Near-Duplicate Detection by Instance-level Constrained Clustering

Hui Yang
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
huiyang@cs.cmu.edu

Jamie Callan
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
callan@cs.cmu.edu

ABSTRACT

For the task of near-duplicated document detection, both traditional fingerprinting techniques used in database community and bag-of-word comparison approaches used in information retrieval community are not sufficiently accurate. This is due to the fact that the characteristics of near-duplicated documents are different from that of both "almost-identical" documents in the data cleaning task and "relevant" documents in the search task. This paper presents an instance-level constrained clustering approach for near-duplicate detection. The framework incorporates information such as document attributes and content structure into the clustering process to form near-duplicate clusters. Gathered from several collections of public comments sent to U.S. government agencies on proposed new regulations, the experimental results demonstrate that our approach outperforms other near-duplicate detection algorithms and as about as effective as human assessors.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**] Clustering

General Terms

Algorithms, Performance, Experimentation.

Keywords

Duplicate detection, clustering, public comments

1. INTRODUCTION

Near-duplicate detection is the task to identify and organize documents that are "nearly identical" to each other. In another word, near-duplicates originated from the same reference copy. Early research on duplicate detection was done mainly for "almost-identical" documents and mostly in the areas of databases, digital libraries, electronic publishing and web search tasks [1][3][4][5][7][9][10][1][12].

Many algorithms consider near-duplicate detection as a "retrieval" problem, i.e., to find the near duplicates for a single document. However, just looking for near-duplicates for a document neither fulfills the needs of organizing a large collection of documents nor that of identifying the original reference copy for near-duplicates. Moreover, one important characteristic of near-duplicate detection is that two documents may be considered near-duplicates even if they only share a relatively small amount of text. Solely relying on bag-of-word similarity measures, even some sophisticated ones carefully calibrated to the near-duplicate detection problem, are brittle under such situation. Additional knowledge about the documents is sometimes the key to tackle the problem.

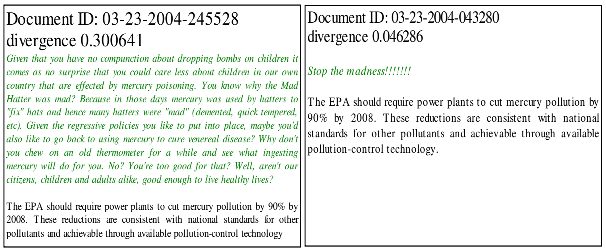
A more flexible framework is hence desirable to make use of the additional information available in the document collection to organize the documents into near-duplicate clusters with the original reference copy recognized. In this work, near-duplicate detection is considered as an instance-level constrained clustering problem. Instance-level constrained clustering is a semi-supervised process which provides a flexible framework to incorporate constraints on document attributes, content structure, and self-defined preferences to guide through the clustering process. We believe that our work is the first such semi-supervised clustering approach developed for the general near-duplicate detection task.

A key component of this semi-supervised clustering algorithm is the use of instance-level constraints that are based on document attributes and editing styles of near-duplicates. Three types of instance-level clustering constraints[1] are used in our system: *must-link* (believed to be in the same class), *cannot-link* (believed to be in different classes) and *family-link* (possibly in the same class) constraints. Note that the instance-level constraints used in semi-supervised clustering are not the same as labeled data used in classification or partial-labeled data in semi-supervised classification, they are pair-wise constraints which are not sufficient to general class labels. Though they are not as strong conditions as class labels, they provide a valuable guidance for the conventional unsupervised clustering process. Moreover, instance-level pair-wise constraints are much easier to generate than class labels and the approach is still largely unsupervised.

A newly-emerging domain for near-duplicate detection is notice and comment rulemaking [13], in which U.S. regulatory agencies receive comments about proposed regulations from the general public. Many of the comments are "form letters" (*exact duplicates*) and modified copies of form letters (*near duplicates*). U.S. law requires agencies to respond to every substantive issue raised by the public, even if the issue is inserted into a form letter. Spotting exact-duplicates is relatively easy, but identifying near-

SIGIR'06, August 6–11, 2006, Seattle, Washington, USA.

[1] *Must-link* and *cannot-link* concepts were first introduced in [14].



Document ID: 03-23-2004-245528
divergence 0.300641

Given that you have no compunction about dropping bombs on children it comes as no surprise that you could care less about children in our own country that are effected by mercury poisoning. You know why the Mad Hatter was mad? Because in those days mercury was used by hatters to "fix" hats and hence many hatters were "mad" (demented, quick tempered, etc). Given the regressive policies you like to put into place, maybe you'd also like to go back to using mercury to cure venereal disease? Why don't you chew on an old thermometer for a while and see what ingesting mercury will do for you. No? You're too good for that? Well, aren't our citizens, children and adults alike, good enough to live healthy lives?

The EPA should require power plants to cut mercury pollution by 90% by 2008. These reductions are consistent with national standards for other pollutants and achievable through available pollution-control technology

Document ID: 03-23-2004-043280
divergence 0.046286

Stop the madness!!!!!!!

The EPA should require power plants to cut mercury pollution by 90% by 2008. These reductions are consistent with national standards for other pollutants and achievable through available pollution-control technology.

Figure 1: Near-duplicate Example

duplicates and their unique component(s) is a more challenging task that is currently done manually. When a proposed regulation attracts a large amount of public interests, an agency may receive hundreds of thousands of comments that must be processed within a few weeks or months, typically requiring significant manual effort and expense. Near-duplicate detection automates the process and largely saves agencies' efforts.

Our system, DURIAN (**DU**plicate **R**emoval **I**n l**A**rge collectio**N**s), is evaluated in experiments with public comments recently collected for U.S. Environmental Protection Agency (EPA) and Department of Transportation (DOT) rulemakings. The experimental results show that system-to-human intercoder agreement is comparable to human-to-human intercoder agreement. They also demonstrate that incorporating instance-level constraints boost the accuracy and efficiency for clustering tasks. It shows that our approach is a more promising framework than unsupervised clustering without any constraint.

Furthermore, although evaluated in the public comment domain, the technique for near-duplicate detection is domain-independent; it could also be applied to near-duplicate detection in other domains such as question answering, web search, information flow, and plagiarism detection.

The rest of the paper is organized as follows. Section 2 defines the problem. Section 3 reviews related research. Section 4 describes the algorithm. Section 5 discusses evaluation methodology. Section 6 presents experimental results. Section 7 concludes.

2. PROBLEM DEFINITION

Near-duplicate detection is different from other Information Retrieval (IR) tasks in how it defines what it means for two documents to be "similar". In many IR tasks document similarity refers to semantic "relevance" among documents, which are could be syntactically very different but still relevant. In contrast, the definition of similarity in duplicate detection in early database research [1][3][1] is very conservative, which is mainly to find syntactically "almost-identical" documents. As pointed out by Metzler et al. [10], for other tasks that need to detect documents with "intermediate level of similarity", there has not been much research done. Near-duplicate detection in notice and public comment rulemaking is one such task and our work is designed to be general enough to apply to other detection tasks for documents with "intermediate level of similarity".

In notice and public comments rulemaking domain there are two broad categories of documents: comments that are written more-or-less from scratch (they contain unique insights and opinions, derivative opinions, spam or viruses), and comments that are written based on a form letter. The former will produce singleton clusters and the later will form near-duplicate clusters. We define two documents to be near-duplicates if they are from the same origin. In particular, several subcategories of near-duplicates are defined based on common editing styles:

- *Block Added:* Add one or more paragraphs (<200 words) to a document;
- *Block Deleted*: Remove one or more paragraphs (<200 words) from a document;
- *Key Block*: Contains at least one paragraph from a document;
- *Minor Change*: A few words altered within a paragraph (<5% or 15 word change in a paragraph) ;
- *Minor Change & Block Edit*: A combination of minor change and block edit;
- *Block Reordering*: Reorder the same set of paragraphs;
- *Repeated*: Repeat the entire document several times in another document;
- *Bag-of-word similar*: >80% word overlap (not in above categories); and
- *Exact*: 100% word overlap.

As we mentioned earlier, in the task of near-duplicate detection, even if two documents share a small amount of text, they may still be considered as near duplicates. Solely relying on similarity measures, even some sophisticated ones carefully fitted to the near-duplicate detection problem, for instance, relevant frequency [7], statistical translation-model-based measure [10], and Kullback-Leibler (KL) divergence [16], becomes fragile. The example given in Figure 1 shows that two documents clearly from the same origin are given very different similarity scores (0.30 and 0.05 respectively) by one of state-of-the-art similarity measure, Dirichlet-smoothed KL divergence, which is used in our baseline unsupervised algorithm as the distance metric (See Section 4.2 for details). It is undesirable that documents which should be grouped together have dissimilar scores. It is also hard to determine what the correct threshold is for a particular cluster if the similarity measure is not able to represent the true grouping. On the other hand, sometimes the similarity score suggests that they should be grouped together however they should not be, for instance, some other documents which have very close scores but are actually submitted for different proposed rules, i.e., different topics. Therefore, just employing a single distance metric in unsupervised algorithm prevents us revealing the near-duplicates in both situations.

This paper studies how to employ additional knowledge to tackle the above problem and boost the near-duplicate detection accuracy. In particular, the tasks include:

- To identify where a particular near-duplicate first originated; and
- To achieve highly effective near-duplicate detection by incorporating additional knowledge via instance-level constrained clustering.

3. RELATED WORK

3.1 Duplicate Detection

The problem of finding near-duplicate documents has been a subject of research in the database and web-search communities for some years. The applications range from plagiarism detection in web publishing to redundancy detection in large datasets. The common duplicate detection techniques are classified into two categories: Fingerprint-based and Fulltext-based.

3.1.1 Duplicate Detection Using Fingerprints

A fingerprint of a document is a set of integers, each of which is the hash value for a substring extracted from the document. In this paper, to be clearer in concept, the term "fingerprint" refers only to document-level fingerprint while the term "integer" or "hash-value" refers to hash function output, which in many other paper is sometimes also called "fingerprint". Each integer is stored in an index for fast access during query process. Similarity between two documents is measured by counting the number of common integers. Algorithms are different in their choices of hash functions, substring size, substring number, and substring selection strategy.

Hashing functions is used to generate hash values for substrings. Popular hash functions include NIST's SHA1 [11] and Rabin [3]. However, many other hash functions are qualified for this task as long as they are reproducible and with a low rate of hash collision.

Substring size is defined by the length of each substrings extracted from a document. Larger size increases the chance of false negatives in duplicate detection while smaller size increases that of false positives, e.g., SCAM [1] used a very small substring, word, as the unit for fingerprinting. Prior research suggested that substrings of 3-5 words are good [7].

Substring number is the number of substrings extracted from a document to build a fingerprint. Some techniques used a fixed number of substrings for efficiency, e.g., I-Match presented in [4], while many others used a variable number of substrings for a more accurate representation of the document, e.g., DSC presented in [3]. A smaller number of substrings have the risk of ignoring short documents and increasing false negatives.

Substring selection strategy is the way to pick which substrings to hash. It can be categorized as *position-based, hash-value-based, anchor-based* and *frequency-based* strategies*. Position-based* strategy selects substrings based on their offsets in a document, a sentence or a paragraph. It includes full fingerprinting [1][3][7], non-overlapping fingerprinting [1], and overlapping fingerprinting [3]. It is popular due to the simplicity. *Hash-value-based* strategy is also popular. The famous shingling approach (or DSC), proposed by Broder, et al. [3] picks substrings whose hash values are multiples of an integer. *Anchor-based* strategy extracts substrings that start with special words or character sequences. It was shown to be one of the best substring selection methods [7], however this approach has to be manually tuned to fit a specific collection and hence is not that practical. *Frequency-based* strategy selects substrings based on their frequency of occurrences in the document, the entire collection [4][5] and/or external collections [9]. Term frequency within a document (tf) and inverse document frequency in a collection (idf) are used to select the substrings. Conrad et al. [5] used 30-60 highest idf words. Chowdhury, et al.'s I-Match [4] also selected terms with high idf. However term selection based on idf alone can be overly sensitive to small changes in document content and hence the false negative is high. The more recent extended I-Match [9] used external collection statistics to select the lexicon. It achieved a better recall by introducing multiple fingerprints. However, it is computationally expensive.

3.1.2 Duplicate Detection Using Full-Text

The simplest full-text approach is to adapt methods originally developed for search engines, for example, vector-space model, which treats a document as bag-of-words, with term weights determined by tf.idf values, and similarity determined by cosine similarity. Traditional cosine-similarity measure focuses on finding a semantic relevant document while near-duplicate detection focuses more on syntactic similarity. Several previous works thus have been done in finding suitable similarity measures to address syntactic similarity among documents. The identity measure proposed by [7] emphasizes that the gap between rare words' term frequency in two documents should be smaller than that between common words' and their best ranking is giving by a term weighting function biased towards rare terms. Metzler et al. [10] used statistical translation models to estimate the probability that one sentence in a document is a translation of another sentence in another document. The probability of aligning to a absent term is estimated by the background language model. The translation probability serves as the basis of the sentence-level and the document-level similarity scores[10].

3.2 Semi-supervised Clustering

Clustering is a very useful tool for data analysis, especially in the initial process of a large dataset analysis where the data labeling is impractical or unavailable. However, the usefulness of clustering is questioned by researchers because of its "blindness" rooted in its nature of "no supervision". Moreover, it is always hard to evaluate the performance of a clustering algorithm since there could be many ways of defining what a "good" set of clusters should be and hence there is no definite gold standard to compare with. The semi-supervised approach to clustering has a short history [8][14][15]. In addition to the distance metric, semi-supervised approaches make use of a small amount of additional knowledge in documents to guide the clustering process. They are valuable guidance which could greatly improve the performance of clustering process. The proposed algorithms differ in the how to supervise the clustering process. Some focus on modifying distance metrics, for example, the Jensen-Shannon divergence trained with gradient descent, or the Euclidean distance modified by a shortest-path algorithm [8]. Some others focus on modifying the clustering process on the fly , for example, using constraints to initialize clusters, or requiring constraints to be satisfied during cluster assignments [14].

4. ALGORITHM

4.1 Document Representation

Our focus is on public comments that are submitted through email and Web forms; these represent the majority of comments submitted for high-profile regulations in recent years. In order to get useful additional knowledge about the documents automatically, DURIAN employs an information extraction (IE) module. Simple, rule-based heuristics enable it to identify document attributes (metadata), such as the comment sender/author, the receiver, timestamp, address block, signature block, salutation, docket identification number, topic, relayer (email-relaying organizations) and footer block (email signatures attached by email service providers). The documents are analyzed and represented in XML format with extracted metadata as well as the main text. This preprocessing not only normalizes the representation by separating the header, salutation, footer and signature lines from the main text, it also provides ready-to-use additional knowledge about the documents to derive the instance-level constraints automatically.

4.2 Distance Metric

For any clustering algorithm it is important to define a distance metric. Insertion and deletion of a block of text to/from a form letter are common in public comments, hence it is not surprising that a form letter and its near-duplicates have unmatched vocabularies. In [10], the authors used a statistical translation model to get document similarity and claimed that their measure for the detection of documents with "intermediate level of similarity" is the best among the current technologies. The statistical translation model turns out to be equivalent to the Dirichlet-smoothed KL-divergence (See [10] for more details). In our work, Dirichlet-smoothed KL-divergence is employed as the distance metric in the baseline unsupervised clustering algorithm.

When comparing the similarity of two documents, the problem of unmatched vocabulary, which is very common in near-duplicate detection, needs to be handled. Instead of assigning zero weights to an absent word, smoothing techniques give the word a probability proportional to its overall probability in a background language model. For any two documents d_a and d_b with word probability distributions p_a and p_b respectively, the KL divergence between them is:

$$\begin{aligned} KL(p_a \,\|\, p_b) &= \sum_w p_a(w)\log\frac{p_a(w)}{p_b(w)} \\ &= \sum_w p_a(w)\log p_a(w) - \sum_w p_a(w)\log p_b(w) \end{aligned} \qquad (1)$$

Here $p_a(w)$ is the probability of word w occurring in document d_a, similar for $p_b(w)$. The first term in Equation 1 can be dropped since it does not depend on distribution p_b and hence is irrelevant for ranking p_b. The KL divergence becomes:

$$KL(p_a \,\|\, p_b) \propto -\sum_w p_a(w)\log p_b(w) \qquad (2)$$

where $$p_b(w) = \begin{cases} p_s(w \mid d_b) & \text{if w is seen} \\ \alpha_d\, p(w \mid C) & \text{otherwise} \end{cases} \qquad (3)$$

α_d is a coefficient provided for each unseen word. Note that all of the probabilities should sum to one, $p_s(w \mid d_b)$ is the smoothed probability of a word present in the document, and $p(w \mid C)$ is the background language model. Due to space limitation, the derivation is skipped a bit however it can be shown that with such a smoothing method, the KL divergence becomes:

$$-\sum_{w \in d_a} p_a(w)\log\frac{p_s(w \mid d_b)}{\alpha_d\, p(w \mid C)} + \log \alpha_d \qquad (4)$$

Note that the scoring is now based on a sum over all the terms that occur in both documents, i.e., all "matched" terms.

By Dirichlet prior smoothing [1], we have:

$$p_s(w \mid d_b) = \frac{tf(w,d_b) + \mu p(w \mid C)}{\mu + |d_b|}, \qquad (5)$$

$$\alpha_d = \frac{\mu}{\mu + |d_b|}, \qquad (6)$$

Where $tf(w,d_b)$ is the term frequency of word w in document d_b, $|d_b|$ is the length of document d_b, μ is a parameter in Dirichlet smoothing and is set to 1 in this work. The background language model $p(w \mid C)$ is estimated by maximum likelihood estimation (MLE):

$$p(w \mid C) = \frac{\sum_{d_k \in C} tf(w,d_k)}{\sum_{d_j \in C} \sum_{w_i \in d_j} tf(w_i,d_j)} \qquad (7)$$

For $p_a(w)$, there are many ways to estimate it. If MLE is chosen and document d_a is used as evidence, $p_a(w)$ can be estimated as:

$$p_a(w) = p(w \mid d_a) = \frac{tf(w,d_a)}{\sum_{w_i \in d_a} tf(w_i,d_a)} \qquad (8)$$

By putting Equations (5)(6)(7)(8) into (4), we can see that the KL-divergence scoring formula is essentially the same as the query likelihood retrieval formula in [1], and also equivalent to the statistical translation model presented in [10]. Since KL-divergence is non-negative and non-symmetric, we define and use the minimum value of two KL-divergences as the distance metric between documents d_a and d_b:

$$dist(d_a,d_b) = \min(KL(p_a \,\|\, p_b), KL(p_b \,\|\, p_a)) \qquad (9)$$

The smoothed KL divergence is thus the distance metric used in the unsupervised clustering algorithm, which is the baseline in our experiments in section 6.2.

4.3 Redundancy-based Reference Copy Detection

One of the tasks of near-duplicate detection is to find the reference copy from which a set of documents derived. DURIAN employs a simple but effective redundancy-based reference copy detection strategy. The assumption is that form letters are submitted in both unmodified and modified forms. Any document that has many exact duplicates is considered an instance of a form letter.[2] In practice, when comment volume is high, a significant fraction of the comments is exact duplicates of form letters, making them relatively easy to find. This fact makes our approach practical.

To identity exact duplicates, all words in a document are converted into a single *document string*, which is a long string of characters with all the words in the document concatenated together (white spaces and punctuations removed). After that, a hash function is applied to the document string to create a unique identifier for this particular document. The hash function used in DURIAN is the NIST's security hash function, SHA1 [11]. It makes sure that the chance of hash value collision is as low as $p(2^{160})$. It is also designed to be very fast and is good for strings of any length. After hashing, a <hash-value, document id> tuple is created for each document and all the tuples are sorted by their hash values. Documents resulting in the same hash value are neighbors since they are sorted. A simple linear scan to the sorted list is performed to identify documents with same hash values,

[2] Documents with just a few exact duplicates are usually comments that a person accidentally submitted several times.

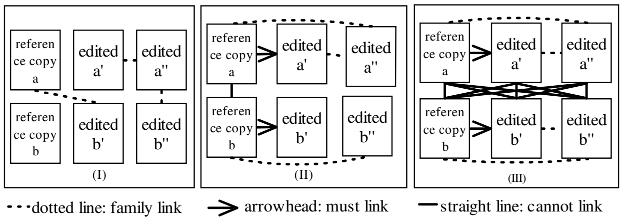


Figure 2: Constraint Transitive Closure Example

which are considered exact duplicates. Given a set of exact duplicates whose size is bigger than m (m=5 in our system), the document with the earliest timestamp is selected to be the reference copy. DURIAN also annotates the reference copy with the number of exact duplicates in the set, and retains the information for further study.

4.4 Incorporating Instance-level Constraints

Instance-level constraints are generated and incorporated in the clustering process for near-duplicate detection. In many cases, public comments share the same metadata, such as email relayer, approximate file size, approximate date, and docket identification number, are likely to be near-duplicates. These attributes can be used to create instance-level constraints that indicate that certain pairs of documents must be, cannot be, or are likely to be in the same duplicate cluster. In this work, two hard constraints, *must-link* and *cannot-link*, are used, and a new type of instance-level constraint, *family-link*, is introduced. Each type of constraints is described in more detail below.

4.4.1 Must-link

Must-links are pair-wise constraints specifying that two documents must occur in the same cluster. They are strong connections of two documents. The must-link conditions in near-duplicate detection include the complete containment of the reference copy (*key block*), and word overlap > 95% (*minor change*). By using the must-links, pairs of documents satisfying the conditions are forced to be in the same cluster even if their KL divergence is high, i.e., even if they are dissimilar in bag-of-word comparison.

4.4.2 Cannot-link

Cannot-links are pair-wise constraints specifying that two documents must go to different clusters, i.e., cannot be in the same cluster. They are strong exclusions between two documents. The cannot-link condition in near-duplicate detection only occurs when two documents cite different docket identification numbers in their texts; this happens more frequently than one might expect because people often use the wrong email address or Web form when submitting comments.

4.4.3 Family-link

Family-link constraints are pair-wise constraints specifying that two documents are likely to be in the same cluster. In near-duplicate detection a family-link is created when two documents have the same email relayer, the same docket identification number, similar file sizes, and the same footer block.

```
Clustering (ReferenceCopies $R$, Non-referenceCopies $NR$) {
 A) Initialize the Duplicate Cluster Collection $N$: $N \leftarrow \varnothing$
    and family-link adjusting parameter $\alpha$=0.05.
 B) Pick one document $d_i$ from $R$, (See Section 4.3.)
 C) Retrieve candidate set $S_i$ for $d_i$. $\forall$ document $s_{ij} \in S_i$,
    a) if $(s_{ij},d_i) \in Must$,          /* if must-link constraint holds*/
       $n_{d_i} \leftarrow n_{d_i} \cup \{s_{ij}\}$, goto e); /*add $s_{ij}$ to duplicate cluster $n_{d_i}$*/
    b) if $(s_{ij},d_i) \in Family$,     /* if family-link constraint holds*/
       $dist(s_{ij},d_i) = dist(s_{ij},d_i) - \alpha$ ; /*adjusting distance metric*/
    c) $\forall$ cluster centroid $d_k$ in $N$,  /* check must- and family-
                                                  links for other clusters*/
       if $(s_{ij},d_k) \in Must$, $n_{d_k} \leftarrow n_{d_k} \cup \{s_{ij}\}$, goto e) ;
       if $(s_{ij},d_k) \in Family$, $dist(s_{ij},d_k) = dist(s_{ij},d_k) - \alpha$ ;
    d) if $dist(s_{ij},d_i) < \theta_i$
       if $d_i \notin N$, $\forall$ cluster centroid $d_k$ in $N$,
         if $dist(s_{ij},d_i) <= \min_k(dist(s_{ij},d_k))$ & $((s_{ij},d_k) \notin Must)$
             $N \leftarrow N \cup \{n_{d_i}\}$,            /*create a new cluster $n_{d_i}$*/
             add $s_{ij}$ into $n_{d_i}$: $n_{d_i} \leftarrow n_{d_i} \cup \{s_{ij}\}$ ,
             eliminate $s_{ij}$ from $n_{d_k}$: $n_{d_k} \leftarrow n_{d_k} - \{s_{ij}\}$;
      if $d_i \in N$ , $\forall$ cluster centroid $d_k$ in $N$,
         if $(dist(s_{ij},d_i) > dist(s_{ij},d_k))$ & $((s_{ij},d_k) \notin Cannot)$
           $n_{d_k} \leftarrow n_{d_k} \cup \{s_{ij}\}$;       /*add if no cannot-link exists*/
    e) process next document in $S_i$;
 E) If $R \neq \varnothing$ , go to step B);
 F) If $R = \varnothing$ &$NR \neq \varnothing$ ,  let $R \leftarrow NR$, $NR \leftarrow \varnothing$ , go to step B);
 G) If $R = \varnothing$ &$NR = \varnothing$ , output $N$ as the final set of clusters.
}
```

Figure 3: Clustering Algorithm

4.4.4 Constraint Transitive Closure

As mentioned above, an initial set of must-links, cannot-links and family-links are created between pairs of documents, based upon document attributes, content structure and preferences, A more complete set of constraints is obtained by taking the transitive closures of these three types of instance-level constraints. For any random documents d_a , d_b and d_c the transitive closures of the three types of constraints are shown below.

Must-link transitive closure:

$d_a =_{\mathrm{m}} d_b$, $d_b =_{\mathrm{m}} d_c \Rightarrow d_a =_{\mathrm{m}} d_c$

Cannot-link transitive closure:

$d_a =_{\mathrm{c}} d_b$, $d_b =_{\mathrm{m}} d_c \Rightarrow d_a =_{\mathrm{c}} d_c$

Family-link transitive closure:

$d_a =_{\mathrm{f}} d_b$, $d_b =_{\mathrm{m}} d_c \Rightarrow d_a =_{\mathrm{f}} d_c$; $d_a =_{\mathrm{f}} d_b$, $d_b =_{\mathrm{c}} d_c \Rightarrow d_a =_{\mathrm{c}} d_c$;

$d_a =_{\mathrm{f}} d_b$, $d_b =_{\mathrm{f}} d_c \Rightarrow d_a =_{\mathrm{f}} d_c$.

$=_{\mathrm{m}}$, $=_{\mathrm{c}}$ and $=_{\mathrm{f}}$ indicate must-link, cannot-link and family-link respectively.

Transitive closure provides a larger and more complete set of instance-level constraints. For example, initially there are only three family-links among six documents in Figure 2 (I). Assume three of them (a, a', a'') are derived from the reference copy a, the other three (b, b', b'') from b. By introducing one cannot-link between two reference copies, and two must-links between the reference copies and two edited copies in Figure 2 (II), many other constraints can be derived(Figure 2 (III)). Note that once a cannot-link is assigned, some previous family-links are removed and changed to cannot-link since cannot-link is a stronger condition.

4.4.5 The Clustering Algorithm

As mentioned earlier, any comment that has more than 5 exact

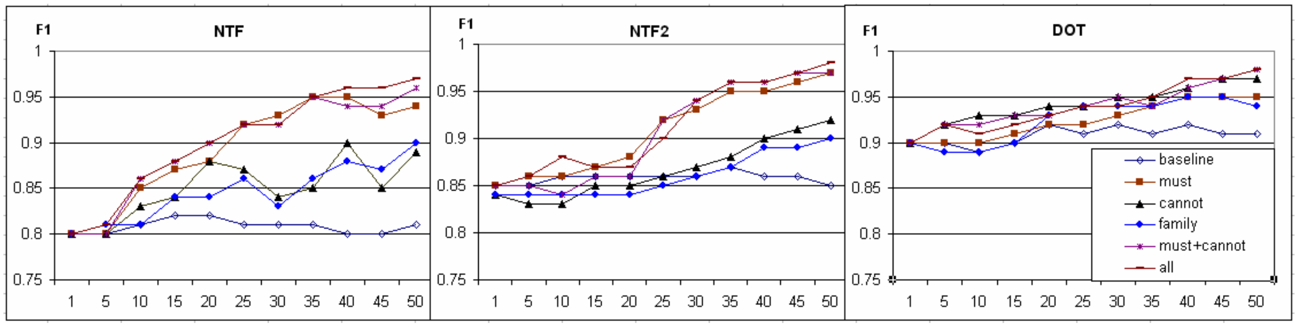


Figure 4: Number of Constraints vs. F1. (All graphs share the same legends)

Table 1: Near-duplicate Detection Intercoder Agreement

	Macro Averaged AC1			Micro Averaged AC1		
	NTF	NTF2	DOT	NTF	NTF2	DOT
Coder A / Coder B	0.93	0.90	0.95	0.99	0.95	0.96
Coder A / Program	0.92	0.80	0.86	0.93	0.90	0.88
Coder B / Program	0.90	0.82	0.94	0.91	0.91	0.98

duplicates (after lexical preprocessing) is considered an instance of a form letter. The copy with the earliest timestamp is the *reference copy* of the form letter. All reference copies are put into a set called *ReferenceCopies*. Some documents that are not reference copies are put into another set called *Non-referenceCopies*, which may be also become seeds later in the clustering process. The two sets of documents are inputs to the clustering algorithm detailed in Figure 3.

5. EVALUATION METHEDOLOGY

5.1 Data Sets

The research was conducted with public comments about the U.S. Environmental Protection Agency's (EPA) proposed *National Emission Standards For Hazardous Air Pollutants For Utility Air Toxics* rule (Docket USEPA-OAR-2002-0056), and the U.S. Department of Transportation's (DOT) proposed *Automobile Fuel Economy Standards* (Docket DOT-2003-16128). The EPA dataset contains 536,975 email messages. The DOT dataset contains 39,593 email messages and scanned letters in pdf format. Although the algorithms easily handle datasets of these sizes, manually obtaining assessments or class labels for the complete datasets would be impractical, thus we created three samples of 1,000 e-mails (two from EPA, one from DOT). These datasets were called NTF (*Name That Form*), NTF2 and DOT by the human assessors ("coders") at the University of Pittsburgh's Qualitative Data Analysis Program, a coding and assessment service operated by the University Center for Social and Urban Research. DURIAN's redundancy-based reference copy detection identified 28 reference copies of form letters for NTF, 26 for NTF2 and 4 for DOT. Exact duplicates of the reference copies were removed automatically. The coders were asked to identify near-duplicates of the known form letters, and to assign labels such as *"block added", "block deleted", "minor change", "block rearrange", "singleton"* (unique comment) and *"repeated copies"*, which are near-duplicate subcategories. The coders also annotated *"header", "signature line" "stakeholder"* and *"unique text"* (the substantive text that the commenter added to a form letter). "Block added" and "key blocks" are found to be the most common editing styles (>50%). "Minor change" is the next major one (>20%). The number of "singleton" clusters is reasonably large (around 10%), which is of potential interest for social science research.

5.2 Evaluation Metrics

Our experiments used two types of metrics. Precision, Recall, and F1 measure effectiveness objectively. And AC1 [6], a modified version of Cohen's kappa intercoder agreement, measure effectiveness relatively. Cohen's kappa is a more common choice, but it suffers from bias and prevalence problem when agreement between assessors is high but skewed to a few categories, as is common in public comment datasets. AC1 corrects this flaw and is defined as:

$$AC1 = \frac{p(A) - p(E)}{1 - p(E)} \qquad (10)$$

where *p(A)* is the observed agreement between two assessments *x* and *y*, *a* is the number of pairs in the same groups in both *x* and *y*, b is the number of pairs in the same groups in *x* but different in *y*, c is the number of pairs in the different groups in *x* but the same in *x*, and d is the number of pairs in the different groups in both *x* and *y*. *p(A)* can be calculated as *(a+d)/m* where *m=a+b+c+d*. *p(E)* is the agreement expected by chance, and is calculated as:

$$p(E) = 2P(1 - P) \qquad (11)$$

where *P = ((a+b)+(a+c))/2m.*

6. EXPERIMENTAL RESULTS

6.1 System-to-Human Intercoder Agreement

The first set of experiments explores the agreement between DURIAN and human assessors. DURIAN is treated as if it is a "coder". The interceder agreements are measured using AC1. In Table 1, Coder A represents coders UCSUR 13 for dataset NTF, UCSUR8 for NTF2 and SUPER for DOT; and Coder B represents UCSUR 16 for NTF, UCSUR9 for NTF2 and G for DOT[3]. Note that the human assessors have very high inter-coder agreement (>90%) on what is a near-duplicate cluster. It shows that for clustering algorithms aiming to solve a clearly-defined task such as near-duplicate detection, providing a "gold standard" is possible. Under this condition, DURIAN shows high system-human intercoder agreements in both macro-averaged (0.82 to 0.94) and micro-averaged AC1 (0.91 to 0.98). Moreover, the system-human intercoder agreements are comparable to human-human intercoder agreements. This is really desirable but we

[3] UCSUR13, UCSUR8, UCSUR9, UCSUR16, SUPER and G are identification numbers for human assessors.

Table 2: Comparison of Duplicate Detection Technologies

Duplicate sub-category	Algorithm	Avg. Precision			Avg. Recall			Avg. F1		
		NTF	NTF2	DOT	NTF	NTF2	DOT	NTF	NTF2	DOT
Exact	Full	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	DSC	0.97	0.98	0.93	0.98	0.97	0.97	0.97	0.97	0.95
	I-Match	0.91	0.90	0.90	0.75	0.85	0.60	0.82	0.87	0.72
	DURIAN	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Minor Change	Full	**0.95**	**0.95**	0.96	0.95	0.95	0.96	0.95	0.95	0.96
	DSC	0.90	0.90	0.88	0.90	0.90	0.90	0.90	0.90	0.89
	I-Match	0.79	0.80	0.80	0.78	0.77	0.80	0.78	0.78	0.80
	DURIAN	**0.95**	**0.95**	**1.00**	**1.00**	**0.98**	**1.00**	**0.97**	**0.96**	**1.00**
Block Added /Keyblock	Full	0.97	**0.98**	0.90	**0.98**	**0.98**	0.86	0.97	**0.98**	0.88
	DSC	0.73	0.70	0.30	0.78	0.73	0.30	0.75	0.71	0.30
	I-Match	0.32	0.35	0.20	0.42	0.36	0.30	0.36	0.35	0.24
	DURIAN	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**
Block Deleted	Full	0.90	0.90	0.90	**1.00**	**0.98**	0.90	0.95	0.94	0.90
	DSC	0.72	0.75	0.72	0.78	0.73	0.70	0.75	0.74	0.71
	I-Match	0.30	0.33	0.35	0.40	0.36	0.35	0.34	0.34	0.35
	DURIAN	**0.98**	**0.98**	**0.96**	0.98	**0.98**	**0.96**	**0.98**	**0.98**	**0.96**
Singleton	Full	0.90	0.90	**1.00**	0.90	0.93	**0.98**	0.90	0.91	**0.99**
	DSC	0.72	0.74	0.85	0.80	0.76	0.84	0.76	0.75	0.84
	I-Match	0.84	0.88	0.90	0.80	0.81	0.90	0.82	0.84	0.90
	DURIAN	**0.94**	**0.94**	0.98	**0.94**	**0.94**	**0.98**	**0.94**	**0.94**	0.98
Rearrange	Full	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	DSC	0.67	0.83	0.50	0.83	0.67	0.56	0.74	0.74	0.53
	I-Match	**1.00**	**1.00**	0.90	**1.00**	**1.00**	0.87	**1.00**	**1.00**	0.88
	DURIAN	**1.00**	**1.00**	0.90	**1.00**	**1.00**	0.90	**1.00**	**1.00**	0.90

should be careful about the micro-averaged AC1. There are some very large letter-writing campaigns in these datasets, which creates huge near-duplicate clusters, which tend to dominate the final result of pair-wise comparisons.

6.2 Impact of Instance-level Constraints

Figure 4 illustrates the impact of different types of constraints on the clustering algorithm. Note that our baseline is a unsupervised clustering algorithm with smoothed KL divergence as the distance metric. For the NTF dataset, baseline has an average F1 of 0.81. F1 reaches 0.96 after 50 constraints of all three types are used (an improvement of 18.5% over the baseline. For the NTF2 data set, the baseline has an average F1 of 0.85, and F1 reaches 0.97 after using 50 constraints (an improvement of 14.1% over the baseline). For the DOT data set, the baseline has an average F1 of 0.90. F1 is at 0.98 after 50 constraints are incorporated, (an improvement of 8.9% over the baseline). Our results on these three data sets indicate that instance-level constraints have greatly boosted the clustering effectiveness.

It can also be observed that some constraints are more effective than others. For both NTF and NTF2, the must-link constraints alone outperform the cannot-link, family-link and even the combination of all three types of constrains. However, on the DOT dataset, the cannot-link works the best of the three types of constraints. This phenomenon reminds us that NTF and NTF2 are drawn from the same collection, in which there are multiple major letter-writing campaigns. In this case, must-links guarantee that two documents with only a small amount of text overlap (low text similarity) are still in the same cluster as long as they satisfy must-link condition. In this case, must-links help to guide many outliners into their correct clusters and thus have a big impact on the overall clustering process. On the other hand, must-link is less effective for the DOT dataset, in which there are very few letter writing campaigns, and hence few large clusters. In this case, cannot-link constraints are of greatest use because they encourage the creation of more clusters, (correctly) resulting in more singleton clusters. Moreover, family-links speed up the clustering process by introducing bias and clustering documents into the correct clusters at an earlier stage.

6.3 Comparing with Other Duplicate Detection Algorithms

A set of experiments is conducted in this work to evaluate several well-known duplicate-detection algorithms on the above three datasets. The set of gold standards used here are the assessments from human coders UCSUR16 for dataset NTF, UCSUR15 for NTF2 and G for DOT. In order to study the effectiveness of duplicate detection techniques on subcategories, the average precision, average recall, and average F1 for each subcategory are studied for competing algorithms. The parameters for the competing methods were tuned using parameter sweeps and/or the best values reported in prior publications [5][7]. The contenders are described below.

Full fingerprinting (full): Every substring of size *s* in the documents is selected and hashed (*s*=3 in our experiments). Every hash value and the document id are stored as a <hash-value, document id> tuple. Duplicate detection is performed by 1) sorting the list of tuples; 2) for any hash-value that also appear in the reference copy, turning this hash-value's flag to 1, keeping the document id information to create another kind of tuple <document id, flag=1>; 3) counting the hash-values with flag=1 for every document, and get <document id, count>. The count is the number of overlap fingerprints between a document and the reference copy. If the overlap is above 80%, the document is considered as a (near) duplicate.

Shingling (DSC) [3]***:*** Performed in the same way as described in full fingerprinting except that every 5 overlapping substrings of size *s* in the documents are selected and hashed. *s* is set to 3 again. If the count of the overlap fingerprints in a document to form letter is above 80%, it is considered as a (near) duplicate.

I-Match [4]***:*** *N* words with the highest idf values in a document are selected, (*N* = 30). Note that the 5 words with the highest idfs are ignored, because they might be mistakes such as misspellings. A single fingerprint is generated for each document. Duplicate detection is performed by sorting all <fingerprint, document id> tuples. Those agreeing with the fingerprint of the form letter are selected as the (near) duplicates.

DURIAN: The algorithm proposed in this paper.

Based on results from Table 2, we can see that full fingerprinting and DURIAN are the most effective algorithms. Full fingerprinting gives the largest possible set of fingerprints for a document. It provides either the best or the second best F1 value in every subcategory. However, it is also the most computationally expensive method. Every substring is stored as a hash number, so the algorithm requires sorting a large list of tuples. On the other hand, DURIAN consistently performs well in all subcategories, occasionally beating the full fingerprint

Table 3: Execution time (in seconds).

	NTF	NTF2	DOT
Full fingerprint	2,233	2,054	1,878
DSC	1,674	1,584	1,444
I-Match	763	568	532
DURIAN	1,566	1,333	933

approach. However, the retrieval and detection time is much lower than for full fingerprinting. Table 3 shows that it uses less than 40% time than Full fingerprinting. Two other techniques DSC and I-Match are not as effective as Full fingerprinting and DURIAN. In general, DSC outperforms I-Match. I-Match is very sensitive to both "block added" and "block deleted", "minor change" editing patterns. When the changed words are critical, i.e., appearing in the fingerprint that I-Match selected, the algorithm fails to detect the near-duplicates. In general, I-Match produces fairly low Precision and Recall.

7. CONCLUSION

As text collections grow in size, and are assembled from diverse sources, duplicate and near-duplicate text becomes an increasingly important problem. Notice and comment rulemaking is an extreme of this problem, but the underlying form letters are usually easy to identify, and comments often arrive with metadata that provides additional clues about near-duplicate relationships.

This paper proposes instance-level constrained clustering as a solution to near-duplicate detection for notice and comment rulemaking. Instance-level constrained clustering has the advantage that varied information based upon document attributes, information extracted from the document text, and structural relationships among pairs of documents can be expressed as constraints on cluster contents, which narrows the search space, thus improving accuracy and efficiency. Experiments with EPA and DOT datasets demonstrate that this approach to near-duplicate detection is about as effective as high-quality manual assessment, at less computational cost than competing methods.

Although notice and comment rulemaking is a problem with distinct characteristics, instance-based constrained clustering is a general solution that can be applied broadly. The ability to incorporate diverse constraints makes it a powerful tool for combining multiple forms of textual and non-textual evidence.

ACKNOWLEDGMENTS

This research was supported by NSF grants EIA-0327979 and IIS-0429102. We are grateful to the USDA, US DOT, and US EPA for providing the public comment data that made this research possible. We are grateful to invaluable comments from the anonymous reviewers.

8. REFERENCES

[1] S. Brin, J. Davis, and H. Garcia-Molina. Copy detection mechanisms for digital documents. In Proceedings of the Special Interest Group on Management of Data (SIGMOD 1995), pages 398–409. ACM Press, May 1995.

[2] Y. Bernstein and J. Zobel, A scalable system for identifying co-derivative documents. In Proceedings of the String Processing and Information Retrieval Symposium (SPIRE), page 55-67, Padova, Italy, September 2004.

[3] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. In Proceedings of WWW6 '97, pages 391–404. Elsevier Science, April 1997.

[4] A. Chowdhury. O. Frieder, D. Grossman, and M. McCabe. Collection statistics for fast Duplicate document detection. In ACM Transactions on Information Systems (TOIS), Volume 20, Issue 2, 2002.

[5] J. Conrad and C. P. Schriber. Online duplicate document detection: signature reliability in a dynamic retrieval environment. Proceedings of the twelfth international conference on Information and knowledge management, pages: 443-452, New Orleans, LA, USA, 2003.

[6] K. Gwet. Kappa Statistic is not Satisfactory for Assessing the Extent of Agreement between Raters. Statistical Methods for Inter-rater Reliability Assessment, No.1, April 2002.

[7] T. Hoad and J. Zobel. Methods for identifying versioned and plagiarized documents. In Journal of the American Society or Information Science and Technology, Vol 54, I 3, 2003.

[8] D. Klein, S. D. Kamvar, and C. D. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In Proceedings of the 19th International Conference on Machine Learning, pages 307-314, 2002.

[9] A. Kołcz, A. Chowdhury, J. Alspector. Improved Robustness of Signature-Based Near-Replica Detection via Lexicon Randomization. Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, page 605-610, Seattle, WA, USA, 2004.

[10] D. Metzler, Y. Bernstein and W. Bruce Croft. Similarity Measures for Tracking Information Flow, Proceedings of the fourteenth international conference on Information and knowledge management, CIKM'05, October 31.November 5, 2005, Bremen, Germany.

[11] NIST, "Secure Hash Standard", Federal Information Processing Standards Publication 180-1, 1995.

[12] N. Shivakumar and H. Garcia-Molina. SCAM: a copy detection mechanism for digital documents. In Proc. International Conference on Theory and Practice of Digital Libraries, Austin, Texas, June 1995.

[13] S.W. Shulman, E-Rulemaking: Issues in Current Research and Practice. International Journal of Public Administration 28: 621-641. 2005.

[14] K, Wagstaff and C, Cardie, 2000. Clustering with instance-level constraints. In Proceedings of ICML-2000. pp. 1103–1110. Palo Alto, CA.

[15] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In Advances in Neural Information Processing Systems, 2003.

[16] H. Yang and J. Callan. Near-Duplicate Detection for eRulemaking. In Proceedings of the 5th National Conference on Digital Government Research (DG.O2005), Atlanta, GA, USA, 15-18 May 2005.

[17] C. Zhai and Lafferty, J. (2001b). A study of smoothing methods for language models applied to ad hoc information retrieval. In Proceedings of SIGIR 2001, pages 334-342.