ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Near Duplicate Document Detection Using Document-Level Features and Supervised Learning

Raveena.S, Nandini.V

PG Scholar, Department of CSE, Sona College of Technology, Salem, Tamil Nadu, India[1]

Associate Professor, Department of CSE, Sona College of Technology, Salem, Tamil Nadu, India[2]

ABSTRACT- This paper addresses the problem of Near Duplicate document. Propose a new method to detect near duplicate document from a large collection of document set. This method is classified into three steps. Feature selection, similarity measures and discriminant function. Feature selection performs pre-processing; calculate the weight of each terms and heavily weighted term is selected as a features of input document. As a result, Feature selection helps to select a set of features from an input document. Similarity measure measures the similarity degree between two documents. Discriminant derivation use SVM classifier to determine the discriminate function from document set based on supervised learning. As a result of this method, discriminant function is to check whether the document is near duplicate or not based on similarity degree. These document-level feature selections provide better (or) more efficient result than sentence-level feature selection.

KEYWORDS- Web mining, near duplicate document, Web database, Feature selection, Similarity measures, Training data, SVM Classifier, Discriminant function.

I. INTRODUCTION

As the World Wide Web is increasingly popular, digital documents are easily generated and put on the internet. By using a search engine, one can collect a large set of documents in a veryshort time (Chowdhury, Frieder, Grossman, & McCabe, 2002; Henzinger, 2006). Through the delete, copy, and paste commands provided by an editor or other tools (de Carvalho, Laender, Goncalves, & da Silva, 2012; Valls & Rosso, 2011), similar documents are likely to appear in various web communities (Conrad, Guo, & Schriber, 2003; Fetterly, Manasse, & Najork, 2003; Manku, Jain, & Sarma, 2007; Narayana, Premchand, & Govardhan, 2009; Pereira,Baeza-Yates, & Ziviani, 2006; Yang & Callan, 2005), e.g., blogs and forums. Such similar documents not only increase the volume of information one may have to go through but also require more storage and bandwidth for communication. To reduce the data volume and increase the search efficiency, detecting similar documents has become an important issue in the field of information retrieval.

Similar documents can be divided into two categories, duplicates and near-duplicates. Two documents are duplicates if they are totally identical (Broder, 2000). Two documents are near-duplicates if one document is a modification of the other document. The modification can be insertion, deletion, or replacement of part of the text(Yang & Callan 2006).Duplicate documents can be easily detected then the near duplicate document. These paper provide method to detect near-duplicate documents efficiently and effectively.

To detect near-duplicate documents, one can adopt the bag of-words model (Bag of words, 2012) for document representation. Let D = {d1,d2, . . . ,dn} be a set of n documents, in which d1,d2, . . . ,dn are individual documents. Each document di, $1 < i < n$, is represented by a feature set fi = {fi,1, fi,2,. . . , fi,m} where m is the number of features selected for D.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Conventionally, a manually designated threshold is provided by the user in advance. If the similarity degree is equal to or higher than the threshold, the two documents are near-duplicates. Otherwise, they are not. Usually, trial-and-error cannot be avoided. Setting a good threshold manually is neither an easy task nor an effective way for near-duplicate document detection.

Sentence-level feature selection provides less efficient result compare to document-level feature selection. Proposed a new method to detect near duplicate document from a large collection of document set. This method is classified into three steps, Feature selection, similarity measures and discriminant function. Feature selection performs pre-processing of input document; calculate the weight of each terms in a given document and heavily weighted term is selected as features of input document. As a result, Feature selection helps to select a set of features from an input document. Similarity measure measures the similarity degree between the given document and each document in a given collection. Discriminant derivation use SVM classifier to determine the discriminate function from training document set based on supervised learning. As a result of this method, discriminant function is to check whether the document is near duplicate or not based on similarity degree. These document-level feature selections provide better (or) more efficient result than sentence-level feature selection.

Supervised learning techniques, in particular support vector machines (SVM) (Martins, 2011), can be applied to determine optimally whether two documents are near-duplicates automatically. Given a training data set with instances belonging to one of two classes, near-duplicate and non-near-duplicate, SVM learns how to separate the instances of one class from the instances of the other class. As matter of fact, an optimal hyperplane can be derived which not only separates the instances on the right side of the hyperplane but also maximizes the margin from the hyperplane to the instances closest to it on either side. If the problem is not linearly separable, one can map the original space to a new space by using nonlinear basis functions. It is generally the case that this new space has many more dimensions than the original space, and, in the new space, the optimal hyperplane can be found.

II. RELATED WORKS

There are many existing technique available to detect near duplicate document. There are as follows:

(Manning, Raghavan, & Schutze, 2008)He proposed shingling technique to detect near duplicate document. Shingling algorithm views each document as a sequence of strings called shingles. Each string is k word long called k-gram. The list of such k-grams is taken to be the feature set of this document. For example, if a document consists of L words, then the feature set of the document contains L-K+1 element. As a result feature selection, measure similarity degree between two document using jaccard or other similarity function. Based on similarity degree, to detect document as a near duplicate document. Some improvements to shingles have been proposed.(Li et al. 2007) took discontinuous k-grams by skipping the words in between. The strings between two pause symbols are treated as features.

(Theobald et al. 2008)He proposed a technique spotsigs to detect near duplicate document. First scan the document to find stop words in it anchor's, k tokens right after an anchor excluding stop words are grouped as a special k-gram. So, called as "spot signature". A feature is taken to be a string starting with a stop word. For example, {*the super computer*} and {*a good movie*} are elements of the feature set. Different stop word lists lead to different feature sets for a given documentSpotSigs (Theobald et al., 2008) adopts some rules to cut down the size of a feature set, e.g., preferring more frequently used stop words. Other methods based on sentences were proposed (Wang & Chang,2009).With these methods, each individual sentence of a document is divided into a series of k-grams. The union of the k-grams of all the sentences is taken as the feature set of the document. However, these methods result in large feature sets for document representation.

(Chowdhury et al) presented an approach called I-Match for detection of near duplicate document detection. *I-Match* maps each individual document into a single hash value using the SHA1 hash algorithm. If hash values of two documents are identical, then two documents are near duplicate. The signature generation process of I-Match views a document as a single bag of words (i.e., terms, unigrams). In addition, only the "important" terms are retained in the

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

bag. It first defines an I-Match lexicon L based on collection statistics of terms using a large document corpus. A commonly used option is the inverse document frequency (*IDF*), where L consists of only terms with mid-range *IDF* values. For each document d that contains the set of unique terms U, the intersection $S = L \cap U$ is used as the set of terms representing d for creating the signature. One potential issue with I-Match occurs when the retained word bag S is small (i.e., $|\boldsymbol{s}| \ll |\boldsymbol{u}|$).Because the documents are effectively represented using only a small number of terms, different documents could be mistakenly predicted as near-duplicates easily. Two documents are considered near-duplicate only if they have enough number of signatures matched.

(Charikar et al.) Proposed a technique called "simhash". It is dimensionality reduction technique for near duplicate document. This technique obtain f bit fingerprint for each document. A pair of documents are near duplicate if and only if fingerprint of document atmost k bit apart. Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different. For example, if f=64 bit, k=3 bit Hamming distance (simhash(Q_1),simhash(Q_2)) $\leq$ k, then (Q_1,Q_2) are near duplicate document.

III. PROPOSED SYSTEM

In general, replaced terms, inserted terms, and missed terms are cases frequently occurring in near-duplicate documents. For the near-duplicate detection methods based on terms, e.g., Shingles, different document representations may be adopted and diverse results can be obtained. For example, suppose we have a document A of the following.

A: People rally on the **sidewalk** as legal arguments over the Patient Protection and Affordable Care Act take place at the Supreme Court.

By replacing sidewalk with pavement, we get another document B.

B: People rally on the **pavement** as legal arguments over the Patient Protection and Affordable Care Act take place at the Supreme Court.

Document A and B have the same meaning, and it is no doubt that they are near-duplicates. However, existing methods may obtain different representations for these two documents. For example, the k-gram feature selection method (Manning et al., 2008) produces several k-grams containing the word sidewalk for document A, while it produces several k-grams containing the word pavement for document B. Therefore, different representations are obtained. Also, for the k-gram method, different values of k could affect both the feature set size and the required computation power. To solve this problem, we propose a method for extracting features from individual sentences in a way to better reveal the characteristics of a document. The method turns out to be more invariant against insertion, deletion, or replacement of terms. As a result, the feature sets obtained are more suitable for near duplicate document detection.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

3.1 ARCHITECTURAL DESIGN

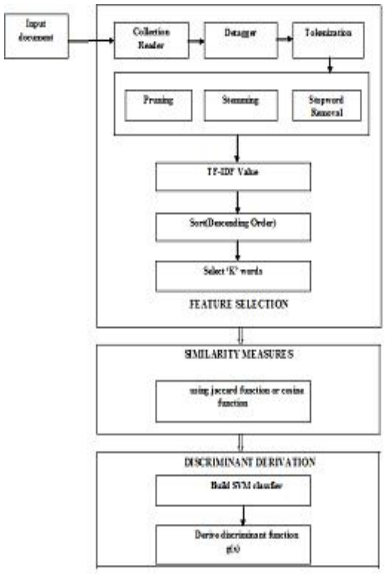

Figure 1 Architectural Design

3.2 SYSTEM OVERVIEW

3.2.1Feature Selection

We propose using weighted keywords to represent individual documents. The weight of a keyword is determined by the tf-idf of the keyword. The tf-idf of a word is the product of the term frequency (tf) and the inverse document frequency (idf) of the word (Manning et al., 2008). For document I, let I_M be the modified document after pre-processing is done. Then we compute the weight for each remaining word, and sort the words in descending order in terms of their weights. For example, the following is document A after stop words and punctuation marks are deleted:

A_m: rally sidewalk legal arguments patient protection affordable care Supreme Court.

The ordering of the terms sorted by tf-idf is

supreme ≥ affordable ≥ protection ≥ patient ≥ legal ≥ sidewalk ≥ argument.

For document B, the ordering of the terms sorted by tf-idf is

Supreme ≥ affordable ≥ protection ≥ patient ≥ pavement ≥ legal ≥ argument.

The set of the top k words are selected as the feature of the underlying document. For example, let k be 4. The feature f_A obtained for document A contains four words, supreme, affordable, protection and patient, i.e.
$f_A = \langle supreme, affortable, protection, patient \rangle$

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

Note that the ordering of the words in f_A matters. The feature f_B obtained for document B also contains these four words:

$f_B = \langle supreme, affortable, protection, patient \rangle$

Therefore, documents A and B are identical from the viewpoint of features. This property is good for near-duplicate document detection. For a document, we represent it by a feature set which is the union of the features of that document. The procedure of finding the feature set for a document can be summarized below.

ALIGN (SRRs)

1. j=1,i=1
2. for all SRRs
3. group the data items of same concept using Similarity Measures
4. If for any two data items Sim>T
5. Group data items
6. Calculate group similarity using mean values to shift the data items of other concept.

Input: a document d in document set D
Output: the feature set fd for d
Procedure find_feature_set (d)
For each document d in D
Perform preprocessing on d
Calculate the weight for each remaining word
Sort the words in descending order
Let f contain the top k words
Add f into f_d
End Procedure

With this method, near-duplicate documents tend to have identical or very similar representations. Note that the number of words selected from a document may affect the performance of near-duplicate document detection. With fewer words, the representation of a document can be shorter in size and the computation time for later processing can be smaller. However, detection accuracy can be worse.

3.2.2 Similarity measures

Conventionally a predesignated threshold is required to determine whether two documents are near-duplicate to each other. Setting such a threshold manually is quite difficult and needs a lot of trial-and-error efforts. To alleviate this difficulty, we adopt a support vector machine (SVM) (Martins, 2011) to learn a classifier based on a set of training patterns. Existing technique based on sentence-level feature selection does not provide efficient similarity degree between two documents using similarity functions.

Similarity Function:

A similarity function is used to calculate the similarity degree between two documents. Let $f_1 = \{f_{1,1}, f_{1,2}, \cdots, f_{1,m}\}$ and $f_2 = \{f_{2,1}, f_{2,2}, \cdots, f_{2,m}\}$ be feature sets of documents d_1 and d_2 respectively. There are some similarity functions as follows:

- **Jaccard function:**

$$sim(d_1, d_2) \equiv J(d_1,d_2) = \frac{f_1 \cap f_2}{f_1 \cup f_2} \quad (1)$$

Where ∩ stands for the AND operator and U for the OR operator in set theory.

- **Cosine function**:

$$sim(d1, d2) \equiv C(d1, d2) = \frac{f_1 . f_2}{||f_1||.||f_2||} \quad (2)$$

Where $f_1 . f_2$ is defined to be $f_1 . f_2 = f_{1,1}f_{2,1} + f_{1,2}f_{2,2} + \cdots + f_{1,m}f_{2,m}$ and $||f_i|| = \sqrt{f_i . f_i}$ for i=1,2.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

- **Euclidean distance**:

$$sim(d1,d2) \equiv Ec(d1,d2) = \sqrt{(f_1 - f_2).(f_1 - f_2)} \quad (3)$$

Where $f_1 . f_2 = \{f_{1,1}f_{2,1} + f_{1,2}f_{2,2} + \cdots + f_{1,m}f_{2,m}\}$.

- **Extended Jaccard function:**

$$sim(d1,d2) \equiv EJ(d1,d2) = \frac{f_1 . f_2}{f_1.f_1 + f_2.f_2 - f_1.f_2} \quad (4)$$

- **Dice function:**

$$sim(d1,d2) \equiv D(d1,d2) = \frac{2f_1 . f_2}{f_1.f_1 + f_2.f_2} \quad (5)$$

Note that $EJ(d1,d2)$ is an extended version of $J(d_1,d_2)$ and $D(d_1,d_2)$ is a simplified version of $EJ(d1,d2)$.

Before going on, we define the similarity vector between two documents d1 and d2, denoted S_V(d1,d2), as follows.

Definition 1. The similarity vector between two documents d1 and d2 is defined as $S_v(d_1,d_2) = \langle X_1, X_2, \ldots X_r \rangle$ where x_i, $1 \le i \le r$, is the similarity degree computed for d_1 and d_2 by a certain similarity function.

The value of r and the similarity functions involved can be selected by the users. For example, we can let $X_1 = J(d_1,d_2)$ and $X_2 = Ec(d_1,d_2)$ for r = 2. A training pattern set is derived from a collection of training documents. Each training pattern in the training data set involves a pair of training documents d_1 and d_2 which are known in advance to be near-duplicate or non-near-duplicate to each other, and is expressed as (x, t) where.

- x = S_v(d1,d2) is the similarity vector between d1 and d2.
- t is the desired output, and t = +1 if d1 and d2 are near-duplicate and t = -1 if d_1 and d_2 are non-near-duplicate.

The process of preparing a training pattern set is summarized below.

```
Input: a set D of training documents
Output: the derived training pattern set X
Procedure derive_training_pattern_set (D)
For each pair of documents d1 and d2 in D
Calculate the similarity vector x = Sv(d1,d2)
If d1 and d2 are near-duplicate
Set t = +1
Else
Set t = -1
Add (x, t) into X
End Procedure
```

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

3.2.3 SVM

When the training pattern set is ready, SVM works on the set and finds a hyperplane g(x) = 0 which optimally separates the training patterns with t = +1 from those with t = -1. Let the resulting training pattern set contain N training patterns, denoted as X = $\{(x_j, t_j) | 1 \le j \le N\}$. We'd like to minimize

$$L_p = \frac{1}{2}||W||^2 + C\sum_j \xi^j \quad (6)$$

where w is the coefficient vector of the hyperplane to be optimized, C is the penalty factor, and $\xi_j \ge 0$, $1 \le j \le N$, are slack variables, subject to the constraints

$$t^j w^T \varphi(x^j) \ge 1 - \xi^j, 1 \le j \le N \quad (7)$$

where $\varphi(x) = (\varphi_1(x), \varphi_2(x), \dots \varphi_h(x))$ is a mapping from the r-dimensional x space to the h-dimensional z space

$$z = \varphi(x) \quad (8)$$

with $z_i = \varphi_i(x), 1 \le j \le N$. The hyperplane in the h-dimensional z space is g(z) = 0, and the desired discriminant function in the x space is

$$g(x) = w^T \varphi(x) = \sum_{i=1}^{h} w^i \varphi_i(x) \quad (9)$$

Eq. (6) can be converted to the dual

$$L_p = \sum_j \alpha^j - \frac{1}{2}\sum_j \sum_i \alpha^j \alpha^i t^j t^i k\langle x^j, x^i \rangle \quad (10)$$

Where $k\langle x^j, x^i \rangle \equiv \varphi\langle x^j \rangle^T \varphi\langle x^i \rangle$ is a kernel function, and a_j and a_i are Lagrange multipliers. Eq. **(10)** should be maximized with respect to α_j, subject to

$$\sum_j \alpha^j t^j = 0 \text{ and } 0 \le \alpha^j \le C, \quad 1 \le j \le N \quad (11)$$

3.2.4 Discriminant derivation

Eq. (10) can be solved using quadratic optimization methods (Martins, 2011). Then the discriminant g(x) is obtained by

$$g(x) = \sum_j \alpha^j K(x^j, x) \quad (12)$$

These will be used in the testing phase.

3.3 Testing phase

Once the discriminant g(x) is obtained, we can use it to determine if a document d_c is a near-duplicate to an input document d_I as follows. Firstly, the feature sets f_c and f_I of d_c and d_I, respectively, are obtained. Then we calculate the similarity vector $S_v(d_I, d_c)$. Let $x = S_v(d_I, d_c)$. Then we calculate g(x). If $g(x) \ge 0$, dc is determined to be a near-duplicate to d_I.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

3.4 System operation

Fig. 2 shows the flowchart of our proposed algorithm for near duplicate document detection. Fig. 2(a) shows the training phase of the algorithm. Given a set of training documents, we calculate the feature set for each document by procedure find_feature_set. Then we derive a training pattern set by procedure derive_training_pattern_set. Then we employ SVM to build a classifier based on the training pattern set. A discriminant function for separating the documents of class +1 and class _1 apart is derived. Fig. 2(b) shows the testing phase of the algorithm. Given an input document d_I, we want to retrieve all documents in D_c that are near-duplicates to d_I. We calculate the feature sets for d_I and each document d_c in D_c. We calculate the similarity vector between d_I and d_c. Then we feed the similarity vector to the discriminant function obtained in the training phase. If the returned value is equal to or greater than 0, dc is determined to be a near-duplicate to d_I. Otherwise, d_c is not a near-duplicate to d_I.

Figure 2 Flow Chart

IV. CONCLUSION

Digital documents are easily generated and put on the internet. Through the delete, copy, and paste commands provided by an editor or other tools, near-duplicate documents are likely to appear in various web communities, e.g., blogs and forums, increasing the volume of information one may have to go through and requiring more storage and bandwidth for communication. Detecting near duplicates is an important issue in the field of information retrieval. However, it is not an easy task. We have presented a novel method for detecting near-duplicates from a large collection of documents. Our method consists of three major components, feature selection, similarity measure, and discriminant derivation. For an input document, some pre-processing work is done on it. Then for each sentence, the heavily weighted terms are selected to be the feature of the sentence. As a result, the input document is turned into a set of features. Then the similarity degree between the input document and each document in the given collection is computed. Finally, a discriminant function is derived from a SVM-based classifier, which is then used to determine whether a document is a near-duplicate to the input document based on the similarity degree between them. Our method has several advantages. The sentence-level features we adopt can better reveal the characteristics of a

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 1, March 2014

Proceedings of International Conference On Global Innovations In Computing Technology (ICGICT'14)

Organized by

Department of CSE, JayShriram Group of Institutions, Tirupur, Tamilnadu, India on 6th & 7th March 2014

document, and learning a discriminant by SVM can avoid trial-and-error efforts required in conventional methods. A variety of experiments have shown that our method is effective in near-duplicate document detection.

V. REFERENCES

[1] Wang, J. -H. & Chang, H. -C. (2009). Exploiting sentence-level features for nearduplicate document detection. In: Proceedings of the 5th Asia information symposium on information retrieval technology (pp. 205–217).

[2] Hajishirzi, H., Yih, W., & Kolcz, A. (2010). Adaptive near-duplicate detection via similarity learning. In: Proceedings of the 33rd international ACM SIGIR conference on research and development in information retrieval (pp. 419–426).

[3] Zhao, Z., Wang, L., Liu, H., & Ye, J. (2011). On similarity preserving feature selection. IEEE Transactions on Knowledge and Data Engineering. 10.1109/TKDE.2011.22.

[4] Narayana, V. A., Premchand, P., & Govardhan, A. (2009). A novel and efficient approach for near duplicate page detection in web crawling. In: Proceedings of IEEE international advance computing conference (pp. 1492–1496).

[5] Arnosti, N. A., & Kalita, J. K. (2011). Cutting plane training for linear support vector machines. IEEE Transactions on Knowledge and Data Engineering. 10.1109/TKDE.2011.24.

[6] Yang, H. & Callan, J. (2006). Near-duplicate detection by instance-level constrained clustering. In: Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval (pp. 421–428).

[7] Broder, A. Z. (2000). Identifying and filtering near-duplicate documents. In:Proceedings of the 11th annual symposium on combinatorial pattern matching(pp. 1–10).

[8] Theobald, M., Siddharth, J., & Paepcke, A. (2008). Spotsigs: Robust and efficient nearduplicate detection in large web collections. In: Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval (pp. 563–570).

[9] Sood, S. & Loguinov, D. (2011). Probabilistic near-duplicate detection using simhash. In: Proceedings of the 20th ACM international conference on information and Knowledge management (pp. 1117–1126).