Identification of Duplicate News Stories in Web Pages

John Gibson, Ben Wellner, Susan Lubar

The MITRE Corporation
202 Burlington Rd.
Bedford MA 01730
USA

Abstract

Identifying near duplicate documents is a challenge often faced in the field of information discovery. Unfortunately many algorithms that find near duplicate pairs of plain text documents perform poorly when used on web pages, where metadata and other extraneous information make that process much more difficult. If the content of the page (e.g., the body of a news article) can be extracted from the page, then the accuracy of the duplicate detection algorithms is greatly increased. Using machine learning techniques to identify the content portion of web pages, we achieve duplicate detection accuracy that is nearly identical to plain text and significantly better than simple heuristic approaches to content extraction. We performed these experiments on a small, but fully annotated corpus.

1. Introduction

News articles published on the Internet typically appear on many different websites in either identical or revised form. For users, identical and nearly identical duplicates are an annoyance. Duplicates slow down the process of finding new information on a topic, and potentially cause missed information if the user mistakenly identifies two documents as identical duplicates when in fact one contains new information. For automated processing such as named entity recognition and visualization, redundant data can cause incorrectly weighted results, markedly skewing search engine results and automated text processing applications.

While it is straightforward to find identical news stories in plain text documents, finding identical news stories embedded in web pages is considerably more complex. This is due to the large amount of "extraneous" information, such as navigation links, ads, Javascript, and other miscellaneous content contained in these pages. While the actual news story text on two separate web pages may be identical, the extraneous content on the pages will not be. Thus standard approaches for determining identical duplicates will fail.

In our system, named CEDAR which stands for Content Extraction and Duplicate Analysis and Recognition, we have taken a two-part approach to this problem. First, we have created a method for extracting news story text from web pages that is not website-specific (Gibson et al., 2007). We then use the extracted content to identify pairs of documents with the same news story content, ignoring any extraneous information on the pages. By calculating a resemblance score for pairs of documents based on a technique called shingling (Broder et al., 1997), we can identify both identical and near duplicate news stories.

2. Background

Duplicates are undesirable for many types of data. These include databases, mailing lists, file systems, email and image data. It is common practice to locate identical pieces of data using hashing strategies. Each piece of data is hashed using one of the standard algorithms, such as MD5. Any data represented by the same hash value is considered to be an identical duplicate.

Near duplicate documents are typically determined by computing a similarity score for each pair of documents in a collection. As presented by Chowdhury (Chowdhury et al., 2002) the two most common similarity measures are *resemblance* (Broder et al., 1997) and *cosine similarity* (Salton et al., 1975).

2.1. Cosine Similarity

To compute cosine similarity, documents are mapped into a vector space, typically based on term weights. The weight for each term is computed by the number of occurrences of the term in the document and an inverse measure of its frequency across a document collection. Document similarity is then measured by the cosine distance between the vectors.

2.2. Shingling

To compute the resemblance of two documents, each is broken into overlapping fragments called shingles. To do this, a shingle length, α, is specified. The first shingle is comprised of the first α words of the document. The second shingle consists of the second word in the document through the word located at $\alpha + 1$, and so on. Resemblance for the two documents is computed as the intersection size of the two documents' shingle sets divided by the size of the union of these sets. Let A and B denote the two sets of shingles for two distinct documents, then their resemblance is defined as:

$$r(A,B) = \frac{|S(A) \cap S(B)|}{|S(A) \cup S(B)|}$$

One of the main drawbacks to shingling is the massive number of shingles generated, especially for large documents. Several strategies are used to reduce the number of shingles, while only slightly reducing the effectiveness of the algorithm. The first is sketching, which is the process of taking a uniform subset of the shingles, for example by discarding all shingles S for which $S \mod 25 \neq$

Approved for public release; Distribution Unlimited; Case #08-0238

0 (Broder et al., 1997). A further refinement is supershingling, which takes the shingles for each document that remain after sketching and shingling them again. These second order shingles are supershingles. If two documents have just a single supershingle in common then they can be considered near duplicates. A final performance enhancement comes from discarding very common shingles. Broder et al. (Broder et al., 1997) experiment with the AltaVista dataset; they discarded all shingles that appeared in 1000 or more documents (Broder et al., 1997). This greatly reduces the size of the shingle sets which leads to large space and time savings.

2.3. Locality Sensitive Hashing

As an alternative to computing the resemblance or cosine distance, there exist hash functions that yield similar values for similar documents. These functions produce collisions with a probability equal to the resemblance or cosine distance. Or as a formal definition from Charikar (Charikar, 2002):

A locality sensitive hashing scheme is a distribution on a family F of hash functions operating on a collection of objects, such that for two objects x, y,

$$Pr_{h \in F}[h(x) = h(y)] = sim(x, y)$$

The main performance advantage of this approach is that the resulting hash values require much less storage space than the shingles of a document or the full term vectors of the cosine distance approach.

3. Related Work

3.1. Duplicate Detection

Recently Henzinger combined shingling and locality sensitive hashing (LSH) to achieve good results on a very large dataset (1.6 billion distinct web pages) (Henzinger, 2006). Henzinger's experiment was particularly interesting because a subset of the data was evaluated manually which allowed for a more rigorous assessment of the accuracy of the technique. The combined algorithm first used shingling and then applied an LSH that estimates the cosine distance. The shingling portion used a variant of Fetterly, et al.'s algorithm (Fetterly et al., 2003) (which is a variant of Broder, et al.'s algorithm). Specifically it used a minvalue sketching technique followed by supershingling. Pages were considered near duplicates if they had at least 2 matching supershingles. Next, the LSH was applied to all of the documents identified as near duplicates by the first pass. Then the bit sequence for each document was divided into pieces and any pair of documents that had a single piece in common were compared more thoroughly. If at least 355 of the 384 bits of the hash matched then the pair was kept as a near duplicate. The combined algorithm yielded substantial gains in precision with only a moderate impact upon recall.

A major difference between Henzinger's work and our own is that it has a different definition of near duplicates. Henzinger's near duplicates are documents that differ not in content, but in boilerplate or other miscellaneous page structure (session ids, username, hit count, etc.). We would consider those to be identical duplicates. This distinction is important because the aim of Henzinger's experiment was to eliminate redundant documents, while ours is to identify all of the related versions of each document. Henzinger's results are difficult to compare to ours for this reason; additionally the experiments in that work operate over a much larger set of web documents than we examine here, are focused more on scalability and examines all types of documents randomly taken from the Web rather than just news stories as in our work. That being said, Henzinger reports pair-wise precision scores of 79%. The level of recall is unknown because only a subset of their corpus was annotated. However, the combined algorithm did have a pair-wise recall score of 79% on the results of the shingling algorithm alone.

3.2. Content Extraction

Content extraction tools such as Columbia University's Crunch aim to reduce the size of web pages by removing what they deem as noise or clutter from the pages (Gupta et al., 2005). Additionally, a tool by the Document Analysis and Recognition Team (DART) at BCL Computers Inc. further reduces text by providing a summary of what remains (Rahman et al., 2001). These tools are motivated by a variety of goals including paring down pages for the visually impaired (Gupta et al., 2005), producing lighter weight content for small screen devices such as PDAs (Gupta et al., 2005; Rahman et al., 2001) and reducing page complexity for subsequent processing as in MetaNews (Kang and Choi, 2003) and in the CLEANEVAL challenge task as part of the Web as a Corpus Workshop (WAC 2007, 2007).

These Content Extraction tools are related to our work in that they are focused on determining which parts of web pages are relevant to their goal. However, they address the broader problem of operating on any type of web page, while we are focused solely on pages containing news articles.

3.3. News Story Content Extraction

MetaNews (Kang and Choi, 2003) and the Columbia Newsblaster project (Evans et al., 2004) both concentrate on gathering news articles on the web. MetaNews uses a two-phased approach. First, it carries out noise removal by throwing out HTML tags that it believes will not contain content. Next it uses pattern matching on the reduced page to extract news articles. Patterns for MetaNews are manually defined for each news site, and no automatic learning is involved. Thus although pattern writing is simpler than for traditional wrapper approaches, this tool is still likely to fail if a page format changes, and adding new sites requires some manual labor.

The Columbia Newsblaster team originally used individual site wrappers to identify news articles. They determined that this approach was difficult for handling new sites. As a result, they implemented a machine learning based approach which is similar to ours. The module relies on “simple surface characteristics of the text” to classify blocks of text as part of an article, or into various other categories such as title, caption, or other.

4. Data

4.1. Harvesting

To create a data set containing duplicate news stories, we started by obtaining article titles from the Reuters and Associated Press (AP) RSS feeds. [1] We then sent each title as a query to Google News and downloaded the top ten results for each query. Because of the large variance in the pagination methods of each site we limited downloads to the first page of each article. In total we harvested 2577 documents from 49 separate websites. We only annotated a subset of these; the resulting data set included 1621 documents from 27 different websites.

After harvesting, we took two steps to turn the data into a reference standard that could be used for training and scoring. First, we annotated the documents to indicate which portions of text were news story content. Second, we identified and recorded which document pairs contained identical or near duplicate news stories.

4.2. News Article Annotation

Manually annotating 2577 documents was a daunting task, so we instead automated the process by writing site-specific taggers. In the end, we reduced the size of our dataset for a variety of reasons. First, we concluded that it was not worth the effort to write taggers for sites that contributed only a few documents to the dataset. Secondly, there were two sites whose page format would have been difficult to annotate automatically. Finally, we decided to exclude over 500 articles that came directly from various Reuters sites. We felt that it would not be as interesting to use these articles as a basis for content extraction because it is possible to obtain a plain text feed directly from Reuters. As a result of these exclusions, our final dataset was comprised of 1621 documents from 27 different websites.

For each of these documents, we first corrected poorly formatted HTML using Beautiful Soup[2] and Tagsoup.[3] We then inserted tags around the body of the news article. Where possible, we also tagged the article's author, date, location, source, and title.

4.3. Duplicate Identification

The second step in creating a reference standard was to identify and record duplicate pairs in the document collection. First we located identical pairs by normalizing the article text and comparing all of the documents directly. This analysis found that there were 564 distinct articles.

Marking near duplicate pairs was carried out as a manual process because it requires a user's judgment to determine whether two articles are near duplicates. We implemented a basic GUI to accelerate the process. It displays two articles side by side, and highlights the differences between the articles' content. A picture of the viewer appears in Figure 1. Because there are over 1 million possible pairings of 1621 documents, a manual comparison of all possible pairs would have been extremely time consuming. Therefore, we used a few techniques to speed up the process. First we display only one document for each unique article as identified by the first pass, bringing the total number of documents for comparison down to 564. When a document pair was marked as near duplicates we automatically added near duplicate pairings for all of the exact duplicates of the two marked documents. Because of the method we used for harvesting documents, we were confident that the bulk of the duplicates would also share the same title. Thus we began by reviewing document pairs that had similar titles. Once we had implemented our shingling algorithm we used it to find the few remaining documents that were near duplicates but had different titles.

We had six annotators review the data and record near duplicate pairs. Because determining whether a fairly different, but related pair is somewhat subjective, we held group discussions to determine the status of questionable pairs. Also, after the first pass of all of the documents was completed, we assigned a different annotator from the original group to carry out a second pass and verify the pairs that were recorded. In the end the reference standard contained 3591 identical duplicate pairs and 1231 near duplicate pairs.

5. Content Identification

Once we had created a reference standard we used the data to develop a machine learning-based system for identifying the content of the news articles.

5.1. Division into Blocks

The first step in our approach was to divide the web page into smaller pieces for our algorithms to identify as CONTENT or NOTCONTENT.

We sanitized the raw HTML by transforming it into XHTML using Tagsoup and Beautiful Soup. In the rare case that a document that could not be transformed into XHTML, we discarded it. Otherwise we tokenized the document. We excluded all words inside style and script tags from tokenization. It is safe to assume that those tags will never contain any content because they will not be rendered by a browser.

Next we partitioned the sequences of tokens into blocks. Intuitively, we define a block as a sequence of text that when rendered in a browser does not cause a line break. More formally we defined blocks as sequences of text that are bounded by any tag except the following: `<a>`, `<ins>`, `<del>`, `<span>`, `<bdo>`, `<em>`, `<strong>`, `<dfn>`, `<code>`, `<samp>`, `<kbd>`, `<var>`, `<cite>`, `<abbr>`, `<acronym>`, `<q>`, `<sub>`, `<sup>`, `<tt>`, `<i>`, `<b>`, `<big>`, `<small>`, `<u>`, `<s>`, `<strike>`, `<basefont>`, and `<font>`.

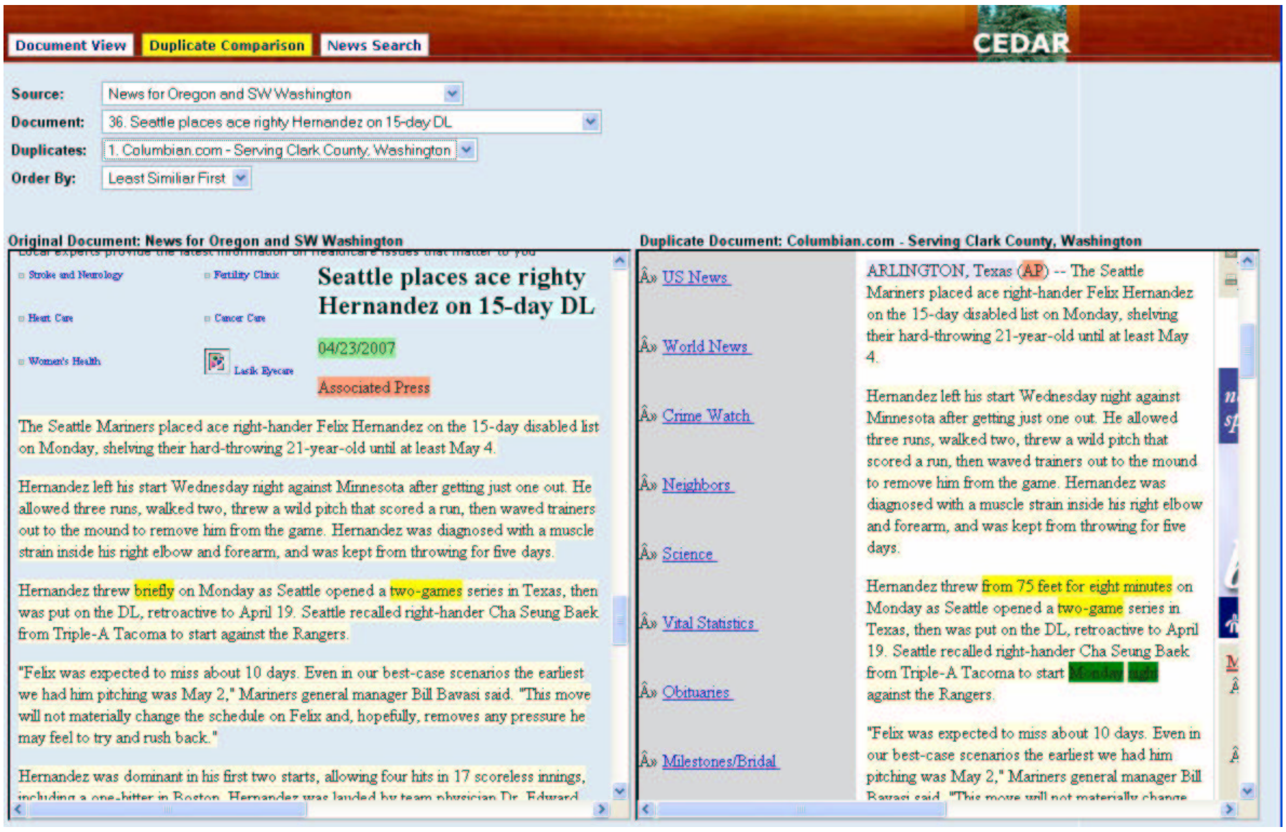
5.2. Feature Generation and Classification

Once the individual blocks were identified we generated a variety of feature types from each block that we subsequently used to train our algorithms. These feature types included a simple bag of words with frequency, a count of the tokens in a block, the percentage of tokens in a block that were contained within an anchor tag (`<a>`), tags in and

[1]We used different feeds from http://www.reuters.com/tools/rss and http://hosted.ap.org/dynamic/fronts/RSS?SITE=AP.

[2]Beautiful Soup was written by Leonard Richardson, et al. and can be found at http://www.crummy.com/software/BeautifulSoup/

[3]TagSoup was written by John Cowan and can be found at http://ccil.org/ cowan/XML/tagsoup/



Source: News for Oregon and SW Washington
Document: 36. Seattle places ace righty Hernandez on 15-day DL
Duplicates: 1. Columbian.com - Serving Clark County, Washington
Order By: Least Similiar First

Original Document: News for Oregon and SW Washington

Local experts provide the latest information on healthcare issues that matter to you
Stroke and Neurology
Fertility Clinic
Heart Care
Cancer Care
Women's Health
Lasik Eyecare

Seattle places ace righty Hernandez on 15-day DL

04/23/2007

Associated Press

The Seattle Mariners placed ace right-hander Felix Hernandez on the 15-day disabled list on Monday, shelving their hard-throwing 21-year-old until at least May 4.

Hernandez left his start Wednesday night against Minnesota after getting just one out. He allowed three runs, walked two, threw a wild pitch that scored a run, then waved trainers out to the mound to remove him from the game. Hernandez was diagnosed with a muscle strain inside his right elbow and forearm, and was kept from throwing for five days.

Hernandez threw briefly on Monday as Seattle opened a two-games series in Texas, then was put on the DL, retroactive to April 19. Seattle recalled right-hander Cha Seung Baek from Triple-A Tacoma to start against the Rangers.

"Felix was expected to miss about 10 days. Even in our best-case scenarios the earliest we had him pitching was May 2," Mariners general manager Bill Bavasi said. "This move will not materially change the schedule on Felix and, hopefully, removes any pressure he may feel to try and rush back."

Hernandez was dominant in his first two starts, allowing four hits in 17 scoreless innings, including a one-hitter in Boston. Hernandez was lauded by team physician Dr. Edward

Duplicate Document: Columbian.com - Serving Clark County, Washington

ARLINGTON, Texas (AP) -- The Seattle Mariners placed ace right-hander Felix Hernandez on the 15-day disabled list on Monday, shelving their hard-throwing 21-year-old until at least May 4.

Hernandez left his start Wednesday night against Minnesota after getting just one out. He allowed three runs, walked two, threw a wild pitch that scored a run, then waved trainers out to the mound to remove him from the game. Hernandez was diagnosed with a muscle strain inside his right elbow and forearm, and was kept from throwing for five days.

Hernandez threw from 75 feet for eight minutes on Monday as Seattle opened a two-game series in Texas, then was put on the DL, retroactive to April 19. Seattle recalled right-hander Cha Seung Baek from Triple-A Tacoma to start Monday night against the Rangers.

"Felix was expected to miss about 10 days. Even in our best-case scenarios the earliest we had him pitching was May 2," Mariners general manager Bill Bavasi said. "This move will not materially change

Figure 1: Example duplicates

around the block, inverse stop wording, named entities, and a few other feature types. A complete list with details can be found in our previous paper (Gibson et al., 2007).

We experimented with a variety of machine learning approaches that made use of the above features to identify content containing blocks. A key insight is to not label each block *independently*, but rather to label them using sequential classification methods that take into account the whole sequence of decisions over all the blocks in a single article. Conditional Random Fields (CRFs), a flexible statistical model able to capture sequential dependencies while also allowing rich, arbitrary features proved to be the most successful approach (Gibson et al., 2007).

We also tried using a simpler maximum entropy model with only word features to judge the effect of the quality of the content extraction step upon duplicate detection.

5.3. Content Extraction Methodology

In order to determine the effectiveness of content extraction for duplicate identification, we needed to run our content extraction system over our entire duplicate data set. However, the content extraction system relies on the same articles as a source of training data. Further, many of the articles in the collection are duplicates, and many are from the same web-site. Ideally, in the most conservative and fair setting, the content extractor would not be trained on any of the same articles (including duplicates and near duplicates) or any of the same sources as to which it is applied.

To achieve this on our collection we used a four-fold cross validation approach. Each of the four folds contained 6 or 7 sources, with each source only belonging to exactly one fold. We assigned sources to folds in round-robin fashion by selecting a source at random weighted by the number articles belonging to that source. This kept each fold at roughly the same size in terms of the number of articles.

Three folds constituted the training data while a fourth served as test data. This would allow each article to be processed with a content extractor that was trained on articles from web-sites not appearing in the test data. However, due to many duplicates in our corpus, the training data will likely contain many of the same articles as in the test data, though from different sources. To compensate for this problem, if any duplicate documents spanned the training and test data for a particular fold, we removed those articles from the training set. An illustration of this process can be seen in Figure 2. This process was repeated four times, with each fold taking its turn as test data.

6. Experiments and Results

Our experiments in this section aim to demonstrate 1) the effect that various system parameters and options have on overall accuracy and 2) how well the system, tuned on development data, performs on held-out evaluation data. We believe that the latter is an indicator of how well the system will perform on new web pages from news sources.

6.1. Evaluation Methods

Before describing our experiments in detail, we highlight two methods for evaluating duplicate detection accuracy.

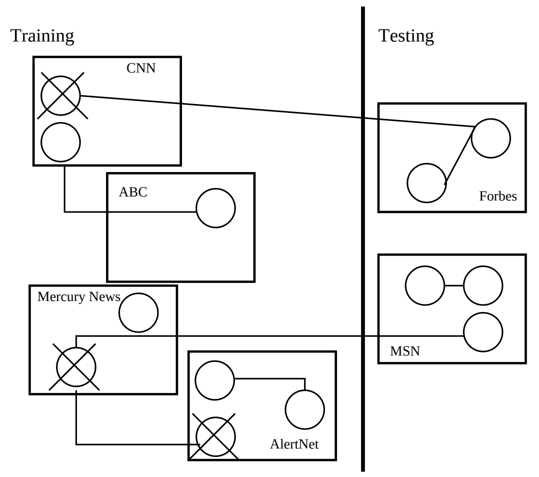


Figure 2: Data partitioning for content extraction model. Boxes indicate sources, circles are documents. Circles connected by lines are duplicates. Crossed out circles were dropped from the training set.

6.1.1. Pair-wise Evaluation

The most natural way to evaluate a duplicate detection system is with a pair-wise evaluation metric. Given a set of documents, the reference standard provides information as to whether each pair of documents is a duplicate pair. A system for duplicate detection is responsible for assigning a similarity score, s, to each pair of documents. Given a threshold, T, each pair with a score, $s \geq T$ is hypothesized to be a duplicate pair. For a particular threshold, T, we can evaluate the precision and recall of the duplicate system where precision is computed as:

$$\frac{\text{\# correct duplicate pairs}}{\text{\# hypothesized duplicate pairs}}$$

and recall computed as:

$$\frac{\text{\# correct duplicate pairs}}{\text{\# true (reference standard) duplicate pairs}}$$

For different applications, different threshold values may be appropriate depending on whether precision or recall is preferred. A way to compare the results of different systems across all threshold values is to look at Receiver Operator Characteristic (ROC) curves, which plots the true positive rate against the false positive rate, essentially capturing the precision rate at all possible recall levels. The area under this curve provides a single number useful for comparing two systems that assign scores to positive and negative instances. The area under the curve, in our context, can be interpreted as the probability that the system/model will provide a higher resemblance score to an arbitrary document pair p_1 than a pair p_2 when p_1 is, in fact, a duplicate pair and p_2 is not. We use the area under the ROC, AU-ROC curve for some of the results that follow. Note that, in particular, the graphs below plot how the AU-ROC changes as a parameter is adjusted - the graphs are not ROC curves themselves.

Note that for our full data set of 1621 web page documents, there are 1,313,010 unordered document pairs of which just 4815 are duplicate pairs.

6.1.2. Cluster-Based Evaluation

While pair-wise evaluation is intuitive and appropriate in many cases, it has the disadvantage of skewing the results when the pair-wise relation is an equivalence relation[4]. This is because mistakes made with documents belonging to larger equivalence classes will be penalized more than documents within small equivalence classes. This can provide for a rather unintuitive evaluation metric. The cluster-wise evaluation metric considers the degree to which the resulting equivalence classes match each other rather than considering all pair-wise relations.

Equivalence class-based evaluation metrics have been used for evaluation within document and cross document co-reference in natural language processing (Vilain et al., 1995; Amit and Baldwin, 1998). For our formal evaluation of the system, we use the B^3 scoring metric (Amit and Baldwin, 1998) and its implementation within the LingPipe suite of NLP tools[5]. Very briefly, the B^3 metric takes as input the reference standard clusters and the clusters derived from the system output (obtained in our case by asserting that all pairs with a resemblance above a certain threshold are duplicates and then taking the transitive closure). For a *document*, d, precision, P_d, and recall, R_d, are computed as follows:

$$P_d = \frac{\text{\# of correct documents in the output cluster containing } d}{\text{\# of documents in the output cluster containing } d}$$

and

$$R_d = \frac{\text{\# of correct documents in the output cluster containing } d}{\text{\# of documents in the truth cluster containing } d}$$

The final precision and recall scores for the output clusters are the average precision and recall scores across all the documents.

6.2. System Configuration Analysis

In this section we consider a set of experiments aimed at identifying the effect of different system parameters on duplication detection accuracy.

The first set of experiments looks at the effect of varying the system parameters:

Block size filter This option removes blocks (described above) if their word/token count is below a certain threshold. The threshold varies from 1 to 20.

Shingle frequency filter This option removes the N% most frequent shingles in the corpus, values range from 0.0% to 10.0%.

Shingle size This parameter controls the size of the shingles in words. It is an integer varying from 1 to 10.

[4]While we have not restricted our similarity relation to be transitive, the manually annotated duplicate pairs reveal a similarity relation that is transitive in nearly all cases. This may be a reflection on our particular data set, however.

[5]LingPipe is available at http://alias-i.com/lingpipe/

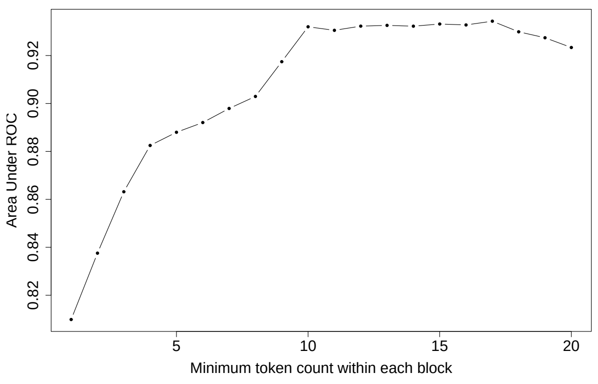


Figure 3: Duplicate detection accuracy with different minimum word counts per block with word shingles of size 6.

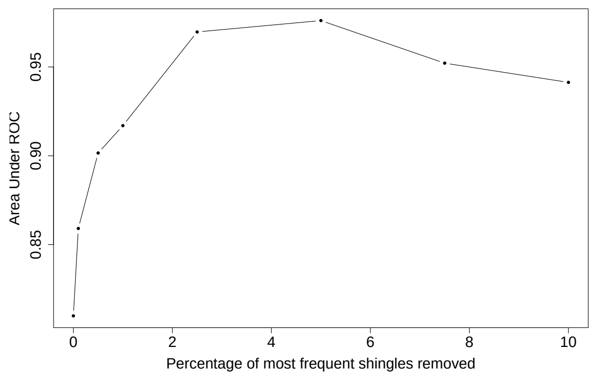


Figure 4: Duplicate detection accuracy with different shingle cutoff percentages using word shingles of size 6.

Figure 3 shows how duplication detection accuracy, in terms of AU-ROC (on the Y-axis), varies as more and more blocks are filtered from the documents based on the minimum token count (shown on the X-axis). As the minimum count is increased, accuracy improves until a minimum count of 10 with increasing minimum counts not resulting in noticeable improvement and eventually degradation in accuracy after reaching a minimum of 17 or more. The shingle size is fixed at 6 while the shingle frequency filter is inactive.

Figure 4 contains a graph illustrating the effect of removing varying percentages of the most frequent shingles. Accuracy improves even more substantially as compared with the block filter here. The explanation for this is that many of the shingles in the non-content portions of the document appear over and over again as part of the web-page boilerplate, in contrast to shingles found in the content portion of the article. Again, shingle size was fixed at 6 and no block-level filtering was performed.

A final graph is shown in Figure 5 demonstrating the effect different shingle sizes have on overall accuracy. The best duplicate identification accuracy is found with shingles of size 2. Surprisingly, single word shingling performs very well, outside the standard deviation of accuracy with shingles of size 4.

In addition to the above contributions, we also examined the effect of content extraction on accuracy. The hypothesis here is that extracting the article content by removing extraneous parts of the document, will improve dupli-

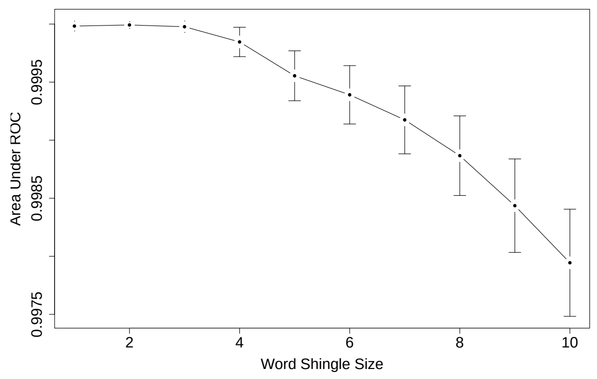


Figure 5: Duplicate detection accuracy with different word shingle sizes. The standard deviation bars, computed using the approach in (Hanley and McNeil, 1982), demonstrate significantly lower accuracy with larger shingles.

Shingle/Block Filter	AU-ROC
Heuristic	0.91076 ± 0.00283
CE-SIMPLE	0.9999922 ± 0.0000284
CE-BEST	0.9999933 ± 0.0000272
Ref. Standard	0.9999935 ± 0.0000259

Table 1: Content Extraction-based results using word shingles of length 2.

cate detection accuracy. We look at four different content extraction (CE) approaches here:

Heuristic Removal of document regions between `<script>` and `<style>` tags.

CE-SIMPLE A simple maximum entropy classifier to identify blocks containing article content. The feature set used for the classifier consisted of only the words present in the block. The document-level accuracy of this CE system is 58% – meaning that it is able to perfectly identify the content portion of a web-page 58% of the time. The block-level precision is 96.0 with recall at 98.1.

CE-BEST The automatic best content extraction system based on CRFs using a richer feature set that considered the presence of particular HTML tags, the presence of named entities (e.g. people, organizations, etc.) and other features. Document-level accuracy for this CE system is 80% with block-level precision and recall at 97.9 and 99.5, respectively.

Ref. Standard Only the portions of the documents considered content by site-specific taggers and reviewed by human annotators were used to compute resemblance.

The results in Table 1 illustrate the effects of these different methods on duplicate detection.

A final analysis shown in Table 2 looks at combining the block filter and the shingle frequency filter with and without using the heuristic-based content extraction. The results here indicate that using the heuristic-based CE *together* with filtering out the top 5% most frequent shingles and removing blocks with fewer than 12 tokens (bottom

Shingle/Block Filter	AU-ROC
Entire Web Document	
None	0.94275 ± 0.00233
MinBlock = 5	0.95883 ± 0.00200
Shingle Freq. = 7.5%	0.99751 ± 0.00005
Heuristic Content Extraction	
Heuristic Only	0.91076 ± 0.00283
Shingle Freq.= 7.5%	0.99500 ± 0.000718
Shingle Freq.= 5.0% & MinBlock = 12	0.99994 ± 0.000079

Table 2: Combinations of block filtering, shingle filtering and heuristic CE using shingles of length 2.

row) achieves very high results in terms of AU-ROC. These results appear to be competitive with duplication detection approaches that use a statistically trained CE system (in Table 1). However, as we demonstrate below, statistically-driven content extraction does significantly improve duplication detection accuracy over the best system that doesn't use statistical CE.

6.3. Formal Evaluation

In this section we describe the results of a somewhat more formal evaluation with a fixed development and test split of the data. While the above analysis based on AU-ROC provides insight into the contributions of different aspects of the system on overall accuracy, in a realistic setting one must pick a fixed threshold.

We split the data into two roughly equal-sized sets of documents such that no equivalence classes of duplicate documents (according to the reference standard) overlapped both sets. The development portion of the data was used to tune the threshold value and to optimize various parameters such as the minimum word count block filter and the shingle frequency cutoff. Table 3 shows the official results in terms of the cluster-based B^3 metric as well as the more standard pair-wise metric.

We also provide scores for using a normalized string comparison on the reference standard's content. This approach finds all of the identical duplicates and none of the near duplicates.

The most obvious result of these experiments is that effective content extraction provides a significantly higher level of accuracy than more basic techniques. By enabling the duplicate detection algorithm to focus on the *article content*, filtering out extraneous web page material, accuracy is improved considerably. The overall results here are very promising indicating that a very high percentage of duplicate document clusters can be identified perfectly.

Another interesting result here is that even a very simple machine learning content extraction approach provides for duplicate detection accuracy that is nearly identical to using the reference standard extracted content. Implementing this approach requires little effort provided a set of training documents with the content portions annotated (see (Gibson et al., 2007)).

7. Conclusions

Our approach to detecting identical and near duplicate news articles embedded in web pages is a two step process. Our system, CEDAR, first extracts the text of the news articles from the pages, and then computes resemblance scores for the articles using a shingling approach. We carried out a number of experiments which show that basing the duplicate detection purely on the extracted content results in more accurate results than computing resemblance across the text of the original documents.

Additionally, we have implemented a flexible, non-brittle approach for identifying news article text in web pages. We created a model for our CRF classifier based on the structure of web news pages across twenty-seven different news websites. The classifier can now be used to find news article text embedded in pages from previously unseen web sites, and does not break when the formatting on a web site changes.

Though this content extraction technique does not always provide perfect results, it is accurate enough to allow our duplicate detection system to outperform itself when using more naive approaches to identifying article text. Moreover, this content extraction module can be used to improve results for many different tasks involving news articles on the web. Some examples are named entity extraction, visualization, search indexing and display on a small screen such as a PDA or cell phone.

8. Future Work

Our future work is focused on making CEDAR deployable. For the content extraction this means optimizing the preprocessing and feature generation as well as moving to a faster implementation language (currently, we are using Python). In situations where the only purpose of extracting content is to improve duplicate detection accuracy, we can use the simpler MaxEnt system which requires many fewer features and is faster in general than the full CRF system.

Because of the relatively small size of the dataset, our duplicate analysis implementation was performed in memory. To handle larger datasets we will experiment with sketching and supershingling, as well as redesigning the code to work on smaller chunks at a time. We can also switch to an entirely different algorithm such as cosine distance or fingerprinting. Finally, if we were only concerned with improving the precision of a system, then we could use our system on the clusters generated by other systems. As long as the clusters themselves are not prohibitively large, our current performance limitations would not be an issue and we could eliminate a large number of false positives.

Another direction for our work is to experiment with processing foreign language text, particularly for languages written in other character sets. It is our expectation that translating the documents to English before shingling will generate poor results. However, by applying word segmentation techniques we should be able to achieve reasonable accuracy using word based shingling. Or, by normalizing the text to remove all segmentation whatsoever, shingling based on characters rather than words should achieve very good results. However, this approach will create an enormous number of shingles and therefore performance can

System	Threshold	Cluster-based			Pair-wise		
		Prec.	Rec.	F-meas.	Prec.	Rec.	F-meas.
Heur. and Shingle Freq. = 5%	0.35	0.929	0.854	0.89	0.977	0.782	0.869
Heur. and MinBlock = 8	0.4	0.919	0.860	0.889	0.919	0.860	0.889
Heur., Shingle Freq. = 5% & MinBlock = 9	0.35	0.981	0.862	0.917	0.981	0.862	0.917
CE-BEST	0.4	0.992	0.979	0.985	0.992	0.979	0.985
CE-SIMPLE	0.4	0.992	0.977	0.985	0.992	0.977	0.985
Ref.	0.4	0.992	0.977	0.985	0.992	0.977	0.985
Ref. String Comparison	N/A	1.0	0.802	0.89	1.0	0.745	0.854

Table 3: Cluster-based and pair-wise results on the evaluation data with threshold and system parameters tuned on the development data.

become unacceptably slow. Our plan is to experiment with character based shingling combined with sketching to increase speed while maintaining reasonable accuracy.

Acknowledgements

This work was supported by the Department of the Army's Communications-Electronics Lifecycle Management Command (C-E LCMC) and performed under MITRE Mission Oriented Investigation and Experimentation (MOIE) project M130 of contract W15P7T-07-C-F600, sponsored by the Pacific Regional Service Center. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the sponsor.

9. References

Amit, B. and B. Baldwin, 1998. Algorithms for scoring coreference chains. In *Proceedings of the Seventh Message Understanding Conference (MUC7)*.

Broder, Andrei Z., Steven C. Glassman, Mark S. Manasse, and Geoffrey Zweig, 1997. Syntactic clustering of the web. In *Selected papers from the sixth international conference on World Wide Web*. Essex, UK: Elsevier Science Publishers Ltd.

Charikar, Moses S., 2002. Similarity estimation techniques from rounding algorithms. In *STOC '02: Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*. New York, NY, USA: ACM.

Chowdhury, Abdur, Ophir Frieder, David Grossman, and Mary Catherine McCabe, 2002. Collection statistics for fast duplicate document detection. *ACM Trans. Inf. Syst.*, 20(2):171–191.

Evans, D. K., J. L. Klavans, and K. McKeown, 2004. Columbia Newsblaster: Multilingual news summarization on the web. In *Proceedings of Human Language Technology Conference/ North American Chapter of the Association for Computational Linguistics*.

Fetterly, Dennis, Mark Manasse, and Marc Najork, 2003. On the evolution of clusters of near-duplicate web pages. In *LA-WEB '03: Proceedings of the First Conference on Latin American Web Congress*. Washington, DC, USA: IEEE Computer Society.

Gibson, John, Ben Wellner, and Susan Lubar, 2007. Adaptive web-page content identification. In *WIDM '07: Proceedings of the 9th annual ACM international workshop on Web information and data management*. New York, NY, USA: ACM.

Gupta, S., D. Daiser, P. Grimm, M. Chiang, and J. Starren, 2005. Automating content extraction of html documents. *World Wide Web - Internet and Information Systems*, 8(2):179–224.

Hanley, JA and BJ McNeil, 1982. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143:29–36.

Henzinger, Monika, 2006. Finding near-duplicate web pages: a large-scale evaluation of algorithms. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM.

Kang, D. and J. Choi, 2003. Metanews: An information agent for gathering news articles on the web. In *International Symposium on Methodologies for Intelligent Systems*.

Rahman, A. F. R., H. Alam, and R. Hartono, 2001. Understanding the flow of content in summarizing html documents. In *International Workshop on Document Layout Interpretation and its Applications (DLIA)*.

Salton, G., A. Wong, and C. S. Yang, 1975. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620.

Vilain, Marc, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman, 1995. A model-theoretic coreference scoring scheme. In *MUC6 '95: Proceedings of the 6th conference on Message understanding*. Morristown, NJ, USA: Association for Computational Linguistics.

WAC 2007, Web as a Corpus, 2007. WAC2007. In *Web as a Corpus*. UCLouvain, Louvain-la-Neuve, Belgium.