Linear Text Segmentation using a Dynamic Programming Algorithm

Athanasios Kehagias
Dept. of Math., Phys.
and Comp. Sciences
Aristotle Univ of Thessaloniki
GREECE
kehagias@egnatia.ee.auth.gr

Fragkou Pavlina , Vassilios Petridis
Dept. of Elect. and Computer Eng.
Aristotle Univ of Thessaloniki
GREECE
fragou@egnatia.ee.auth.gr,
petridis@eng.auth.gr

Abstract

In this paper we introduce a dynamic programming algorithm to perform linear text segmentation by global minimization of a segmentation cost function which consists of: (a) within-segment word similarity and (b) prior information about segment length. The evaluation of the segmentation accuracy of the algorithm on Choi's text collection showed that the algorithm achieves the best segmentation accuracy so far reported in the literature.

Keywords: Text Segmentation, Document Retrieval, Information Retrieval, Machine Learning.

1 Introduction

Text segmentation is an important problem in information retrieval. Its goal is the division of a text into *homogeneous* ("*lexically coherent*") segments, i.e segments exhibiting the following properties: (a) each segment deals with a particular subject and (b) contiguous segments deal with different subjects. Those segments can be retrieved from a large database of unformatted (or loosely formatted) text as being relevant to a query.

This paper presents a dynamic programming algorithm which performs *linear* segmentation [1] by global minimization of a *segmentation cost*. The *segmentation cost* is defined by a function consisting of two factors: (a) *within-segment word similarity* and (b) *prior information about segment length*. Our algorithm has the advantage of being able to be applied to either large texts - to segment them into their constituent parts (e.g. to segment an article into sections) - or to a stream of independent, concatenated texts (e.g. to segment a transcript of news into separate stories).

For the calculation of the segment *homogeneity* (or alternatively *heterogeneity*) of a text, several segmentation algorithms using a variety of criteria have been proposed in the literature. Some of those use linguistic criteria such as cue phrases, punctuation marks, prosodic features, reference, syntax and lexical attraction (Beeferman et al., 1997; Hirschberg and Litman, 1993; Passoneau and Litman, 1993). Others, following Halliday and Hasan's theory (Halliday and Hasan, 1976), utilize statistical similarity measures such as word cooccurrence. For example the linear discourse segmentation algorithm proposed by Morris and Hirst (Morris and Hirst, 1991) is based on *lexical cohesion relations* determined by use of Roget's thesaurus (Roget, 1977). In the same direction Kozima's algorithm (Kozima, 1993; Kozima and Furugori, 1993) computes the semantic similarity between words using a semantic network constructed from a subset of the Longman Dictionary of Contemporary English. Local minima of the similarity scores correspond to the positions of topic boundaries in the text.

Youmans (Youmans, 1991) and later Hearst (Hearst and Plaunt, 1993; Hearst, 1994)

[1] As opposed to *hierarchical* segmentation (Yaari, 1997)

focused on the similarity between *adjacent* part of texts. They used a sliding window of text and plotted the number of first-used words in the window as a function of the window position within the text. In this plot, segment boundaries correspond to deep valleys followed by sharp upturns. Kan (Kan et al., 1998) expanded the same idea by combining word-usage with visual layout information.

On the other hand, other researchers focused on the similarity between *all* parts of a text. A graphical representation of this similarity is a *dotplot*. Reynar (Reynar, 1998; Reynar, 1999) and Choi (Choi, 2000; Choi et al., 2001) used dotplots in conjunction with divisive clustering (which can be seen as a form of *approximate* and *local* optimization) to perform *linear* text segmentation. A relevant work has been proposed by Yaari (Yaari, 1997) who used *divisive / agglomerative clustering* to perform *hierarchical* segmentation. Another approach to clustering performs *exact* and *global* optimization by dynamic programming; this was used by Ponte and Croft (Ponte and Croft, 1997; Xu and Croft, 1996), Heinonen (Heinonen, 1998) and Utiyama and Isahara (Utiyama and Isahara, 2001).

Finally, other researchers use probabilistic approaches to text segmentation including the use of *hidden Markov models* (Yamron et al., 1999), (Blei and Moreno, 2001). Also Beeferman (Beeferman et al., 1997) calculated the probability distribution on segment boundaries by utilizing word usage statistics, cue words and several other features.

2 The algorithm

2.1 Representation

Suppose that a text contains T sentences and its vocabulary contains L distinct words (e.g words that are not included in the stop list, other wise most sentences would be similar to most others). This text can be represented by a $T \times L$ matrix F defined as follows: for $t = 1, 2, ..., T$ and $l = 1, 2, ..., L$ we set

$$F_{t,l} = \begin{cases} 1 & \text{iff l-th word is in t-th sentence} \\ 0 & \text{else.} \end{cases}$$

The *sentence similarity matrix* of the text is a $T \times T$ matrix D where for $s, t = 1, 2, ..., T$ we set

$$D_{s,t} = \begin{cases} 1 & \text{if} \quad \sum_{l=1}^{L} F_{s,l}F_{t,l} > 0; \\ 0 & \text{if} \quad \sum_{l=1}^{L} F_{s,l}F_{t,l} = 0. \end{cases}$$

This means that $D_{s,t}$ = 1 if the s-th and t-th sentence have at least one word in common. Every part of the original text corresponds to a submatrix of D. It is expected that submatrices which correspond to actual segments will have many sentences with words in common, thus will contain many "ones". Further justification for the use of this similarity matrix and graphical representation can be found in (Petridis et al., 2001), (Reynar, 1998; Reynar, 1999) and (Choi, 2000; Choi et al., 2001)

We make the assumption that segment boundaries always occur at the ends of sentences. A segmentation of a text is a partition of the set $\{1, 2, ..., T\}$ into K subsets (i.e. *segments,* where K is a variable number) of the form $\{1, 2, ..., t_1\}$, $\{t_1 + 1, t_1 + 2, ..., t_2\}$, ..., $\{t_{K-1} + 1, t_{K-1} + 2, ..., T\}$ and can be represented by a vector $\mathbf{t} = (t_0, t_1, ..., t_K)$, where $t_0, t_1, ..., t_K$ are the *segment boundaries* corresponding to the last sentence of each subset.

2.2 Dynamic Programming

Dynamic programming guarantees the optimality of the result with respect to the input and the parameters. Following the approach of (Heinonen, 1998) we use a dynamic programming algorithm which decides the locations of the segment boundaries by calculating the globally optimal splitting $\mathbf{t}$ on the basis of a similarity matrix (or a curve), a preferred fragment length and a cost function defined. Given a similarity matrix D and the parameters μ, σ, r, γ (the role of each of which will be described in the sequel) the dynamic programming algorithm tries to minimize a *segmentation cost function* $J(\mathbf{t}; \mu \sigma, r, \gamma)$ with respect to $\mathbf{t}$ (here $\mathbf{t}$ is the independent variable which is actually a vector specifying the boundary position of each segment and the number of segments K while $\mu \sigma, r, \gamma$ are parameters) which is defined as follows:

$$J(\mathbf{t}; \mu \sigma, r, \gamma) = \sum_{k=1}^{K} \left[\gamma \cdot \frac{(t_k - t_{k-1} - \mu)^2}{2 \cdot \sigma^2} \right.$$

$$- \left[(1-\gamma) \cdot \frac{\sum_{s=t_{k-1}+1}^{t_k} \sum_{t=t_{k-1}+1}^{t_k} D_{s,t}}{(t_k - t_{k-1})^r} \right] . \qquad (1)$$

Hence the sum of the costs of the K segments constitutes the total segmentation cost; the cost of each segment is the sum of the following two terms (with their relative importance weighted by the parameter γ):

1. The term $\frac{(t_k - t_{k-1} - \mu)^2}{2 \cdot \sigma^2}$ corresponds to the length information measured as the deviation from the average segment length. In this sense, μ and σ can be considered as the mean and standard deviation of segment length measured either on the basis of words or on the basis of sentences appearing in the document's segments and can be estimated from *training data.*

2. The term $\frac{\sum_{s=t_{k-1}+1}^{t_k} \sum_{t=t_{k-1}+1}^{t_k} D_{s,t}}{(t_k - t_{k-1})^r}$ corresponds to (word) similarity between sentences. The numerator of this term is the total number of ones in the D submatrix corresponding to the k-th segment. In the case where the parameter r is equal to 2, $(t_k - t_{k-1})^r$ correspond to the area of submatrix and the above fraction corresponds to "segment density". A "generalized density" is obtained when $r \neq 2$ and enables us to control the degree of influence of the surface with regard to the "information" (i.e the number of ones) included in it. Strong intra-segment similarity (as measured by the number of words which are common between sentences belonging to the segment) is indicated by large values of $\frac{\sum_{s=t_{k-1}+1}^{t_k} \sum_{t=t_{k-1}+1}^{t_k} D_{s,t}}{(t_k - t_{k-1})^r}$, irrespective of the exact value of r.

Segments with high density and small deviation from average segment length (i.e a small value of the corresponding $J(\mathbf{t}; \mu\,\sigma, r, \gamma)$ [2]) provide a "good" segmentation vector $\mathbf{t}$. The *global* minimum of $J(\mathbf{t}; \mu\,\sigma, r, \gamma)$ provides the *optimal* segmentation $\widehat{\mathbf{t}}$. It is worth mentioning that the optimal $\widehat{\mathbf{t}}$ specifies both the optimal number of segments K and the optimal positions of the segment boundaries $t_0, t_1, ..., t_K$. In the sequel, our algorithm is presented in a form of pseudocode.

Dynamic Programming Algorithm

```
Input: The $T \times T$ similarity matrix $D$; the parameters $\mu$, $\sigma$, $r$, $\gamma$.
Initilization
For $t = 1, 2, ..T$
    $Sum = 0$
    For $s = 1, 2, ..., t-1$
        $Sum = Sum + D_{s,t}$
    End
    $S_{s,t} = \frac{Sum}{(t-s)^r}$
End
Minimization
$C_0 = 0$, $Z_0 = 0$
For $t = 1, 2, , T$
    $C_t = \infty$
    For $s = 1, 2, ..., t-1$
        If $C_s + S_{s,t} + \frac{(t-s-\mu)^2}{2\sigma^2} \leq C_t$
            $C_t = C_s + S_{s,t} + \frac{(t-s-\mu)^2}{2\sigma^2}$
            $Z_t = s$
        EndIf
    End
End
BackTracking
$K = 0$ , $s_k = T$
While $Z_{s_k} > 0$
        $k = k + 1$
        $s_k = Z_{s_{k-1}}$
End
$K = K + 1$, $Z_k = 0$, $\hat{t}_0 = 0$
For $k = 1, 2, ..., K$
    $\hat{t}_k = s_{K-k}$
End
Output: The optimal segmentation vector $\widehat{\mathbf{t}} = (\hat{t}_0, \hat{t}_1, ..., \hat{t}_K)$.
```

3 Evaluation

3.1 Measures of Segmentation Accuracy

The performance of our algorithm was evaluated by three indices: *precision, recall* and Beeferman's P_k *metric.*

Precision and recall measure segmentation *accuracy*. For the segmentation task, *Precision* is defined as "the number of the estimated segment boundaries which are actual segment boundaries" divided by "the number of the estimated segment boundaries". On the other hand, *Recall* is defined

[2] Small in the *algebraic* sense; $J(\mathbf{t}; \mu\,\sigma, r, \gamma)$ can take both positive and negative values.

as "the number of the estimated segment boundaries which are actual segment boundaries" divided by "the number of the true segment boundaries". High segmentation accuracy is indicated by high values of *both* precision and recall. However, those two indices have some shortcomings. First, high precision can be obtained at the expense of low recall and conversely. Additionally, those two indices penalize equally every inaccurately estimated segment boundary whether it is near or far from a true segment boundary.

An alternative measure P_k which overcomes the shortcomings of precision and recall and measures segmentation *in*accuracy was introduced recently by Beeferman et al (Beeferman et al., 1997). Intuitively, P_k measures the proportion of "sentences which are wrongly predicted to belong to the same segment (while actually they belong in different segments)" or "sentences which are wrongly predicted to belong to different segments (while actually they belong to the same segment)". P_k is a measure of how well the true and hypothetical segmentations agree (with a low value of P_k indicating high accuracy (Beeferman et al., 1997)). P_k penalizes near-boundary errors less than far-boundary errors. Hence P_k evaluates segmentation accuracy more accurately than precision and recall.

3.2 Experiments

Our experiments were conducted using Choi's publicly available text collection (Choi, 2000; Choi et al., 2001). This collection consists of 700 texts, each text being a concatenation of ten text segments. Each segment consists of "the first n sentences of a randomly selected document from the *Brown Corpus* (Francis and Kucera, 1982). (News articles ca**.pos and the informative text cj**.pos)"[3]. The 700 texts can be divided into four datasets Set0, Set1, Set2, Set3, according to the range of n (the number of sentences in a document) as listed in Table 1.

The sample texts were preprocessed i.e. punctuation marks and stop words were removed, while the remaining words were stemmed according to Porter's stemming algorithm (Porter, 1980).

	Set0	Set1	Set2	Set3
Range of n	3-11	3-5	6-8	9-11
no. of texts	400	100	100	100

Table 1

Range of n (number of sentences) and number of documents for the datasets Set0, Set1, Set2, Set3 (Choi's text collection).

We next present two groups of experiments each of which contains two suites of experiments. The difference between the two suites lies in the selection of the parameter values. Our segmentation algorithm uses four parameters: $\mu\,\sigma, \gamma$ and r, where μ and σ can be interpreted as the average and standard deviation of segment length; it is not immediately obvious how to calculate these. One possibility is to calculate the average and standard deviation of the segment length based on the number of *sentences* appearing in the document's segments; this is done in the first suite and for both groups of experiments. The second is based on the number of *words* apparearing in the document's segments; this is done in the second suite and for both groups of experiments.We want to examine this influence on the length model as well as the influence of γ and r in the segmentation accuracy (as measured by Beeferman's P_k) .

In the first group of experiments and for both suites, the following procedure is repeated for Set0, Set1, Set2, Set3.

1. Appropriate μ and σ values are determined using all the texts of the dataset (using the standard statistical estimates based either on the number of sentences or on the number of words).

2. Parameter γ is set to take the values 0.00, 0.01, 0.02, ... , 0.09, 0.1, 0.2, 0.3, ... , 1.0 and r to take the values 0.33, 0.5, 0.66, 1. This yields 20×4=80 possible combinations of γ and r values.

3.Our segmentation algorithm is executed for each (γ, r) combination .

An idea of the influence of γ and r on P_k for both suites of experiments of the first group can be observed in Figures 1-4 (corresponding to Set0, Set1, Set2, Set3). In those figures *Exp 1* refers to the first suite of experiments while *Exp 2* refers to the second suite of experiments.

It can be seen from Figures 1-4 that the best achieved values of P_k are the ones listed in Table 2 corresponding to the results of the first group,

[3] It follows that segment boundaries will always appear at the end of sentences.

where the first three rows correspond to the results obtained by the first suite of experiments, and the last three rows correspond to the results obtained by the second suite of experiments. More precisely, the 1st and the 4th rows contain the values of Precision, the 2nd and the 5th rows contain the values of Recall, while the 3rd and the 6th rows contain the values of P_k.

Set0	Set1	Set2	Set3	All Sets
81.27%	89.54%	89.82%	94.22%	85.53%
84.20%	89.55%	90.00%	94.22%	87.24%
7.00%	4.75%	2.40%	1.00%	5.16%
81.47%	86.47%	83.03%	83.99%	82.77%
80.66%	82%	81.78%	85.22%	81.66%
8.43%	6.82%	5.97%	5.02%	7.36%

Table 2

Exp.Group1: The best Precision, Recall and P_k values for the datasets Set0, Set1, Set2, Set3 and the entire dataset (Choi's text collection) obtained with optimal γ, r values for both experiments of the first group (non validated).

However, only if the optimal values for γ, r as well as the values of μ, σ are known in advance, we can obtain the results of Table 2. In a practical application none of these values will be a priori available. A procedure for determining appropriate values of μ, σ, γ, r is necessary in order to provide a more realistic evaluation of our algorithm.

In the second group of experiments and for both suites, for the determination of the appropriate μ, σ, γ, r values, we first use *training data* and a *parameter validation* procedure. Then our algorithm is evaluated on (previously unseen) *test data*. More specifically, for each of the datasets Set0, Set1, Set2, Set3 we perform the procedure described in the sequel:

1. Half of the texts in the dataset are chosen randomly to be used as training texts; the rest of the samples are set aside to be used as test texts.
2. Appropriate μ and σ values are determined using all the training texts and the standard statistical estimators.
3. Appropriate γ and r values are determined by running the segmentation algorithm on all the training texts with the 80 possible combinations of γ and r values; the one that yields the minimum P_k value is considered to be the optimal (γ, r) combination.
4. The algorithm is applied to the test texts using previously estimated γ, r, μ and σ values.

The above procedure is repeated five times for each of the four datasets for both suites of experiments and the resulting values of precision, recall and P_k are averaged. The results of those experiments are listed in Table 3. Table 3 is exactly the same with Table 2 but now contains the results of the second group of experiments.

Set0	Set1	Set2	Set3	All Sets
82.66%	88.17%	88.68%	92.37%	85.70%
82.78%	87.70%	88.71%	92.44%	85.73%
7.00%	5.45%	3.00%	1.33%	5.39%
83.89%	84.69%	84.50%	88.30%	84.73%
81.41%	84.00%	83.37%	88.09%	83.02%
7.16%	7.54%	5.51%	3.08%	6.40%

Table 3

Exp.Group 2:The Precision, Recall and P_k values for the datasets Set0, Set1, Set2, Set3 and the entire dataset (Choi's text collection) obtained with optimal γ, r values for both experiments of the second group (validated).

Set0	Set1	Set2	Set3	All Sets
9.00%	10.00%	7.00%	5.00%	8.00%
14.00%	10.00%	11.00%	12.00%	13.00%
12.00%	10.00%	9.00%	8.00%	11.00%
12.00%	11.00%	10.00%	9.00%	11.00%
13.00%	18.00%	10.00%	10.00%	13.00%
23.00%	19.00%	21.00%	20.00%	22.00%
10.00%	9.00%	7.00%	5.00%	9.00%
11.00%	13.00%	6.00%	6.00%	10.00%
7.00%	**5.45%**	**3.00%**	**1.33%**	**5.39%**
7.16%	**7.54%**	**5.51%**	**3.08%**	**6.40%**

Table 4

Comparison of several algorithms with respect to the P_k values obtained for the datasets Set0, Set1, Set2, Set3 from both experiments and the entire dataset (Choi's text collection).

Table 4 provides all the results published so far in the literature (Choi, 2000; Choi et al., 2001; Utiyama and Isahara, 2001) regarding Choi's text collection, where we list only the values of P_k since the ones of Precision and Recall are not known. In Table 4, rows 4, 5 and 6 correspond to C99, C99b and C99b,-r algorithms described in (Choi, 2000). Rows 7 and 8 correspond to U00 and U00b proposed in (Utiyama and Isahara, 2001) while rows 1, 2 and 3 correspond to CWM1, CWM2 and CWM3 proposed in (Choi et al., 2001). Row 9 corresponds to the results obtained by the first suite of experiments of our algorithm while row 10 to the ones obtained by the second suite of experiments, both rows for the second group. In both cases, they are still better than any previously reported on Choi's dataset, which means that our algorithm performs considerably better than all the remaining ones. It is worth mentioning than, the best performance has been achieved for γ in the range [0.08, 0.4] and for r equal to either 0.5 or 0.66 for both suites of experiments.

3.3 Discussion

From all the results obtained, we can conclude that our segmentation algorithm on Choi's text collection achieves significantly better results than the ones previously reported (Choi, 2000; Choi et al., 2001; Utiyama and Isahara, 2001). The computational complexity of our algorithm is comparable to that of the other methods (namely $O(T^2)$ where T is the number of sentences) [4]. Finally, our algorithm has the advantage of automatically determining the optimal number of segments.

We believe that the good performance of our algorithm is the result of the combination of the following facts: First, the use of a segment length term in the cost function seems to improve segmentation accuracy significantly, as it can be seen in Figures 1-4. Second, *measuring segment length on the basis of sentences rather on the basis of words improves segmentation accuracy.* Third, the use of "generalized density" ($r \neq 2$) appears to significantly improve performance. Even though the use of "true density" ($r = 2$) appears more natural, the best segmentation performance (minimum value of P_k) is achieved for significantly smaller values of r (as it can be see from the Figures 1-4 and the obtained results). This performance in most cases is improved when using appropriate values of μ,σ,γ and r derived from training data and parameter validation.

Finally, it is worth mentioning that our approach is "global" in two respects. First, sentence similarity is computed globally through the use of the D matrix and dotplot. Second, this global similarity information is also optimized globally by the use of the dynamic programming algorithm. This is in contrast with the local optimization of global information (used by Choi) and global optimization of local information (used by Heinonen).

4 Conclusion

We have presented a dynamic programming algorithm which performs text segmentation by global minimization of a segmentation cost consisting of two terms: within-segment word similarity and prior information about segment length. The performance of our algorithm is quite satisfactory considering that it yields the best results reported so far on the segmentation of Choi's text collection. In the future we intent to focus on the calculation of the length model based on the average number of sentences as opposed to the calculation of the length model based on the average number of words in the documents's segments.We also intent to use other measures of sentece similarity. We also plan to apply our algorithm to a wide spectrum of text segmentation tasks. We are interested in segmentation of non artificial e.g real texts, texts having a diverse distribution of segment length, long texts, change-of-topic detection in newsfeeds and segmentation of non-English (particularly Greek) texts.

[4] Our algorithm was executed on a Pentium III 600Mhz computer with 256Mbyte RAM. For segmenting a single text, our algorithm takes on average 0.91seconds, U00b (Utiyama and Isahara, 2001) 1.37, U00 (Utiyama and Isahara, 2001) 1.36, C99b 1.45 (Choi, 2000), (Choi et al., 2001) and C99 (Choi, 2000; Choi et al., 2001) 1.49 seconds.

References

Beeferman, D., Berger, A., and Lafferty, J. 1997. *Text segmentation using exponential models.* In Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing, pp. 35-46.

Blei, D.M. and Moreno, P.J. 2001. *Topic segmentation with an aspect hidden Markov model.* Tech. Rep. CRL 2001-07, COMPAQ Cambridge Research Lab.

Choi, F.Y.Y. 2000. *Advances in domain independent linear text segmentation.* In Proceedings of the 1st

Meeting of the North American Chapter of the Association for Computational Linguistics, pp. 26-33.

Choi, F.Y.Y., Wiemer-Hastings, P. & Moore, J. 2001. *Latent semantic analysis for text segmentation.* In Proceedings of the 6th Conference on Empirical Methods in Natural Language Processing, pp.109–117.

Francis, W.N. and Kucera, H. 1982. *Frequency Analysis of English Usage: Lexicon and Grammar.* Houghton Mifflin.

Halliday, M. and Hasan, R. 1976. *Cohesion in English.* Longman.

Hearst, M. A. and Plaunt, C. 1993. *Subtopic structuring for full-length document access.* In Proceedings of the 16th Annual International of Association of Computer Machinery - Special Interest Group on Information Retrieval (ACM / SIGIR) Conference on Research and Development in Information Retrieval, pp. 59-68.

Hearst, M. A. 1994. *Multi-paragraph segmentation of expository texts.* In Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistic, pp. 9-16.

Heinonen, O. 1998. *Optimal Multi-Paragraph Text Segmentation by Dynamic Programming.* In Proceedings of 17th International Conference on Computational Linguistics (COLING-ACL'98), pp.1484-1486.

Hirschberg, J., and Litman, D. 1993. *Empirical studies on the disambiguation and cue phrases.* Computational Linguistics,vol.19, pp.501-530.

Kan, M., Klavans, J.L. and McKeown, K. R. 1998. *Linear segmentation and segment significance.* In Proceedings of the 6th International Workshop of Very Large Corpora, pp. 197-205.

Kozima, H. 1993. *Text Segmentation based on similarity between words.* In Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics, pp. 286-288.

Kozima, H and Furugori, T. 1993. *Similarity between words computed by spreading activation on an English dictionary.* In Proceedings of 6th Conference of the European Chapter of the Association for Computational Linguistics, pp. 232-239.

Morris, J. and Hirst, G. 1991. *Lexical cohesion computed by thesaural relations as an indicator of the structure of text.* Computational Linguistics, vol.17, pp.21-42.

Passoneau, R. and Litman, D.J. 1993. *Intention - based segmentation: Human reliability and correlation with linguistic cues.* In Proceedings of the 31st Meeting of the Association for Computational Liguistics, pp. 148-155.

Petridis, V., Kaburlazos, V., Fragkou, P., Kehagias, A. 2001. *Text Classification using the -FLNMAP Neural Network.* Proceedings of the IJCNN'01 on Neural Networks.

Ponte, J. M. and Croft, W. B. 1997. *Text segmentation by topic.* In Proceedings of the 1st European Conference on Research and Advanced Technology for Digital Libraries, pp.120-129.

Porter, M., F. 1980. *An algorithm for suffix stripping.* newblock Program, vol.14, pp.130-137.

Reynar, J.C. 1998. *Topic Segmentation: Algorithms and Applications.* Ph.D. Thesis, Dept. of Computer Science, Univ. of Pennsylvania.

Reynar, J.C. 1999. *Statistical models for topic segmentation.* In Proceedings of the 37th Annual Meeting of the Association for Computational Liguistics, pp. 357-364.

Roget, P.M. 1977. *Roget's International Thesaurus.* Harper and Row, 4th edition.

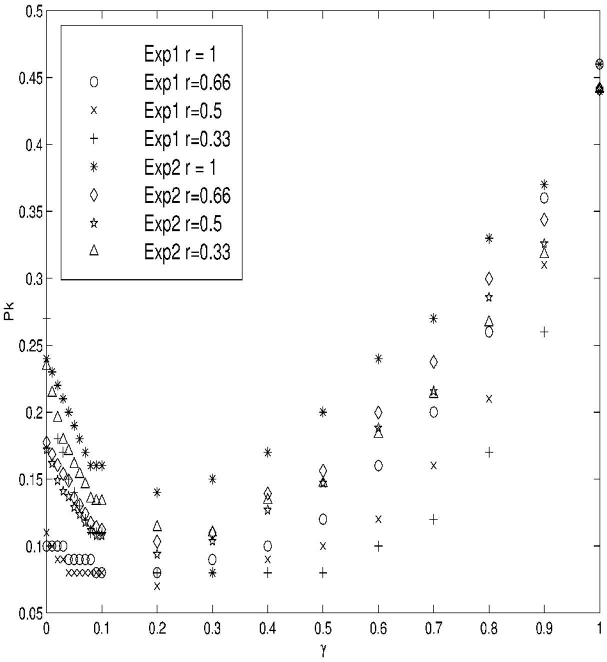
Utiyama, M., and Isahara, H. 2001. *A statistical model for domain - independent text segmentation.* In Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics, pp.491-498.

Xu, J. and Croft, W.B. 1996. *Query expansion using local and global document analysis.* In Proceedings of the 19th Annual International of Association of Computer Machinery - Special Interest Group on Information Retrieval (ACM / SIGIR) Conference on Research and Development in Information Retrieval, pp. 4-11.

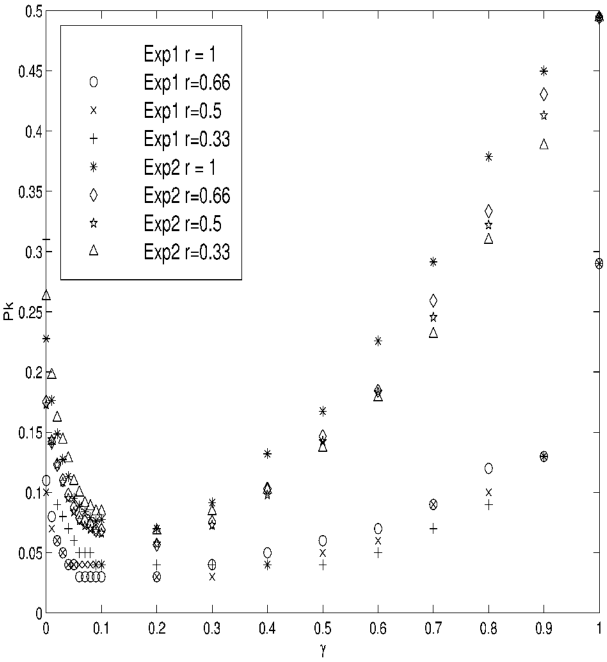
Yaari, Y. 1997. *Segmentation of expository texts by hierarchical agglomerative clustering.* In Proceedings of the Conference on Recent Advances in Natural Language Processing, pp.59-65.

J. Yamron, I. Carp, L. Gillick, S.Lowe, and P. van Mulbregt. 1999. *Topic tracking in a news stream.* In Proceedings of DARPA Broadcast News Workshop, pp. 133-136.

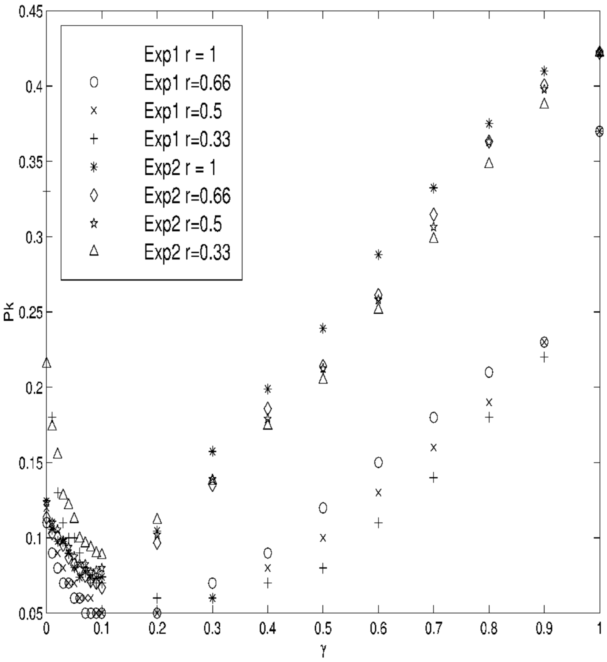
Youmans, G. 1991. *A new tool for discourse analysis: The vocabulary management profile.* Language, vol. 67, pp.763-789.



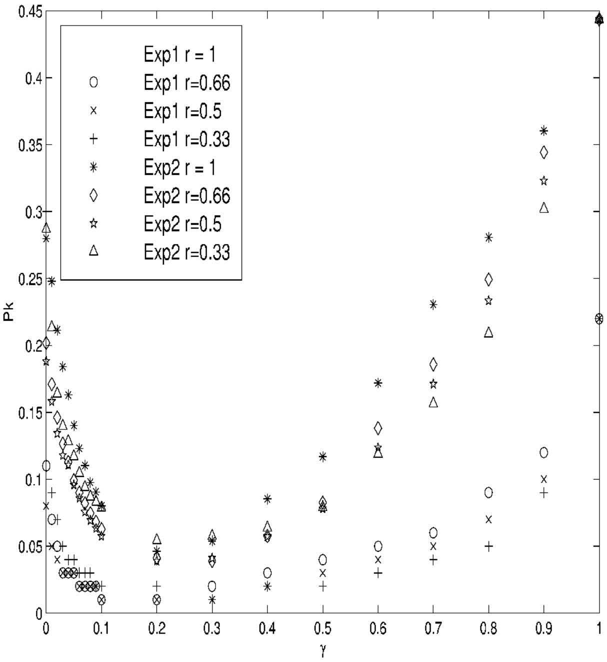
(a) **Figure 1:**Pk plotted as a function of γ and r for Set0



(c) **Figure 3:**Pk plotted as a function of γ and r for Set2



(b) **Figure 2:**Pk plotted as a function of γ and r for Set1



(d) **Figure 4:**Pk plotted as a function of γ and r for Set3