

Length Normalization in Degraded Text Collections

Amit Singhal*
Gerard Salton
Chris Buckley

Department of Computer Science
Cornell University, Ithaca, NY 14853
{singhal, gs, chrisb}@cs.cornell.edu

Abstract

Optical character recognition (OCR) is the most commonly used technique to convert printed material into electronic form. Using OCR, large repositories of machine readable text can be created in a short time. An information retrieval system can then be used to search through large information bases thus created. Many information retrieval systems use sophisticated term weighting functions to improve the effectiveness of a search. Term weighting schemes can be highly sensitive to the errors in the input text, introduced by the OCR process. This study examines the effects of the well known *cosine normalization* method in the presence of OCR errors, and proposes a new, more robust, normalization method. Experiments show that the new scheme is less sensitive to OCR errors and facilitates use of more diverse basic weighting schemes. When used in a correct text collection, the new normalization scheme yields significant improvements in retrieval effectiveness over cosine normalization.

1 Background

OCR is widely used to scan printed text, text not otherwise available in machine readable form, and convert it into electronic form. Advances in OCR technology have decreased the error rates of OCR devices, but due to the poor quality of some printed material and other deteriorations introduced by the OCR system, the error rate of an OCR process can still be substantial.[6] This poses a special challenge when scanned text collections are searched by an automatic information retrieval

*This study was supported in part by the National Science Foundation under grant IRI-9300124.

system. OCR errors can corrupt the document view formed by an information retrieval system, and can substantially hinder the successful retrieval of good documents for user-queries. Most modern information retrieval systems use sophisticated term weighting functions to assign importance to the individual words of a document, for document-content identification.[7, 12, 11, 10] These weighting functions use the occurrence statistics of words in the documents to assign importance to different words. As the occurrence statistics of words can change substantially due to OCR errors, weighting schemes are specially susceptible to degradation in the quality of the input text.

How does the OCR process affect the *retrieval effectiveness* of an information retrieval system? This problem has been extensively studied by Taghva, *et. al.*, at the Information Science Research Institution (ISRI) of University of Nevada, Las Vegas. In the numerous studies performed at ISRI [3, 15, 16, 17, 18, 19], Taghva and his colleagues have noticed that in terms of recall-precision measures, commonly used to evaluate information retrieval systems, the retrieval effectiveness of a retrieval system is not substantially affected by OCR errors in the input text. The main reason for this, they claim, is the excessive redundancy of information present in the full text. This redundancy compensates for the information lost in an OCR operation. They finally conclude that, in conjunction with an automatic information retrieval system, a degraded text collection produced by a reasonable OCR system, can be used almost as effectively as the correct text collection.

However, Taghva and his colleagues did notice several interesting facts when sophisticated term weighting schemes were used in conjunction with degraded text. Specifically, whenever they used the well known cosine normalization function for length normalization of term weights, the document ranking generated by a system for the degraded text collection was substantially different from the ranking generated for the corrected collection. In [15] they explain (see Section 2 for an overview of the term weighting jargon used in this explanation):

...from our variance test, cosine normalization is a significant factor. We know that “garbage strings” increase the length of the OCR document vectors. This is specially true for documents with many misrecognized terms and/or graphic text. We found that the OCR vector length was approximately three times the length of the corresponding corrected-text document vector for those documents with a high discrepancy in ranking. Aggravating this fact, “garbage strings” tend to have high term weights. When these terms are normalized, their significance to the document increases in relationship to other terms in the collection.

Due to OCR errors, numerous words are misrecognized and meaningless “garbage strings” are

generated. Term weighting schemes are affected by the presence of such garbage strings, and assign misleading weights to different terms in a document. It is evident from their study that due to the wide corruption of the unnormalized term weights (specially the inverse document frequency factor), cosine normalization is not desirable in degraded text collections.

Research in information retrieval has shown that document length normalization is important for effective retrieval of documents of all sizes.[11] Therefore, an alternative to cosine normalization is needed in OCR environments. This study focuses on developing a new, robust, length normalization technique that is less sensitive to OCR errors. Not only is the new normalization scheme more robust in presence of OCR errors, it also yields significant improvements when used in place of cosine normalization in correct text collections.

Section 2 briefly introduces term weighting. Section 3 describes the Smart system, which was used to perform the experiments for this study. Section 4 introduces a new length normalization technique. Section 5 discusses the document collection used in this study. Section 6 describes the experiments. Section 7 discusses the results of this study. Section 8 studies the effect of the new normalization method on correct text collections. Section 9 concludes the study with key observations.

2 Term Weighting

Since different terms have different importance in a text, an importance indicator – the *term weight* – is associated with every term.[14, 11] Higher weights are assigned to more important terms. Term weighting is one of the most important parts of a retrieval system. Judicious assignment of term weights can substantially improve the search effectiveness of a system. Several term weighting schemes have been proposed over years. Most of these schemes use the following factors to assign weight to a document term:

- the term's frequency in the document,
- prevalence of the term in the entire collection, and
- the length of the document.

Typically a term that occurs frequently in a text is more important in the text than an infrequent term. Therefore, the number of occurrences of a term, often called the *term frequency* or *tf*, in a text is used as the term's weight. Common words tend to occur in numerous documents in a collection, and are poor indicators of a document's content. The more documents a term occurs in, the less important it may be. Therefore, a term's weight should be inversely proportional to the number of documents that the term occurs in, in the entire collection, called the collection frequency (or the *document frequency*) of the term. An *inverse document frequency*, or *idf*, factor is commonly used in term weights to incorporate this effect.

Long and verbose documents usually use the same terms repeatedly. As a result, the term frequency factors may be large for long documents. Long documents also have numerous different terms. This increases the number of word matches between a query and a long document, increasing its chances of retrieval over shorter documents. To compensate for this effect, *normalization* of term weights is often used. Normalization is a way of imposing some penalty on the term weights of the longer documents. *Cosine normalization* is an effective normalization technique. Every term weight in a document is divided by the Euclidean length of the $tf \times idf$ weighted document vector, $\sqrt{w_1^2 + w_2^2 + \dots + w_n^2}$, where w_i is the $tf \times idf$ weight of the i -th term in the document. This gives us the final weight for a term as:

$$\frac{tf \times idf \text{ weight}}{\text{Euclidean length of the document vector}}$$

Terms absent from a document are considered to have a zero weight.

3 The Smart System

The Smart system [9] is a sophisticated text processing system based on the vector space model [13], developed over the last thirty five years. In the vector space model, every information item – including the stored texts and any natural language information request – is stored as a set, or vector, of terms. Smart automatically assigns weights to the terms in a vector and uses a vector *inner-product* function to determine the vocabulary overlap between any two text vectors. A natural language query entered by the user is converted into a weighted term vector, and using the inner-product function, a numeric similarity is computed between the query vector and the

Term Frequency		Inverse Document Frequency		Normalization	
First Letter	$f(tf)$	Second Letter	$f(\frac{1}{df})$	Third Letter	$f(length)$
n (natural)	tf	n (no)	1	n (no)	1
l (logarithmic)	$1+\log(tf)$	t (full)	$\log(\frac{N}{df})$	c (cosine)	$\sqrt{w_1^2 + w_2^2 + \dots + w_n^2}$
a (augmented)	$0.5 + 0.5 \times \frac{tf}{\max tf}$				

Table 1: Term Weights in the Smart System

vector for every document in the collection. The documents are presented to the user in decreasing order of their similarity to the user-query.

In the Smart system, term weighting schemes are denoted by triples of letters. The first letter in a triple is a shorthand for the function of the term frequency factor being used in the term weights, the second letter corresponds to the inverse document frequency function and the third letter corresponds to the normalization factor applied to the term weights. Table 1 summarizes the most commonly used functions in term weighting. The $\max tf$ in the augmented tf weight formula is the maximum term frequency of any term in the document under consideration. The N in the full inverse document frequency formula corresponds to the total number of documents in the collection.

A retrieval experiment can now be characterized by a pair of triples – $ddd.qqq$ — where the first triple corresponds to the term weighting used for the documents in a collection, and the second triple corresponds to the query term weights. For example, when $anc.ltn$ is used to symbolize a retrieval run, the document term weights use in the run are,

$$\frac{0.5 + 0.5 \times \frac{tf}{\max_{tf}}}{\sqrt{(0.5 + 0.5 \times \frac{tf}{\max_{tf}})^2}}$$

and the query term weights are,

$$(1 + \log tf) \times \log\left(\frac{N}{df}\right)$$

We will use this term weight triple notation to characterize all our experiments in this study.

4 New Normalization Technique

4.1 Problems with Cosine Normalization

Cosine normalization introduces a mutual dependence between the weights of the terms in a document. Due to the presence of all the unnormalized document term weights in the normalization factor, when cosine normalization is used for document length normalization, the weight of a document term depends on the individual weights of all other terms in the document. An error in approximation of one term's weight can affect the weights of all other terms. This dependence is undesirable in environments where individual weights of the terms might be erroneous and do not accurately represent the actual importance of the terms in content representation of the documents. OCR environments are specially susceptible to such errors in term weight estimation.

Due to OCR errors, one word in the original text might get split into two or more words in the scanned text. The OCR system can even misinterpret some graphic material as one or more completely new words. This can increase the length of the document vectors, and, in turn, can increase the value of the cosine normalization factor. This increase in the cosine normalization factor artificially depresses the weights of the correctly recognized terms of a document, making the retrieval of the document harder against a user-query.

This skewing of weights for correctly recognized words is worse when the *idf* factor is used in term weights. OCR errors can create numerous garbage strings (misrecognized words), that occur in very few documents in the entire collection. Due to their infrequent occurrence in the collection, such erroneous strings tend to have a very high inverse document frequency value and, therefore, a higher unnormalized $tf \times idf$ weight. High individual term weights for such misrecognized terms can significantly increase the cosine normalization factor, thereby corrupting the weights for all other terms in the document.

For example, Table 2 shows the text for a short TREC [4] document – DOE1-01-0942, the correct *ltc* weighted document vector, and the document vector if the word `system` is misrecognized as `system`. Assuming that this recognition error occurs only once, the corrupted term – `system` – gets an unnormalized $tf \times idf$ weight of 13.5174 ($tf = 1$, $N = 742,202$, $df = 1$, $idf = \log(\frac{N}{df}) = 13.5175$), as opposed to its initial unnormalized weight of 1.3446 (the correct *df* for the word `system` is

Text: A specific solid system with first order phase transition in its surface is analysed. To accomplish this task, we use the Landau theory. (M.W.O.).		
	Correct Normalization Factor: 15.6092	Corrupted Normalization Factor: 22.0338
Term	Correct <i>ltc</i> weights	Corrupted <i>ltc</i> weights
surface	0.20609	0.14600
accomplish	0.26372	0.18682
task	0.22716	0.16092
solid	0.22955	0.16261
m.w.	0.58059	0.41129
landau	0.46977	0.33271
order	0.13781	0.09761
transition	0.23660	0.16757
theory	0.24174	0.17121
specific	0.16075	0.11385
analysed	0.11849	0.08392
phase	0.21261	0.15058
system/ system	0.08614	0.61349

Table 2: The text, correct *ltc* weighted vector and the corrupted *ltc* weighted vector for TREC document DOE1-01-0942. The OCR process misrecognized the word `system` as `system`.

193,451). This single corruption in the source text elevates the cosine normalization factor from 15.6092 to 22.0338, an increase of more than 40%, and unfairly depresses the weight of every other term in the document. For example, the weight of the useful term `Landau` has been reduced from 0.47 to 0.33. Such deviations can be much more severe in documents with multiple misrecognized words since these words can skew the weight estimation for good terms to a greater extent.

4.2 Using Byte Size in Normalization

To remove this undesirable effect of cosine normalization in the presence of OCR errors, we would like to replace it by some other normalization scheme that does not introduce any mutual dependence between the term weights in a document. For this, normalization functions that do not use individual term weights are required. The aim of term weight normalization is to reduce, if not remove, the unfair advantage that the longer documents have in retrieval over the shorter documents. Since longer documents have more words and thus a greater number of bytes, functions of the number of bytes in a document could possibly be used for length normalization of the document term weights. Some researchers have successfully used the document size (in bytes) for length normalization.[8] In this study, we focus on using a function of the document size in bytes (which

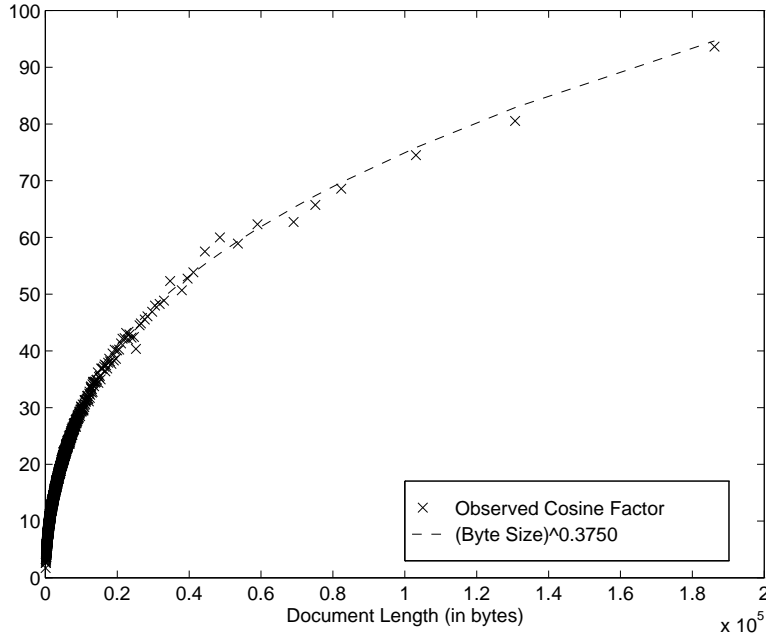


Figure 1: Learning function of byte size that can be used for term weight normalization. 'X' denotes the observed value for cosine normalization factor for lnn weighted document vectors. The dashed line corresponds to the function $byte\ size^{0.3750}$.

we call the *byte size* of the document) for length normalization in term weighting.

Using a function of byte size for normalization, in place of cosine normalization, would remove the mutual dependence of term weights, introduced by cosine normalization. In OCR environments, the byte sizes of the documents are less distorted, and this distortion is much more uniform across documents, than the corruption in the cosine normalization factors. Using a function of byte size should be more stable than using cosine normalization and should, in principle, yield less deviant retrieval results.

As we have used the TREC collection [4, 5] in this study, to learn what function of byte size can be used for term weight normalization, we sampled every twentieth document from TREC disks one and two. This gave us a varied sample of several documents from all the different sub-collections of TREC. Our previous experiments with the TREC collection have shown that using *lnc* weights on document terms yields good results.[2] To replace the cosine normalization factor and still perform length normalization comparable to cosine normalization, we need to learn the relationship of the cosine normalization factor with the byte sizes of the documents. To do this, we computed the

cosine normalization factor used in *lnc* weighting of the sampled set of documents and plotted it against the byte sizes of the documents. The cosine normalization factor for a document with t terms is given by:

$$\sqrt{(1 + \log(tf_1))^2 + (1 + \log(tf_2))^2 + \dots + (1 + \log(tf_t))^2}$$

To obtain a smoother plot, we sorted the list of $\langle \textit{byte size}, \textit{cosine normalization factor} \rangle$ pairs by byte size and divided this sorted list into bins of ten pairs each. We selected the median byte size and the median cosine normalization factor from every bin to represent that bin on a graph. Figure 1 is a plot of the median cosine normalization factors against the median byte length for every bin. The variation in the cosine factor with the byte length suggests that a fractional power of the byte size ($\textit{byte size}^\alpha$ where $\alpha < 1$) can be used to replace the cosine normalization factor. Figure 1 shows that a value of 0.3750 for α yields a function very similar to the actual cosine normalization factor. Therefore we used $\textit{byte size}^{0.3750}$ for term weight normalization in our experiments. We represent this new byte size normalization scheme by the letter **b** in the weighting triple notation of *Smart*.

4.3 Performance on Correct Text

To test the effectiveness of this new byte size normalization scheme in general retrieval environments, we evaluated it on the TREC collection. Table 3 shows the average recall–precision analysis for the *lnc.ltc* run, as compared to the *lnc.ltc* run. Not only does the byte size normalization significantly outperform the cosine normalization in terms of average precision, it is substantially superior to cosine normalization at every single recall point. These results suggest that mutual independence of term weights might also be desirable in correct text collections, but we need to investigate this phenomenon further before making any more claims about the effectiveness of byte size normalization on correct text collections.

Run	lnc.ltc	lnb.ltc
# Queries	200	200
Total number of documents over all queries		
Retrieved	200000	200000
Relevant	56360	56360
Rel. Ret.	34992	37054
Interpolated Recall - Precision Averages		
at 0.00	0.7840	0.8578
at 0.10	0.5739	0.6279
at 0.20	0.5068	0.5499
at 0.30	0.4416	0.4855
at 0.40	0.3817	0.4205
at 0.50	0.3141	0.3538
at 0.60	0.2356	0.2725
at 0.70	0.1671	0.1970
at 0.80	0.0834	0.1073
at 0.90	0.0239	0.0326
at 1.00	0.0000	0.0000
Avg. precision (non-interpolated)		
	0.3004	0.3378 (+12.5%)

Table 3: Average recall-precision analysis for byte size normalization to cosine normalization, using TREC queries 1-200 and documents from disk 1 and 2.

5 The Collection

Information retrieval experiments performed on very small text collections do not justly reflect a retrieval system’s performance on a larger, more diverse text collection. For this reason, we wanted to use a large text collection for our experiments. In absence of a large collection of scanned text, we used simulated OCR degradation on disks one and two the TREC collection, a total of 742,202 documents.[4, 5] We used a simplified text degradation routine which randomly altered every character in the text with a given probability. This simple simulation of OCR errors depicts the extreme corruption that an OCR process can cause in the input text. In reality, OCR errors tend to be repetitive and the erroneous characters are not independent of the input text sequence. We believe that this difference from actual OCR environment is not significant for this set of experiments.

We simulated a poor quality OCR process where every character was randomly altered with a probability 0.05 (5% degradation). As the collection size is around 2 GB, and 5% of the characters are randomly changed, around 100,000,000 characters would be changed by the degradation pro-

TREC Disks 1 and 2 – 742,202 Documents		
	Correct Text	Simulated OCR (5% degradation)
# Unique Terms in Collection	511,324	8,423,852
Avg. # of <i>Unique</i> Terms per Document	108.51	168.87
Avg. # of Terms per Document	210.52	239.49

Table 4: Characteristics of the collections obtained at different degradation levels compare to the original TREC collection.

cess. Since many random changes would produce a completely new sequence of letters, we expect the dictionary for the degraded collection to be much larger than the original dictionary. Some characteristics of the degraded collection and the correct collection are listed in Table 4. As expected, there is a large increase in the number of unique terms in the degraded collection. Since the random degradation might corrupt one occurrence of a term and not change another occurrence of the same term, one unique term can generate several difference unique terms in a degraded document. Due to this effect, we notice an increase in the average number of unique terms per document. Also, due to possible insertion of random word terminators (space, punctuation, . . .), in the middle of a word, a word in the correct text might split into two or more words in the degraded text. This effect is reflected by an increase in the average number of (total) terms in a document.

Surprisingly, Croft, Taghva, *et. al.*, observed a reduction in the dictionary size in their experiments with simulated OCR degradation.[3] They degraded four different collections (CACM, NPL, WEST and WSJ) and the resulting dictionary size for their degraded collections was always smaller than the original dictionary size. This is a result that we will need to reconcile. In that study, Croft, Taghva and their colleagues also reported a decrease in the total term occurrences (and thus the average number of terms in a document) in all the degraded collections. This, again, does not agree with our observations. The characteristics of our degraded collections do, in fact, agree with the actual OCR collections used in earlier studies.[17] In experiments conducted with small real OCR collections, Taghva and others, like us, did notice a huge increase (four fold) in the dictionary size for the degraded text collection.

6 Experiments

Changes in the average precision, computed across a set of queries, do not reflect the effects of degradation in a retrieval experiment. Random variations in ranks of the retrieved documents might lower the average precision for one query, whereas, for another query, similar random variations might increase the average precision. Even though both random variations are undesired in a retrieval system, when averages are computed across queries, the final change in average precision might be lower than the change in average precision for individual queries. Also, it is possible that the average precision for a query, in retrieval from the correct text collection and the degraded text collection, has very little variation, but there is a large variation in the ranks of individual documents in the two retrieval runs. Therefore, the *variation between the document rankings produced for the correct text and the degraded text*, is a more meaningful measure to study the effects of OCR errors on a retrieval system.[15] To compare the effects of the cosine and the byte size normalization on retrieval from the degraded text, we studied this variation in the document rankings produced by the different normalization methods (none, cosine and byte size).

As a first test, we computed the average number of common documents, retrieved per query within the top one thousand, from the correct collection and the degraded collection by the different normalization (weighting) schemes. A higher average number of common documents would signify a higher correlation between retrievals from the correct and the degraded collection. A scheme that produces a higher number of common documents is less susceptible to OCR errors during retrieval. To study the variations in the ranks of individual documents, we retrieved the top one thousand documents for every query from the correct collection and from the degraded collection, using a particular weighting method. We then computed the difference in the ranks assigned to the *same document* in the two retrieval runs. If a document was retrieved in one run and not in the other, its rank was assumed to be 1001 in the run in which it was not retrieved within the top one thousand. From this analysis, we obtained a list of rank differences ($|rank_1 - rank_2|$) for the documents retrieved from the correct and the degraded collection by a weighting scheme. We computed the mean and the standard deviation for this list to estimate the variation in rankings generated by the weighting scheme for the two collections.

A low mean would suggest that the variation in the rankings generated by a weighting scheme,

for the correct text and for the degraded text, is low. A low variation in rankings would, in turn, suggest that the degradation of text did not substantially affect the retrieval process in presence of that particular weighting scheme. Therefore, if a weighting scheme generates a low mean variation in document ranks, it is less susceptible to errors in the input text and is a good scheme in presence of OCR errors. Similarly, a low standard deviation would signify that few documents had very wide difference in the ranks assigned to them in the two retrieval runs, and is also desirable from a weighting scheme.

Our experiments with the TREC corpus have shown that the logarithmic term frequency weights provide superior retrieval effectiveness as compare to other tf weights. We therefore used the logarithmic tf factor in these experiments. Our previous experiments have also shown that the use of the idf factor on the document weights tends to hurt retrieval effectiveness.[2, 1] This can possibly be attributed to the presence of cosine normalization – if a correct document has one or more very rare terms, terms with very high idf values, similar to the misspelled term in Table 2, these terms unfairly depress the weights of the other document terms. To test this hypothesis, we ran experiments with both the presence, and the absence of the idf factor in document term weights.

We experimented with the absence of any normalization, the use of cosine normalization and the use of byte size normalization on document term weights. Since normalization of query term weights just acts as a scaling factor for all the query–document similarities, and has no effect on the relative ranking of the documents, there was no need to vary the normalization factor for the query term weights. Also, since the idf factor should be used at least once in the similarity computation, we always used the idf factor on the query term weights. With the above combination of the document and query term weights, we ran six different runs on the correct and the degraded collection – *ltn.ltc*, *ltn.ltc*, *lnc.ltc*, *lnc.ltc*, *lmb.ltc*, and *lmb.ltc*. All experiments were done with fifty TREC queries, numbered 151 to 200.

7 Results

The results obtained are listed in Table 5. These results, yet again, confirm the long known need for length normalization of document term weights. The two runs in which no normalization

Run		Correct Text	OCR (5% degradation)			
		Avg. Precision	Avg. Precision	Avg. Common Docs./Query	Rank Difference	
					Mean	Standard Deviation
1	ltn.ltc	0.0090	0.0068 (-24.9%)	936.30	58.90	64.12
2	ltn.ltc	0.0413	0.0287 (-30.5%)	910.12	83.54	92.17
3	lnc.ltc	0.2730	0.2358 (-13.6%)	726.78	217.83	196.60
4	lnc.ltc	0.2245	0.2046 (-8.9%)	700.56	236.59	208.84
5	lnc.ltc	0.3047	0.2660 (-12.7%)	769.76	189.94	174.34
6	lnc.ltc	0.3071	0.2736 (-10.9%)	778.36	183.06	172.63

Table 5: Results of various runs on correct and degraded text.

was used (document weights *ltn* and *lnc*, rows 1 and 2) are extremely poor in comparison to the other four runs in which some type of normalization was used. The variation in the document rankings produced for the correct text and the degraded text, for these two runs, even though quite low, is not indicative of a realistic retrieval situation. In absence of any length normalization, the poor retrieval effectiveness of these runs is due to the excessive bias in favor of retrieval of the long documents. Since the TREC collection has few very long documents, when two different retrievals favor the long documents, they are bound to retrieve a large number of common documents.

Table 5 shows that, with a reasonable (normalized) weighting scheme, the deterioration in retrieval effectiveness due to OCR degradation is in the range of 9–14% (rows 3 to 6). These numbers are somewhat larger than the deterioration reported by Croft, Taghva, *et. al.*, in earlier studies (1–10%).[3] The difference in their results and the results from this study can be an artifact of the different collections that are being used in the two studies. Also, the text degradation schemes used in the two studies are substantially different. Overall, these results confirm the findings of Taghva and his colleagues that it is possible to use an automatic information retrieval system in conjunction with degraded text collections, without much loss in average recall-precision figures.

When no idf factor is used in the document term weights, and cosine normalization is replaced by byte size normalization, the number of common documents retrieved in the top one thousand, from the correct collection and the degraded collection, increases from 726.78 per query to 769.76 per query (rows 3 and 5). This increase is substantially higher – 700.56 per query to 778.36 per query – when the idf factor is used on the document term weights (rows 4 and 6). This signifies that when byte size normalization is used, more documents that were retrieved from the correct collection are being retrieved from the degraded collection. It is desirable to have little difference

in the retrieval characteristics of the correct and the degraded text. Using byte size normalization helps in reducing this difference. These numbers confirm our belief that cosine normalization is not desirable in the presence of OCR errors, even more so when the idf factor is used in the document term weights.

We also observe a decrease in the mean variation in the document ranks between the documents retrieved from the two collections. When no idf factor is used in the document term weights, switching from cosine normalization to byte size normalization reduces the mean variation in the ranks of the retrieved documents from 217.83 to 189.94 (rows 3 and 5). This decrease is much more noticeable when the idf factor is used in the document term weights – down from 236.59 to 183.06 (rows 4 and 6). Once again, we notice that removing the cosine normalization factor in presence of the idf factor, is more important than removing it when no idf factor is being used on the document term weights. It is also useful to remove cosine normalization when no idf factor is used, but the need to remove it is much less pressing.

From rows 3 and 4 of Table 5, we can notice that when cosine normalization is used, the presence of idf increased the standard deviation in the rank variation from 196.60 (*lnc.ltc*) to 208.84 (*ltc.ltc*). Whereas, rows 5 and 6 show that with byte size normalization, the standard deviation actually goes down, slightly, from 174.34 (*lnb.ltc*) to 172.63 (*ltb.ltc*). This suggests that using cosine normalization in the presence of the idf factor also aggravated the variations in the rankings produced in the two collections. As the standard deviation in rank variation is much smaller when byte size normalization is used (around 170 as opposed to around 200, rows 3 and 4 vs. rows 5 and 6), this measure also suggests that using byte size normalization produces a lower variation in document ranks than using the cosine normalization factor.

8 Effects on Correct Text

When comparing byte size normalization with cosine normalization in Table 3, we observed that byte size normalization performs significantly better than cosine normalization. This observation is highlighted in Table 6. Byte size normalization performs 11.61% better than cosine normalization (0.3047 for *lnb.ltc* as opposed to 0.2730 for *lnc.ltc*). This would suggest that removing mutual dependence between document term weights might be inherently useful in information retrieval.

	Cosine Normalization		Byte Size Normalization	
	No idf	With idf	No idf	With idf
Avg. Precision	0.2730	0.2245 (-17.8%)	0.3047 (+11.61%)	0.3071 (+0.8%)
Number of Rel. Docs. Retrieved	6258	6002 (-256)	6294	6651 (+357)

Table 6: Effect of using the idf factor in the document term weight with cosine and byte size normalization. The figures are for TREC queries 151–200 and documents from disks 1 and 2. Top 1000 documents were retrieved for every query. The 11.61% improvement for byte size normalization with no idf is over cosine normalization with no idf.

This improvement in performance should be studied further to determine the precise effects of replacing cosine normalization with byte size normalization in correct collections.

An interesting effect of using byte size normalization in place of cosine normalization is that when cosine normalization is used, the introduction of the idf factor in the document term weights produces significantly worse results than using no idf. Table 6 shows that when cosine normalization is used, using the idf factor in the document term weights has significantly worse average precision than not using it (-17.8%). Also, when the idf factor is used with cosine normalization, the system retrieves 256 fewer relevant documents across all the fifty queries. Whereas, when byte size normalization is used, introducing idf in document term weights not only has a slightly better average precision (+0.8%), it also retrieved an extra 357 relevant documents in the top one thousand documents retrieved per query across the fifty queries.

These results support our hypothesis that when the idf factor is used in document term weights with cosine normalization, very rare terms, terms with very high idf value, unfairly suppress the weights of other document terms, even in correct text collections. These results also show that the use of byte length normalization in place of cosine normalization opens up the possibility of using the idf factor in document term weights. In environments where the use of the idf factor on documents term weights is important, using cosine normalization can hurt retrieval effectiveness. In such collections, byte size normalization has an added advantage over cosine normalization.

9 Conclusions

As a result of misrecognized words, an OCR system introduces meaningless garbage strings in a text collection. These garbage strings widely corrupt the cosine normalization factor, commonly used in term weighting. This corruption prohibits the use of cosine normalization in OCR environments. Byte size normalization is an attractive alternative due to its robustness to OCR errors. When used for the corrupted text, byte size normalization scheme produces results more similar to the results from the correct text collection. This new normalization scheme also shows promise for correct text databases when compared to cosine normalization. Byte size normalization produces significantly better retrieval results than cosine normalization on correct text. It also facilitates the use of the inverse document frequency factor in term weighting. Using the idf factor on document term weights, in presence of byte size normalization, retrieves more relevant documents and mildly improves the average precision. This is an added advantage of using byte size normalization in place of cosine normalization in correct text collections.

References

- [1] Chris Buckley, James Allan, and Gerard Salton. Automatic routing and ad-hoc retrieval using SMART : TREC 2. In D. K. Harman, editor, *Proceedings of the Second Text REtrieval Conference (TREC-2)*, pages 45–56. NIST Special Publication 500-215, March 1994.
- [2] Chris Buckley, Gerard Salton, and James Allan. Automatic retrieval with locality information using SMART. In D. K. Harman, editor, *Proceedings of the First Text REtrieval Conference (TREC-1)*, pages 59–72. NIST Special Publication 500-207, March 1993.
- [3] W. B. Croft, S. Harding, K. Taghva, and J. Borsack. An evaluation of information retrieval accuracy with simulated OCR output. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval*, pages 115–126, Las Vegas, NV, April 1994.
- [4] D. K. Harman. Overview of the first Text REtrieval Conference (TREC-1) . In D. K. Harman, editor, *Proceedings of the First Text REtrieval Conference (TREC-1)*, pages 1–20. NIST Special Publication 500-207, March 1993.

- [5] D. K. Harman. Overview of the second Text REtrieval Conference (TREC-2). In D. K. Harman, editor, *Proceedings of the Second Text REtrieval Conference (TREC-2)*, pages 1–20. NIST Special Publication 500-215, March 1994.
- [6] Thomas A. Nartker and Stephen V. Rice. OCR accuracy: UNLV’s third annual test. *INFORM*, 8(8):30–36, September 1994.
- [7] S.E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, May-June 1976.
- [8] S.E. Robertson and S. Walker. Some simple effective approximations to the 2–poisson model for probabilistic weighted retrieval. In W. Bruce Croft and C.J. van Rijsbergen, editors, *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 232–241, New York, July 1994. Springer-Verlag.
- [9] Gerard Salton, editor. *The SMART Retrieval System—Experiments in Automatic Document Retrieval*, Englewood Cliffs, NJ, 1971. Prentice Hall Inc.
- [10] Gerard Salton. *Automatic text processing—the transformation, analysis and retrieval of information by computer*. Addison-Wesley Publishing Co., Reading, MA, 1989.
- [11] Gerard Salton and Chris Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.
- [12] Gerard Salton and M.J. McGill. *Introduction to Modern Information Retrieval*. McGraw Hill Book Co., New York, 1983.
- [13] Gerard Salton, A. Wong, and C.S. Yang. A vector space model for information retrieval. *Journal of the American Society for Information Science*, 18(11):613–620, November 1975.
- [14] Gerard Salton, C.S. Yang, and C.T. Yu. A theory of term importance in automatic text analysis. *Journal of the American Society for Information Science*, 26(1):33–44, January–February 1975.
- [15] Kazem Taghva, Julie Borsack, and Allen Condit. Effects of OCR errors on ranking and feedback using the vector space model. Technical Report 94-06, Information Science Research Institute, University of Nevada, Las Vegas, August 1994.

- [16] Kazem Taghva, Julie Borsack, and Allen Condit. Evaluation of model-based retrieval effectiveness with OCR text. Technical Report 94-09, Information Science Research Institute, University of Nevada, Las Vegas, October 1994.
- [17] Kazem Taghva, Julie Borsack, and Allen Condit. Results of applying probabilistic IR to OCR text. In *Proceedings of the Seventeenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 202–211, Dublin, Ireland, July 1994.
- [18] Kazem Taghva, Julie Borsack, Allen Condit, and Srinivas Erva. The effects of noisy data on text retrieval. *Journal of the American Society for Information Science*, 45(1):50–58, January 1994.
- [19] Kazem Taghva, Julie Borsack, Allen Condit, and Padma Inaparthi. Querying short OCR'd documents. Technical Report 94-10, Information Science Research Institute, University of Nevada, Las Vegas, February 1995.