interpretation for a given toponym. For example, consider the phrase "University of Maryland", which could be interpreted as a whole, *University of Maryland*, referring to the school, or as *Maryland*, the state. Rather than immediately deciding on one of these toponyms, our recognition process keeps both, even though they overlap. Thus, we can keep and consider all of them in toponym resolution, though they must be appropriately filtered when processing toponyms in the window.

Our algorithm for computing adaptive context features, called AdaptiveContext, is shown in Algorithm 1. It takes as input the toponyms T in the document being processed, as well as the window breadth w_b and window depth w_d under consideration. The algorithm proceeds by iterating over all toponyms $t \in T$ (line 2). For each t, an array

Algorithm 1 Compute adaptive context features.

```
1: procedure AdaptiveContext(T, w_b, w_d)
   input: Topos T, window breadth w_b and depth w_d
   output: Proximity and sibling features
2:     for t ∈ T do
3:         P ← {}
4:         O ← {o ∈ T : Name(t) ≠ Name(o) ∧
               DocDist(t, o) ≤ w_b}
5:         for o ∈ O do
6:             L_o ← Locs(o)[1 ... min{w_d, |Locs(o)|}]
7:             for l_t ∈ Locs(t) do
8:                 d_min ← min{∀l_o ∈ L_o, GeoDist(l_t, l_o)}
9:                 P[l_t] ← P[l_t] ∪ {d_min}
10:                for lev ∈ {country, admin1, admin2} do
11:                    if ∃l_o ∈ L_o : Sibling(l_t, l_o, lev) then
12:                        Increment SibFeature(t, l_t, lev)
13:                    end if
14:                end for
15:            end for
16:        end for
17:        for l_t ∈ Locs(t) do
18:            ProxFeature(t, l_t) ← Avg(P[l_t])
19:        end for
20:    end for
21: end procedure
```

P is initialized which will hold minimum distances to interpretations of toponyms in the window around t, which will be used in computing the proximity features for t (3). Next, other toponyms O within the window around t are collected by finding toponyms $o \in T$ whose document distance is smaller than the window breadth w_b, and also have a different name than t (4). We then loop over each toponym $o \in O$ to begin comparing interpretations of t and o (5). First, we collect the location interpretations associated with o, up to a limit of w_d interpretations, the window depth (6). Then, we loop over each interpretation l_t of t (7), and find the interpretation l_o of o with minimum geographic distance from l_t (8). We add the interpretation l_o to the location set $P[l_t]$ associated with l_t which will be used for computation of the proximity feature for l_t (9). Next, we compute the sibling features for each level lev of our geographic hierarchy (10) by checking whether there exists an interpretation l_o of o with l_t as its sibling (11). If so, we increment the sibling feature for that level (12). Finally, after looping over all toponyms of O, the sibling features are fully computed for each interpretation l_t of t, but the proximity feature remains to be completed. We do so for each l_t (17) by averaging the geographic distances computed for l_t, which results in the final proximity feature values (18). We use the median geographic distance as our averaging measure.

3.4 Feature Propagation

Oftentimes, documents will mention the same toponyms multiple times. When considering pairs of toponyms for use in computing adaptive context features, described in the previous section, these toponym repetitions are ignored because they impart no useful information, since the interpretations for each pair will be the same. However, we can still make use of toponym repetition within a single document because the toponyms appear in different contexts (i.e., at different offsets) within the document. Since our adaptive context features are context-sensitive, we can apply stronger feature values computed for the toponym in one context to the same toponym in other, weaker contexts.

To leverage these repetitions, as a final processing step, we compute additional features for each (t, l_t) pair by *propagating* feature values among toponyms in the document that share the same name. We propagate feature values that indicate strong evidence that a given toponym interpretation is correct. For the proximity feature, this corresponds to the lowest average distance values, while for the sibling features, we propagate the largest sibling counts for each level of resolution we consider.

4. EXPERIMENTS

In this section, we describe the extensive experiments performed on our own and competing geotagging methods. We first establish the general difficulty of geotagging using our large gazetteer, due to a large amount of toponym ambiguity (Section 4.1), and then introduce the datasets to be used for measuring geotagging performance, and characterize the toponyms present in them (Section 4.2). In terms of geotagging accuracy, we compare our own adaptive method, referred to as "Adaptive", against two existing prominent competing methods: Thomson Reuters's OpenCalais, and Yahoo!'s Placemaker. Both OpenCalais and Placemaker are closed-source commercial products, but they do provide public Web APIs which allow for automated geotagging of documents, and hence they have been used extensively in state-of-the-art geotagging and entity recognition research (e.g., [1, 24, 28, 38, 40]). In addition to not being able to make direct algorithmic comparisons due to their black box nature, neither OpenCalais nor Placemaker offer a means of tuning the precision/recall balance, so we could not explore this aspect of the systems. We discuss how well these systems fare against our own methods in terms of toponym recognition (Section 4.3) and toponym resolution (Section 4.4). For the latter, we also consider various combinations of features and show how they affect resolution accuracy, and use a feature ranking method to measure the importance of each feature when used in resolving toponyms. Finally, we vary the adaptive context parameters of window breadth (w_b) and depth (w_d), and show how they affect the feature computation time and accuracy of the Adaptive method (Section 4.5).

Note that in all our accuracy experiments, we measure performance using *precision* and *recall* as measured over the correct interpretations. We also used 10-fold cross validation to avoid misleading performance numbers due to potential overfitting. Also, we used 100 trees in our random forests, with 5 attributes for each tree, and accepted classifications with at least 0.5 confidence score (i.e., at least half of the trees voted for the interpretation). All experiments were conducted on a Dell Precision 470 workstation with a dual core Intel Xeon 3GHz CPU and 8G RAM.

4.1 Gazetteer Ambiguity

First, we examined our gazetteer to understand the level of ambiguity of toponyms present in it. The gazetteer contains a total of 8.1M location interpretations, 5.1M distinct names, and 7.0M alternate names in languages other than English. The gazetteer's large size ensures a high level of ambiguity and ensuing greater difficulty in performing geotagging correctly, when compared to gazetteers used by other systems such as Web-a-Where [3]. For each toponym in the gazetteer, we counted the number of interpretations associated with it, and plotted the results. Results are shown in Figure 3. Toponyms in the gazetteer exhibit a power-law relationship in terms of the number of interpretations, in that

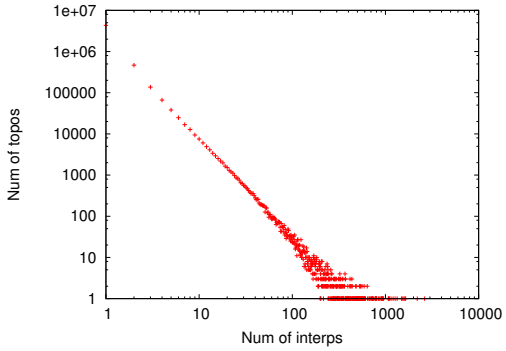


Figure 3: Toponyms and the number of interpretations they have exhibit a power-law relationship.

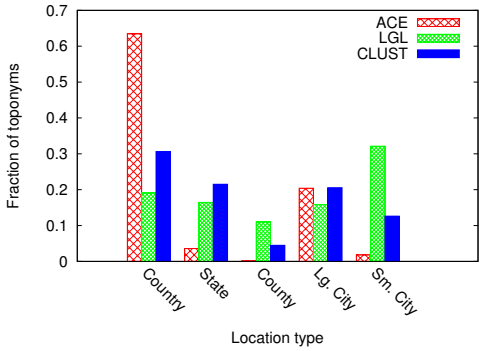


Figure 4: Breakdown of location types within each of our test corpora.

Table 1: Corpora used in evaluating geotagging.

	ACE	*LGL*	*CLUST*
Documents	104	621	13327
Median doc word count	236	242	309
News sources	4	114	1607
Annotated docs	104	621	1080
Annotated topos	2359	4765	11564
Distinct topos	295	1177	2320
Median topos per doc	12	6	8
Median topo ambig per doc	3	14	7

the vast majority of toponyms a small number of interpretations, while a few toponyms have a very large number of interpretations. Of course, most of these unambiguous toponyms will not be mentioned in a given document, and in our datasets, described in the next section, the documents' toponyms have higher levels of ambiguity.

4.2 Datasets

In choosing the datasets for our evaluation, we wanted news data from a variety of sources, and for a variety of audiences. To achieve this end, we used three datasets of news in our evaluation: *ACE*, *LGL*, and *CLUST*. The first, *ACE* [22], consists of articles from four large news sources: Agence France-Presse, Associated Press World, New York Times, and Xinhua. These articles tend to have a broad world interest and concern topics such as international diplomacy and trade, so they tend to mention large, well-known places. Thus, *ACE* serves in our evaluation as a test of the geotagging methods' capability for correctly recognizing and resolving well-known, prominent places. On the other hand, to test smaller places, we used the *LGL* [21] dataset, which consists of articles from about 100 smaller, more local news sources. These articles are intended for more geographically localized audiences, and concern local events that mention small places. Our third dataset, *CLUST* [18], contains a variety of articles from both large and small news sources.

Table 1 presents statistics that broadly illustrate characteristics of our three test corpora. *ACE* is relatively small compared to *LGL* and *CLUST*, both in terms of number of documents and news sources. However, *ACE* tends to have more toponyms per article, which may be due to the content consisting of generally international news involving many different countries and other locations, which would all be mentioned in the articles. In addition, we measured toponym ambiguity in the articles by checking, for each document, the median number of gazetteer interpretations for the toponyms in the document. the median number of interpretations present for toponyms in each document. *LGL* has the largest amount of toponym ambiguity, followed by *CLUST* and *ACE*. This is not overly surprising, given that *LGL* was constructed deliberately focusing on highly ambiguous toponyms [21]. However, the measurements show a high level of ambiguity in all three datasets.

We also classified the annotated locations present in the documents according to their types, which are shown in Figure 4. We normalized the type counts for each corpus to illustrate the fractions of each type within each corpus. For cities, we further divided the locations into large cities (over 100k population) and small cities (less than 100k population). These location types clearly show the important differences between the three corpora. The vast majority of *ACE*'s toponyms, 83%, consist of countries and large cities, indicating *ACE*'s broad geographic scope. This is not overly surprising given that it consists of newswire, which is usually intended for a broad geographic audience. In contrast, 60% of *LGL*'s toponyms are small cities, counties, and states, and among all three datasets, *LGL* contains the smallest fraction of countries and large cities, showing that *LGL* mainly concerns smaller, more local places, with a correspondingly smaller geographic audience. *CLUST* falls in the middle, with the largest fraction of states among the three datasets, and in between the other two in terms of countries, counties, and small cities. Bearing these observations in mind, in terms of overall geographic relevance, *ACE* and *LGL* can be said to have wide and narrow relevance respectively, while *CLUST* falls in the middle, illustrating our three datasets' utility in testing geotagging at coarse, middle, and fine-grained geographic scopes.

4.3 Recognition Accuracy

Though the main focus of this paper is improved toponym resolution, for completeness, we tested each system's toponym recognition performance when isolated from the subsequent toponym resolution step. Note that OpenCalais and Placemaker also provide lat/long values with each toponym, but we disregard these when testing toponym recognition using these systems because it is more information than we need for this experiment. Table 2 shows the performance results for each method's toponym recognition step. For all three datasets, the Adaptive method shows higher recall performance than either OpenCalais or Placemaker, which as we discussed earlier is the crucial measure to consider for toponym recognition used before toponym resolution, as well as a higher overall F_1-score for *LGL* and *CLUST*. While OpenCalais and Placemaker do have higher precision, this

Table 2: Recognition performance.

	P	R	F_1
ACE			
Adaptive	0.748	0.867	0.804
OpenCalais	0.883	0.681	0.769
Placemaker	0.899	0.767	0.828
LGL			
Adaptive	0.671	0.723	0.696
OpenCalais	0.588	0.222	0.322
Placemaker	0.675	0.658	0.666
CLUST			
Adaptive	0.732	0.861	0.791
OpenCalais	0.759	0.425	0.545
Placemaker	0.798	0.692	0.741

Table 3: Resolution accuracy of various methods.

	P_{Resol}	$R_{Recog+Resol}$
ACE		
Adaptive	1635/1659 = 0.986	1635/2359 = 0.693
OpenCalais	1062/1080 = 0.983	1062/2359 = 0.450
Placemaker	1161/1219 = 0.952	1161/2359 = 0.492
LGL		
Adaptive	2799/2970 = 0.942	2799/4765 = 0.587
OpenCalais	1260/1632 = 0.772	1260/4765 = 0.264
Placemaker	2516/3466 = 0.726	2516/4765 = 0.528
CLUST		
Adaptive	7143/7440 = 0.960	7143/11564 = 0.618
OpenCalais	5397/6352 = 0.850	5397/11564 = 0.467
Placemaker	7524/8642 = 0.871	7524/11564 = 0.650

is mitigated by their relative lack of recall. Also, Adaptive's precision is restored by its toponym resolution processing, which will be shown in the next section. These results are also consistent with previously-reported performance [18].

4.4 Resolution Accuracy

In the next experiment, we measured the accuracies of each method's toponym resolution in isolation—that is, if each method were given a set of toponyms, how well the method would select the correct lat/long interpretation for each toponym. Because OpenCalais and Placemaker do not allow for the specification of ground truth toponyms, it is not possible to make direct comparisons of toponym resolution's recall for these systems. Instead, we report the precision for the resolution process in isolation (P_{Resol}), and the recall for the combined recognition and resolution processes ($R_{Recog+Resol}$). For P_{Resol}, we only report accuracy for toponyms that were correctly recognized by each system. Also, for the Adaptive method, we used a window breadth w_b of 80 tokens and unlimited window depth w_d. To determine whether a given interpretation is correct, we check the geographic distance between the interpretation's lat/long values and the ground truth lat/long values, and if it lies within a small threshold (10 miles) we consider it correct. This method allows for the inevitable minor variations between the tested systems, due to their having selected interpretations from different gazetteers.

Table 3 shows the performance results. Of all three methods, the Adaptive method has the best overall precision, especially so for the *LGL* and *CLUST* datasets. Adaptive also maintains this high precision while having high toponym recall. This is best seen for the *LGL* dataset where Adaptive has a 17% advantage over OpenCalais, and a 22% advantage over Placemaker, along with a recall advantage of 32% over OpenCalais and 6% advantage over Placemaker. These performance numbers indicate our method's superior performance in terms of the toponym resolution task. Examining performance for all the methods across the three datasets, the methods performed best on *ACE*, worst on *LGL*, and in the middle for *CLUST*. These results follow our intuition that correctly geotagging documents containing smaller, less well-known locations (*LGL*) is more difficult than for larger, more well-known locations (*ACE*).

Table 4: Toponym resolution accuracy for different feature combinations.

	ACE			*LGL*			*CLUST*		
	P	R	F_1	P	R	F_1	P	R	F_1
I	0.91	0.41	0.57	0.96	0.26	0.40	0.93	0.30	0.45
I,P	0.97	0.59	0.74	0.96	0.47	0.63	0.98	0.38	0.55
I,P,A=B_1	0.99	0.84	0.91	0.96	0.61	0.75	0.98	0.71	0.82
B_1,*D*	0.99	0.90	0.94	0.96	0.62	0.75	0.98	0.72	0.83
B_1,*L*	0.98	0.86	0.92	0.95	0.90	0.93	0.97	0.77	0.86
B_1,*D,L*=B_2	0.99	0.90	0.94	0.95	0.90	0.93	0.97	0.76	0.86
$B_1, W_{80,\infty}$	0.98	0.88	0.93	0.94	0.65	0.77	0.96	0.71	0.82
$B_2, W_{80,\infty}$	0.99	0.88	0.93	0.94	0.88	0.91	0.96	0.73	0.83

Our next set of experiments tested various combinations of features used in the Adaptive method, to illustrate each feature's overall utility. We used different combinations of the features described in Section 2.3, as well as the adaptive context features described in Section 3. Table 4 contains the performance results, with feature abbreviations corresponding to those used in Section 2.3, and feature combinations indicated with commas (e.g., *I,P* combines `interps` and `population`). In addition, we considered two baseline feature combinations B_1 and B_2. B_1 tested only the domain-independent features (*I,P,A*), while B_2 also included those features tailored for the news domain (*D,L*). We again used our adaptive context feature ($W_{80,\infty}$) with window breadth of 80 tokens and unlimited window depth. In general, resolution precision was high for all feature combinations, so the main difference was resolution recall. For *ACE* and *CLUST*, the `dateline` and `locallex` features did not improve B_1 much, but `locallex` did make a large difference for *LGL*. Our adaptive context features in general improved B_1, for *LGL* in particular. However, in combination with B_2, the adaptive context features showed little improvement and in some cases lower performance, which is not overly surprising in that domain-specific features will exhibit domain-specific performance, and sometimes, adding features to a model will decrease performance. However, taken as a whole, the results illustrate our adaptive context features' utility for general geotagging purposes, especially over more simplistic features such as `population`.

We also conducted an experiment to measure the importance or utility of our features for classifying toponym interpretations. This process, also known as *feature selection* or *dimension reduction* [10], ranks the individual features in terms of their overall utility. For our feature importance measure, we used the *gain ratio* [10], a commonly-used, entropy-based measure for decision tree construction. We computed the gain ratio for each feature, and normalized the resulting importance values within each dataset. Results are presented in Figure 5. Interestingly, for each dataset, the `interps` and `altnames` features outranked `population`.

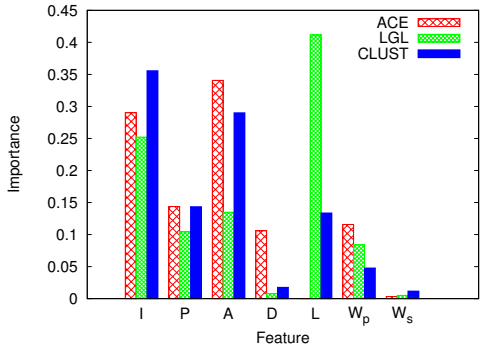
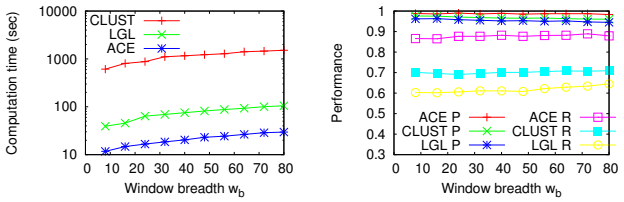


Figure 5: Importance of features used in the Adaptive method, as measured by the gain ratio.



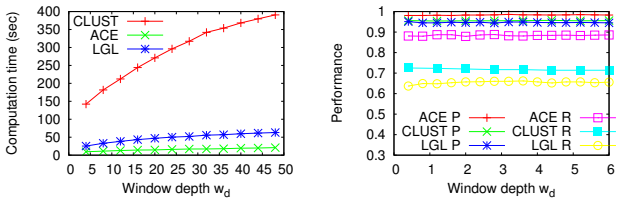
(a) (b)

Figure 6: Performance results when varying window breadth, showing changes in (a) time and (b) accuracy.

The `locallex` feature was highly important for *LGL*, though this is not too surprising considering the dataset's content of smaller, local news articles. The `windowprox` and `windowsib` have lower importance values, but interestingly, the `windowprox` feature has almost the same feature value as `population`. `windowsib`'s low importance value may be due to it being little-used in the three datasets.

4.5 Adaptive Parameters

In our final set of experiments, we tested how varying the adaptive parameters of our window features, namely the window breadth w_b and window depth w_d, would affect the speed and accuracy tradeoff for our methods. We used our adaptive context features in combination with our first baseline comparison method, B_1, described in the previous section, which is a combination of the `interps`, `population`, and `altnames` features. First, we varied the window breadth between 1–80 tokens and measured the resulting tradeoff. Figure 6a and Figure 6b show the results in terms of computation time and method accuracy, respectively. As the window breadth increases, the computation time increases linearly, which is to be expected. The computation time for *CLUST* is larger than for the other datasets due to its size. Interestingly, even with a small window breadth, precision remains high, and recall is respectable for all three datasets, giving evidence that the features are applicable even for domains where little time is allocated for geotagging. Also, while increasing the window breadth, recall also increases for the datasets, showing the time/accuracy tradeoff as expected. Results are similar for when varying window depth, shown in Figure 7a and Figure 7b.



(a) (b)

Figure 7: Performance results when varying window depth, showing changes in (a) time and (b) accuracy.

5. CONCLUSION

Our investigations of adaptive context features have shown their utility and flexibility for improving the geotagging of streaming news. In future work, we plan to test different weightings of toponyms in the window to judge their effect on resolution accuracy. For example, toponyms that are further away in the window could be given less weight, using linear or Gaussian weighting schemes, essentially leveraging Tobler's law [39] which states that "Everything is related to everything else, but near things are more related than distant things". In addition, we could consider clusters of news articles about the same topic, which are collected in the NewsStand system, and design other features using these clusters. For example, we might examine other documents in a cluster to get additional toponyms for consideration in geotagging the current document. This can be thought of as creating one large virtual document consisting of some or all of the documents in a cluster, and then extending the window to include toponyms in those other documents. As before, with large clusters, we may not want to consider all toponyms or all interpretations in other documents in the cluster, due to inhibitive performance penalties. In summary, adaptive context features serve as a flexible, useful addition to geotagging algorithms for streaming news and other textual domains.

6. REFERENCES

[1] R. Abascal-Mena and E. López-Ornelas. Geo information extraction and processing from travel narratives. In *ELPUB'10: Proceedings of the 14th International Conference on Electronic Publishing*, pages 363–373, Helsinki, Finland, June 2010.

[2] M. D. Adelfio, M. D. Lieberman, H. Samet, and K. A. Firozvi. Ontuition: Intuitive data exploration via ontology navigation. In *GIS'10: Proceedings of the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 540–541, San Jose, CA, Nov. 2010.

[3] E. Amitay, N. Har'El, R. Sivan, and A. Soffer. Web-a-Where: Geotagging web content. In *SIGIR'04: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 273–280, Sheffield, UK, July 2004.

[4] I. Anastácio, B. Martins, and P. Calado. Classifying documents according to locational relevance. In *EPIA'09: Proceedings of the 14th Portuguese Conference on Artificial Intelligence*, pages 598–609, Aveiro, Portugal, Oct. 2009.

[5] W. G. Aref and H. Samet. Efficient processing of window queries in the pyramid data structure. In *PODS'90: Proceedings of the 9th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 265–272, Nashville, TN, Apr. 1990.

[6] P. N. Bennett, F. Radlinski, R. W. White, and E. Yilmaz. Inferring and using location metadata to personalize web search. In *SIGIR'11: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 135–144, Beijing, China, July 2011.

[7] L. Breiman. Random forests. *Machine Learning*, 45(1): 5–32, Oct. 2001.

[8] D. Buscaldi and B. Magnini. Grounding toponyms in an Italian local news corpus. In *GIR'10: Proceedings of the 6th Workshop on Geographic Information Retrieval*, Zurich, Switzerland, Feb. 2010.

[9] R. O. de Alencar, C. A. Davis, Jr., and M. A. Gonçalves. Geographical classification of documents using evidence from Wikipedia. In *GIR'10: Proceedings of the 6th Workshop on Geographic Information Retrieval*, Zurich, Switzerland, Feb. 2010.

[10] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, New York, second edition, 2001.

[11] E. Garbin and I. Mani. Disambiguating toponyms in news. In *HLT/EMNLP'05: Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 363–370, Vancouver, Canada, Oct. 2005.

[12] Y.-H. Hu and L. Ge. A supervised machine learning approach to toponym disambiguation. In A. Scharl and K. Tochtermann, editors, *The Geospatial Web*, pages 117–128. Springer, London, 2007.

[13] U. Irmak and R. Kraft. A scalable machine-learning approach for semi-structured named entity recognition. In *WWW'10: Proceedings of the 19th International World Wide Web Conference*, pages 461–470, Raleigh, NC, Apr. 2010.

[14] W. Kienreich, M. Granitzer, and M. Lux. Geospatial anchoring of encyclopedia articles. In *IV'06: Proceedings of the 10th International Conference on Information Visualization*, pages 211–215, London, July 2006.

[15] J. L. Leidner. *Toponym Resolution in Text: Annotation, Evaluation and Applications of Spatial Grounding of Place Names*. PhD thesis, University of Edinburgh, Edinburgh, Scotland, 2007.

[16] H. Li, R. K. Srihari, C. Niu, and W. Li. InfoXtract location normalization: a hybrid approach to geographic references in information extraction. In *Proceedings of the HLT-NAACL 2003 Workshop on Analysis of Geographic References*, pages 39–44, Edmonton, Canada, May 2003.

[17] M. D. Lieberman and J. Lin. You are where you edit: Locating Wikipedia users through edit histories. In *ICWSM'09: Proceedings of the 3rd International AAAI Conference on Weblogs and Social Media*, pages 106–113, San Jose, CA, May 2009.

[18] M. D. Lieberman and H. Samet. Multifaceted toponym recognition for streaming news. In *SIGIR'11: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 843–852, Beijing, China, July 2011.

[19] M. D. Lieberman, H. Samet, J. Sankaranarayanan, and J. Sperling. STEWARD: Architecture of a spatio-textual search engine. In *GIS'07: Proceedings of the 15th ACM International Symposium on Geographic Information Systems*, pages 186–193, Seattle, WA, Nov. 2007.

[20] M. D. Lieberman, H. Samet, and J. Sankaranarayanan. Geotagging: Using proximity, sibling, and prominence clues to understand comma groups. In *GIR'10: Proceedings of the 6th Workshop on Geographic Information Retrieval*, Zurich, Switzerland, Feb. 2010.

[21] M. D. Lieberman, H. Samet, and J. Sankaranarayanan. Geotagging with local lexicons to build indexes for textually-specified spatial data. In *ICDE'10: Proceedings of the 26th International Conference on Data Engineering*, pages 201–212, Long Beach, CA, Mar. 2010.

[22] I. Mani, J. Hitzeman, J. Richer, and D. Harris. *ACE 2005 English SpatialML Annotations*. Linguistic Data Consortium, Philadelphia, PA, Jan. 2008. LDC Catalog Number LDC2008T03.

[23] B. Martins, H. Manguinhas, and J. Borbinha. Extracting and exploring the geo-temporal semantics of textual resources. In *ICSC'08: Proceedings of the 2nd IEEE International Conference on Semantic Computing*, pages 1–9, Santa Clara, CA, Aug. 2008.

[24] B. Martins, I. Anastácio, and P. Calado. A machine learning approach for resolving place references in text. In *AGILE'10: Proceedings of the 13th AGILE International Conference on Geographic Information Science*, pages 221–236, Guimarães, Portugal, May 2010.

[25] NTS Realty Holdings. NTS Realty Holdings limited partnership announces refinancing of debt on eight multifamily properties. URL `http://www.earthtimes.org/articles/show/nts-realty-holdings-limited-partnership,1062375.shtml`. Accessed 16 Jan 2010.

[26] R. C. Pasley, P. D. Clough, and M. Sanderson. Geo-tagging for imprecise regions of different sizes. pages 77–82.

[27] R. S. Purves, P. Clough, C. B. Jones, A. Arampatzis, B. Bucher, D. Finch, G. Fu, H. Joho, A. K. Syed, S. Vaid, and B. Yang. The design and implementation of SPIRIT: a spatially aware search engine for information retrieval on the Internet. *IJGIS: International Journal of Geographical Information Science*, 21(7):717–745, Aug. 2007.

[28] T. Qin, R. Xiao, L. Fang, X. Xie, and L. Zhang. An efficient location extraction algorithm by leveraging web contextual information. In *GIS'10: Proceedings of the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 53–60, San Jose, CA, Nov. 2010.

[29] G. Quercini, H. Samet, J. Sankaranarayanan, and M. D. Lieberman. Determining the spatial reader scopes of news sources using local lexicons. In *GIS'10: Proceedings of the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 43–52, San Jose, CA, Nov. 2010.

[30] E. Rauch, M. Bukatin, and K. Baker. A confidence-based framework for disambiguating geographic terms. In *Proceedings of the HLT-NAACL 2003 Workshop on Analysis of Geographic References*, pages 50–54, Edmonton, Canada, May 2003.

[31] H. Samet, H. Alborzi, F. Brabec, C. Esperança, G. R. Hjaltason, F. Morgan, and E. Tanin. Use of the SAND spatial browser for digital government applications. *CACM: Communications of the ACM*, 46(1):61–64, Jan. 2003.

[32] H. Samet, B. E. Teitler, M. D. Adelfio, and M. D. Lieberman. Adapting a map query interface for a gesturing touch screen interface. In *WWW'11: Proceedings of the 20th International World Wide Web Conference*, pages 257–260, Hyderabad, India, Mar. 2011.

[33] J. Sankaranarayanan, H. Samet, B. Teitler, M. D. Lieberman, and J. Sperling. TwitterStand: News in tweets. In *GIS'09: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 42–51, Seattle, WA, Nov. 2009.

[34] C. A. Shaffer, H. Samet, and R. C. Nelson. QUILT: a geographic information system based on quadtrees. *IJGIS: International Journal of Geographical Information Science*, 4(2):103–131, Apr. 1990.

[35] D. A. Smith and G. Crane. Disambiguating geographic names in a historical digital library. In *ECDL'01: Proceedings of the 5th European Conference on Research and Advanced Technology for Digital Libraries*, pages 127–136, Darmstadt, Germany, Sept. 2001.

[36] J. Strötgen, M. Gertz, and P. Popov. Extraction and exploration of spatio-temporal information in documents. In *GIR'10: Proceedings of the 6th Workshop on Geographic Information Retrieval*, Zurich, Switzerland, Feb. 2010.

[37] B. E. Teitler, M. D. Lieberman, D. Panozzo, J. Sankaranarayanan, H. Samet, and J. Sperling. NewsStand: A new view on news. In *GIS'08: Proceedings of the 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 144–153, Irvine, CA, Nov. 2008.

[38] R. Tobin, C. Grover, K. Byrne, J. Reid, and J. Walsh. Evaluation of georeferencing. In *GIR'10: Proceedings of the 6th Workshop on Geographic Information Retrieval*, Zurich, Switzerland, Feb. 2010.

[39] W. Tobler. A computer movie simulating urban growth in the detroit region. *Economic Geography*, 46(2):234–240, 1970.

[40] A. Weichselbraun. A utility centered approach for evaluating and optimizing geo-tagging. In *KDIR'09: Proceedings of the 1st International Conference on Knowledge Discovery and Information Retrieval*, Madeira, Portugal, Oct. 2009.

[41] B. P. Wing and J. Baldridge. Simple supervised document geolocation with geodesic grids. In *ACL'11: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 955–964, Portland, OR, June 2011.

[42] A. G. Woodruff and C. Plaunt. GIPSY: Automated geographic indexing of text documents. *JASIS: Journal of the American Society for Information Science*, 45(9): 645–655, Oct. 1994.