A Dynamic Programming Algorithm for Linear Text Segmentation

P. Fragkou, V. Petridis and Ath. Kehagias

September 20, 2002

Abstract

In this paper we introduce a dynamic programming algorithm which performs linear text segmentation by global minimization of a segmentation cost function which incorporates two factors: (a) within-segment word similarity and (b) prior information about segment length. We evaluate segmentation accuracy of the algorithm by precision, recall and Beeferman's segmentation metric. On a segmentation task which involves Choi's text collection, the algorithm achieves the best segmentation accuracy so far reported in the literature. The algorithm also achieves high accuracy on a second task which involves previously unused texts.

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

1 INTRODUCTION

An important problem in information retrieval is *text segmentation*. The goal is to divide a text into *homogeneous* segments, so that each segment deals with a particular subject while contiguous segments deal with different subjects. In this manner documents relevant to a query can be retrieved from a large database of unformatted (or loosely formatted) text.

In this paper we propose a dynamic programming algorithm which performs *linear* segmentation[1] by global minimization of *segmentation cost*. We use a segmentation cost function which incorporates two factors: (a) *within-segment word similarity* and (b) *prior information about segment length*.

Two main versions of the text segmentation problem appear in the literature. The first version concerns segmentation of a single large text into its constituent parts (e.g. to segment an article into sections); the second version concerns segmentation of a stream of independent, concatenated texts (e.g. to segment a transcript of a TV news program into separate stories). The algorithm we present here can be applied to either of these problems, but the experiments we present only deal with texts of the second type.

A major issue in text segmentation is the choice of segment homogeneity (and heterogeneity) criteria. Some segmentation algorithms use linguistic criteria such as cue phrases, punctuation marks, prosodic features, reference, syntax and lexical attraction [1, 2, 3, 15, 24, 25]. Another approach utilizes statistical similarity measures such as word cooccurrence (according to Halliday and Hasan [8] parts of a text which have similar vocabulary are likely to belong to a coherent topic segment). Several researchers have used the statistical approach in the past.

For example, Morris and Hirst [22, 23] introduced a linear discourse segmentation algorithm based on *lexical cohesion relations* determined by use of Roget's thesaurus [31, 32]. Similarly, Kozima [18, 19, 20] has proposed a method which computes the semantic similarity between words using a semantic network constructed from a subset of the Longman Dictionary of Contemporary English. Spreading activation on the network is used to compute the similarity between two words and represents the

[1] As opposed to *hierarchical* segmentation [35].

strength of lexical cohesion; it also provides information about similarity and coherence which can be used for linear text segmentation: local minima of the similarity scores correspond to the positions of topic boundaries in the text.

Consider a sliding window of text and plot 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 (this is in accordance to the previously mentioned hypothesis that shifts in topic are likely to be accompanied by changes in word usage). This approach to text segmentation was introduced by Youmans [38, 39] and has also been used by Hearst [9, 10, 11, 12, 13] and by Kan [16, 17] (who combined word-usage with visual layout information).

The works of the previous paragraph utilize the similarity between *adjacent* parts of the text. A more general approach involves the similarity between *all* parts of a text. This similarity can be represented graphically by a *dotplot* (see Section 2.1). Dotplots in conjunction with divisive clustering have been used by Reynar [28, 29, 30] and by Choi [5, 6] to perform linear text segmentation. *Divisive / agglomerative clustering* has also been used by Yaari [35, 36] to perform *hierarchical* segmentation.

Clustering can be seen as a form of *approximate* and *local* optimization; a more sophisticated approach to segmentation can be obtained by use of dynamic programming to perform *exact* and *global* optimization. This has been used by Ponte and Croft [26, 34], Heinonen [14], Utiyama and Isahara [33] and others.

Finally, probabilistically motivated approaches to text segmentation include the use of *hidden Markov models* [4, 21, 37] and Beeferman's work [1, 2] which utilizes word usage statistics, cue words and several other features to construct a probability distribution on segment boundaries.

2 THE SEGMENTATION ALGORITHM

2.1 Representation

Suppose that a text contains T sentences and has a vocabulary of L distinct words. Now consider a $T \times L$ matrix F defined as follows: for $t = 1, 2, ..., T$ and $l = 1, 2, ..., L$ we set

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

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

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

Hence $D_{s,t} = 1$ if the s-th and t-th sentence have at least one word in common. In Figure 1 we give a *dotplot* of a D matrix (this particular matrix corresponds to a 91-sentence text which belongs to the collection of Section 3.2). Ones are plotted as black squares and zeros as white squares.

Figure 1

Assuming that segment boundaries occur at the ends of sentences, a segmentation of the text is a partition of the set $\{1, 2, ..., T\}$ into K subsets (i.e. *segments*) of the form $\{1, 2, ..., t_1\}$, $\{t_1 + 1, t_1 + 2, ..., t_2\}$, ..., $\{t_{K-1} + 1, t_{K-1} + 2, ..., T\}$. A shorter representation of the segmentation can be given in terms of a vector $\mathbf{t} = (t_0, t_1, ..., t_K)$, where $t_0, t_1, ..., t_K$ are the *segment boundaries* and must satisfy:

$$0 = t_0 < t_1 < ... < t_{K-1} < t_K = T.$$

Note that K, the length of the vector, is variable; hence $\mathbf{t}$ can describe any number of segments (however we must have $K \leq T$, the number of sentences).

2.2 The Cost Function

Every part of the original text corresponds to a submatrix of D. It can be expected that submatrices which correspond to actual segments will contain many 1's, since the respective segments will have many sentences with words in common (indeed, in Figure 1 we see several "high-density" regions which we expect to correspond to actual segments). Hence a "good" segmentation should maximize the *density* of 1's in the submatrices of D which correspond to segments. On the other hand, in many situations we have additional information regarding the *length* of the segments. For example, we may know that the *average segment length* is μ. This information should be used to improve the segmentation accuracy.

The above considerations can be formalized by defining the *segmentation cost function* $J(\mathbf{t}; \mu, \sigma, r, \gamma)$ (here $\mathbf{t}$ is the independent variable and μ, σ, r, γ are parameters) as follows

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

Hence the total segmentation cost is the sum of the costs of the K segments; the cost of each segment is the sum of two terms (with their relative importance weighted by the parameter γ). The interpretation of the two terms is as follows.

1. The term $\frac{(t_k - t_{k-1} - \mu)^2}{2 \cdot \sigma^2}$ corresponds to length information; in particular it measures deviation from the average segment length. Indeed, μ and σ can be interpreted as the mean and standard deviation of segment length which can be estimated from *training data*. Small values of this term indicate agreement with the expected segment length[2].

2. The term $\frac{\sum_{t=t_{k-1}+1}^{t_k} \sum_{t=t_{k-1}+1}^{t_k} D_{s,t}}{(t_k - t_{k-1})^r}$ corresponds to (word) similarity between sentences. Note that $\sum_{t=t_{k-1}+1}^{t_k} \sum_{t=t_{k-1}+1}^{t_k} D_{s,t}$ is the total number of ones in the D submatrix corresponding to the k-th segment; also, when then the parameter r is equal to 2 then $(t_k - t_{k-1})^r$ is the area of submatrix. Hence, when $r = 2$, the term $\frac{\sum_{t=t_{k-1}+1}^{t_k} \sum_{t=t_{k-1}+1}^{t_k} D_{s,t}}{(t_k - t_{k-1})^r}$ corresponds to "segment density". When $r \neq 2$ we obtain a "generalized density". Irrespective of the exact value of r, large values of $\frac{\sum_{t=t_{k-1}+1}^{t_k} \sum_{t=t_{k-1}+1}^{t_k} D_{s,t}}{(t_k - t_{k-1})^r}$ indicate strong intra-segment similarity (as measured by the number of words which are common between sentences belonging to the segment).

A "good" segmentation vector $\mathbf{t}$ gives segments with high density and small deviation from average segment length and hence the corresponding $J(\mathbf{t}; \mu, \sigma, r, \gamma)$ takes a small value[3]. The *optimal* segmentation $\widehat{\mathbf{t}}$ gives the *global* minimum of $J(\mathbf{t}; \mu, \sigma, r, \gamma)$. Note that the optimal $\widehat{\mathbf{t}}$ specifies both the optimal number of segments K and the optimal positions of the segment boundaries $t_0, t_1, ..., t_K$.

2.3 Dynamic Programming

Hence the task is to obtain the segmentation vector $\mathbf{t}$ which minimizes $J(\mathbf{t}; \mu, \sigma, r, \gamma)$. Following the approach of [14] we use a dynamic programming algorithm which finds the globally optimal $\widehat{\mathbf{t}}$. The input to the algorithm is the similarity matrix D and the parameters μ, σ, r, γ; the output is $\widehat{\mathbf{t}}$, computed in time $O(T^2)$ (where T is the number of sentences).

[2] Experiments not reported in this paper indicate that segmentation is more accurate when segment length is measured in terms of sentences than in terms of words. The reason for this is probably that the number of words appearing in a segment has larger variation than the number of sentences.

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

Dynamic Programming for Text Segmentation

Input: The $T \times T$ similarity matrix D; the parameters μ, σ, r, γ.
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 $C_t \leq \frac{(t-s-\mu)^2}{2\sigma^2}$
            Break out of this loop
        EndIf
        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
$K = K + 1$
$Z_k = 0$
$\widehat{t_0} = 0$
For $k = 1, 2, ..., K$
    $\widehat{t_k} = s_{K-k}$
End
Output: The optimal segmentation vector $\widehat{\mathbf{t}} = (\widehat{t_0}, \widehat{t_1}, ..., \widehat{t_K})$.

---

3 EXPERIMENTS

3.1 Measures of Segmentation Accuracy

In the following experiments, we evaluate the performance of our algorithm by three indices: *precision, recall* and Beeferman's P_k *metric.*

Precision and recall measure segmentation *accuracy*; they are defined as follows:

$$\text{precision} = \frac{\text{no. of estimated segment boundaries which are actual segment boundaries}}{\text{no. of estimated segment boundaries}};$$

$$\text{recall} = \frac{\text{no. of estimated segment boundaries which are actual segment boundaries}}{\text{no. of true segment boundaries}}.$$

Precision and recall take values in the range 0% to 100%. High values of *both* precision and recall indicate high segmentation accuracy. However, a shortcoming of these two indices is that high precision can be obtained at the expense of low recall and conversely. A further shortcoming is that every inaccurately estimated segment boundary is penalized equally whether it is near or far from a true segment boundary.

To overcome the shortcomings of precision and recall, Beeferman et al. have recently [2, 3] introduced P_k, a measure of segmentation *in*accuracy. 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 formally defined as follows. Given a document of T sentences, first define for every $s, t = 1, 2, ..., T$ the quantities $\delta_0(s,t)$ and $\delta_1(s,t)$ as follows

$$\delta_0(s,t) = \begin{cases} 1 & \text{iff sentences } s \text{ and } t \text{ belong to the same segment in the true segmentation;} \\ 0 & \text{else.} \end{cases}$$

$$\delta_1(s,t) = \begin{cases} 1 & \text{iff sentences } s \text{ and } t \text{ belong to the same segment in the hypothetical segmentation;} \\ 0 & \text{else.} \end{cases}$$

Next introduce a function $d(s,t)$ $(s, t = 1, 2, ..., T)$ which is a *distance probability distribution* over the set of possible distances between sentences randomly chosen from the document. Finally, define

$$P_k = \sum_{1 \le s \le t \le T} d(s,t) \mathbf{1}\left(\delta_0(s,t) = \delta_1(s,t)\right)$$

where $\mathbf{1}(a = b)$ is the *indicator function* (equal to 1 when $a = b$ and to 0 otherwise). It can be shown that for appropriate choices of d, P_k takes values in the range 0% to 100% and is a measure of how well the true and hypothetical segmentations agree (with a low value of P_k indicating high accuracy). Appropriate choices of d and methods for the computation of P_k are given in [2, 3], where it is also shown that P_k penalizes near-boundary errors less than far-boundary errors, hence evaluating segmentation accuracy more accurately than precision and recall.

3.2 Experiment Group no.1

In this group of experiments we use Choi's publicly available text collection [5, 6]. 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* [7]. (News articles ca**.pos and the informative text cj**.pos)"[4]. The 700 texts can be divided into four datasets Set0, Set1, Set2, Set3, according to the range of n as listed in Table 1.

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

	Set0	**Set1**	**Set2**	**Set3**
Range of n (number of sentences in a document)	**3-11**	**3-5**	**6-8**	**9-11**
no. of texts	400	100	100	100

Table 1

The texts were preprocessed by removing punctuation marks and stop-words (determined by a *stoplist*) and stemming the remaining words by Porter's algorithm [27].

We next present two suites of experiments. The difference of the two suites lies in the selection of parameter values. Recall that the segmentation algorithm uses four parameters: μ, σ, γ and r. As already mentioned, μ and σ can be interpreted as the average and standard deviation of segment length; it is not immediately obvious how to choose values for γ and r.

In the first suite of experiments the goal is to determine the influence of γ and r on segmentation accuracy (as measured by Beeferman's P_k) The following procedure is repeated for Set0, Set1, Set2, Set3.

1. We determine appropriate μ and σ values using all the texts of the dataset (using the standard statistical estimates).

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

3. For each (γ, r) combination we run the segmentation algorithm.

The influence of γ and r on P_k can be observed in Figures 2-5 (corresponding to Set0, Set1, Set2, Set3).

Figures 2-5 here.

It can be seen from Figures 2-5 that the best achieved values of P_k are the ones listed in Table 2. These results are better than any other published in the literature [5, 6, 33] regarding Choi's text collection.

Group	P_k
Set0 (3-11)	7.00%
Set1 (3-5)	4.75%
Set2 (6-8)	2.40%
Set3 (9-11)	1.00%
All Sets	5.16%

Table 2

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

In the second suite of experiments we first use *training data* and a *parameter validation* procedure to determine appropriate μ, σ, γ, r values. Then we evaluate the algorithm on (previously unseen) *test data*. More specifically, the following procedure is performed for each of the datasets Set0, Set1, Set2, Set3.

1. We choose randomly half of the texts in the dataset to be used as training texts; the rest of the samples are set aside to be used as test texts.

2. We determine appopriate μ and σ values using all the training texts and the standard statistical estimators.

3. We determine appropriate γ and r values by running the segmentation algorithm on all the training texts with the 80 possible combinations of γ and r values; the optimal (γ, r) combination is the one which yields the minimum P_k value.

4. We apply the algorithm 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 and the resulting values of precision, recall and P_k are averaged. The average values are listed in Table 3.

Group	**Precision**	**Recall**	P_k
Set0 (3-11)	82.66%	82.78%	7.00%
Set1 (3-5)	88.17%	87.70%	5.45%
Set2 (6-8)	88.68%	88.71%	3.00%
Set3 (9-11)	92.37%	92.44%	1.33%
All Sets	85.70%	85.73%	5.39%

Table 3

While the P_k values of Table 3 are slightly worse than the ones of Table 2, they still are better than any previously reported on Choi's dataset. In Table 4 we list the P_k values achieved by our algorithm to the ones obtained by various other segmentation algorithms operating on Choi's dataset [5, 6, 33]. The first row lists the algorithm used, the second row lists the publication in which the result appears; the next four rows list P_k for Set0, Set1, Set2, Set3; the final row lists the P_k value averaged over all samples[5]. The results of our segmentation algorithm appear in the last row. It can be seen that our algorithm performs considerably better than all the remaining ones. Let us note that 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.

Method	CWM1	CWM2	CWM3	C99b	C99	C99b,-r	U00b	U00	Ours
Publication	[6]	[6]	[6]	[5]	[5]	[5]	[33]	[33]	
Set0	9.00%	14.00%	12.00%	12.00%	13.00%	23.00%	10.00%	11.00%	7.00%
Set1	10.00%	10.00%	10.00%	11.00%	18.00%	19.00%	9.00%	13.00%	5.45%
Set2	7.00%	11.00%	9.00%	10.00%	10.00%	21.00%	7.00%	6.00%	3.00%
Set3	5.00%	12.00%	8.00%	9.00%	10.00%	20.00%	5.00%	6.00%	1.33%
All Sets	8.00%	13.00%	11.00%	11.00%	13.00%	22.00%	9.00%	10.00%	5.39%

Table 4

3.3 Experiment Group no.2

In the second group of experiments, we use a text collection compiled from the "Press: Reporting" group of Brown Corpus documents. Our collection consists of ten datasets: Set0, Set1, Set2, ... , Set9. Each dataset contains ten texts and each text is generated according to the following procedure (which guarantees that each text contains ten segments).

[5]Only P_k results are listed, since precision and recall results are not given in [5, 6, 33].

1. Integers $L_{\min}, L_{\max}$ are chosen; as will be seen in the next step, $L_{\min}$ is the minimum and $L_{\max}$ is the maximum number of *lines* in the segment.

2. For $i = 1, 2, ..., 10$ a random integer L_i is generated under a uniform distribution in the set $\{L_{\min}, L_{\min+1}, ..., L_{\max}\}$.

3. The i-th segment is formed by extracting L_i consecutive lines from a randomly selected Brown Corpus document (starting at the first line of the document).

4. The ten segments are concatenated to form the text.

Each of the ten datasets uses different $L_{\min}$ and $L_{\max}$ values; the 10 different $(L_{\min}, L_{\max})$ pairs are listed in Table 5.

	Set0	**Set1**	**Set2**	**Set3**	**Set4**	**Set5**	**Set6**	**Set7**	**Set8**	**Set9**
$L_{\min}$	15	15	15	15	20	20	20	25	25	30
$L_{\max}$	20	25	30	35	25	30	35	30	35	35

Table 5

Similarly to Experiment Group no.1, we conduct two suites of experiments. In the first suite our aim is to find the best possible segmentation performance (i.e. using the optimal values of γ and r). In Figures 6-9 we plot P_k as a function of γ and r (for datasets Set4, Set7, Set8 and Set9).

Figures 6-9 here.

In the second suite of experiments we compute μ, σ and optimal γ and r values by the previously described validation procedure. In Table 6 we list the segmentation accuracy indices for each of the ten datasets (as obtained by the validated parameter values).

Group	**Precision**	**Recall**	**Beeferman**
Set0	67.22%	68.89%	9.92%
Set1	59.47%	60.00%	8.93%
Set2	66.64%	67.78%	13.59%
Set3	63.63%	68.89%	11.50%
Set4	79.72%	78.89%	5.13%
Set5	71.84%	75.56%	7.56%
Set6	68.06%	70.00%	8.63%
Set7	72.33%	75.56%	6.84%
Set8	63.13%	70.00%	8.62%
Set9	70.89%	74.45%	5.87%
All Sets	68.30%	71.00%	8.60%

Table 6

Since the above collection has not been previously used in the literature, we cannot provide a comparative assesment. However, let us note that this collection furnishes a harder problem than the one used in Experiment Group no.1, because both the vocabulary and the number of lines / sentences contained in each segment are larger.

3.4 Discussion

The application of our segmentation algorithm on Choi's text collection yields significantly better results than the ones previously reported [5, 6, 33]. Our algorithm also performs quite well on the previously untested text collection of Experiment Group no.2. 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). In Table 7 we give execution times (for segmenting a single text) of our algorithm and some of the algorithms of [5, 6, 33]. Note that our algorithm was executed on a Pentium III 600Mhz computer with 256Mbyte RAM; it is possible that the remaining algorithms of Table 7 were executed on slower machines.

Algorithm	U00b	U00	C99b	C99	Our Algo
Avg Exec. Time in sec	1.37	1.36	1.45	1.49	0.91

Table 7

Finally, an additional attractive feature of our algorithm is the automatic determination of optimal number of segments.

Let us now point out what we consider to be the reasons for the good performance of our algorithm.

1. The use of a segment length term in the cost function improves segmentation accuracy significantly. This can be seen in Figures 2-9. In these figures the P_k value computed for $\gamma = 0$ corresponds to segmentation without any length information. It can be seen that in this case the P_k values are significantly worse than the optimal ones. We also stress that measuring segment length in sentences rather than words significantly improves segmentation accuracy.

2. In addition, the use of "generalized density" ($r \neq 2$) significantly improves performance. While the use of "true density" ($r = 2$) appears more natural, it can be seen in Figures 2-9 that the best segmentation performance (minimum value of P_k) is achieved for significantly smaller values of r.The use of "generalized density" allows us to better control the influence of segment area in proportion to the "information contained in the segment".

3. However, segment length information and "generalized density" will only yield improved segmentation if used with appropriate values of μ, σ, γ and r. This becomes possible by the use of training data and parameter validation.

4. Finally, let us note 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. Compare this to the local optimization of global information (for instance Choi uses divisive clustering, which performs local optimization, to segment a global similarity matrix) as well as to the global optimization of local information (for instance Heinonen uses dynamic programming to globally optimize local, adjacent sentences similarity).

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; in particular it yields the best results reported so far on the segmentation of Choi's text collection. In the future we plan to apply our algorithm to a wide spectrum of text segmentation tasks. We are interested in

segmentation of long texts, change-of-topic detection in newsfeeds and segmentation of non-English (particularly Greek) texts[6]. Also, we want to examine the interplay between the length and similarity terms from a theoretical point of view (for instance by introducing appropriate probabilistic models which allow for a Maximum Likelihood interpretation of the cost function).

References

[1] Beeferman, D., Berger, A., and Lafferty, J. (1997). "A model of lexical attraction and repulsion". In *Proceedings of the 35th Annual Meeting of the Association of Computational Linguistics*, pp. 373-380.

[2] 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.

[3] Beeferman, D., Berger, A. and Lafferty, J.(1999). "Statistical models for text segmentation". *Machine Learning*, vol. 34, pp.177-210.

[4] 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.

[5] 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.

[6] 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.

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

[8] Halliday, M. and Hasan, R.(1976). *Cohesion in English.* Longman.

[9] Hearst, M. A. (1993). "TextTiling: A quantitative approach to discourse segmentation". Tech. Rep. 93/24, Dept. of Computer Science, University of California, Berkeley.

[10] Hearst, M. A. (1994). *Context and Structure in Automated Full-Text Information Access.* Ph.D. Thesis, Report No. UCB/CSD-94/836, Dept. of Computer Science, University of California, Berkeley.

[11] 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.

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

[6] Compare the following remarks in [33]: "*it is important to assess the performance of systems by using real texts. These texts should also be domain independent. They should also be multi-lingual if we want to test the multilinguality of systems*".

[13] Hearst, M. A. (1992). "Automatic acquisition of hyponyms from large text corpora". In *Proceedings of the 14th International Conference on Computational Linguistics*, pp.539-545.

[14] 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.

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

[16] 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.

[17] Kan, Min-Yen.(2000). *Combining visual layout and lexical cohesion features for text segmentation.* Tech. Rep. CUCS-002-01, Dept. of Computer Science , Columbia University.

[18] 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.

[19] 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.

[20] Kozima, H and Furugori, T. (1994). "Segmenting narrative text into coherent scenes". *Literary and Linguistic Computing*, vol.9, pp. 13-19.

[21] Mittendorf, E. and Schuble, P. (1996). "Document and passage retrieval based on hidden Markov models". 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. 318-327.

[22] Morris, J.(1988). *Lexical cohesion, the thesaurus and the structure of the text.* Tech. Rep. CSRI-219, Computer Systems Research Institute, Univerity of Toronto.

[23] 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.

[24] 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.

[25] Passoneau, R. and Litman, D.J. (1996). "Empirical analysis of three dimensions of spoken discourse: Segmentation, coherence and linguistic devices". In *Computational and Conversational Discourse: Burning Issues- An Interdisciplinary Account*, Hovy, E.H. and Scott, D., R., editors, pp. 161-194, Springer.

[26] 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.

[27] Porter, M., F.(1980). "An algorithm for suffix stripping". *Program*, vol.14, pp.130-137.

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

[29] Reynar, J.C. (1994). "An automatic method of finding topic boundaries". In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pp. 331-333.

[30] Reynar, J.C. and Ratnaparkhi, A. (1997). "A maximum entropy approach to identifying sentence boundaries". In *Proceedings of the 5th Conference on Applied Natural Language Processing*, pp. 16-19.

[31] Roget, P.M. (1911). *Roget's International Thesaurus.* Cromwell, 1st edition.

[32] Roget, P.M. (1977). *Roget's International Thesaurus.* Harper and Row, 4th edition.

[33] 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.

[34] 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.

[35] 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.

[36] Yaari, Y. (1999). *Intelligent exploration of expository texts.* Ph.D. thesis. Dept. of Computer Science, Bar-Ilan University.

[37] 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.

[38] Youmans, G. (1990). "Measuring lexical style and competence: The type-token vocabulary curve". *Style*, vol.24, pp.584-599.

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

Table Captions

Table 1: Range of n (number of sentences) and number of documents for the datasets Set0, Set1, Set2, Set3 (Choi's text collection).

Table 2: The best P_k values for the datasets Set0, Set1, Set2, Set3 and the entire dataset (Choi's text collection) obtained with optimal γ, r values.

Table 3: The precision, recall and P_k values for the datasets Set0, Set1, Set2, Set3 and the entire dataset (Choi's text collection) obtained with validated γ, r values.

Table 4: Comparison of several algorithms with respect to the P_k values obtained for the datasets Set0, Set1, Set2, Set3 and the entire dataset (Choi's text collection).

Table 5: $L_{\min}$ and $L_{\max}$ (minimum and maximum number of lines) for datasets Set0, Set1, ... , Set9 (our new text collection).

Table 6: Precision, recall and P_k for datasets Set0, Set1, ... , Set9 (our new text collection) using validated parameter values.

Table 7: Comparison of our algorithm and the algorithms of [5, 6, 33] with respect to average execution times for segmenting a single text.

Figure Captions

Figure 1: Plot of the sentence similarity matrix D for a text with 91 sentences. 1's are plotted as black squares and 0's are plotted as white squares.

Figure 2: P_k plotted as a function of γ and r for the texts of Set0 (Choi's text collection).

Figure 3: P_k plotted as a function of γ and r for the texts of Set1 (Choi's text collection).

Figure 4: P_k plotted as a function of γ and r for the texts of Set2 (Choi's text collection).

Figure 5: P_k plotted as a function of γ and r for the texts of Set3 (Choi's text collection).

Figure 6: P_k plotted as a function of γ and r for the texts of Set4 (our new text collection).

Figure 7: P_k plotted as a function of γ and r for the texts of Set7 (our new text collection).

Figure 8: P_k plotted as a function of γ and r for the texts of Set8 (our new text collection).

Figure 9: P_k plotted as a function of γ and r for the texts of Set9 (our new text collection).

Footnotes

[1]As opposed to *hierarchical* segmentation [35].

[2]Experiments not reported in this paper indicate that segmentation is more accurate when segment length is measured in terms of sentences than in terms of words. The reason for this is probably that the number of words appearing in a segment has larger variation than the number of sentences.

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

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

[5]Only P_k results are listed, since precision and recall results are not given in [5, 6, 33].

[6]Compare the following remarks in [33]: "*it is important to assess the performance of systems by using real texts. These texts should also be domain independent. They should also be multi-lingual if we want to test the multilinguality of systems*".

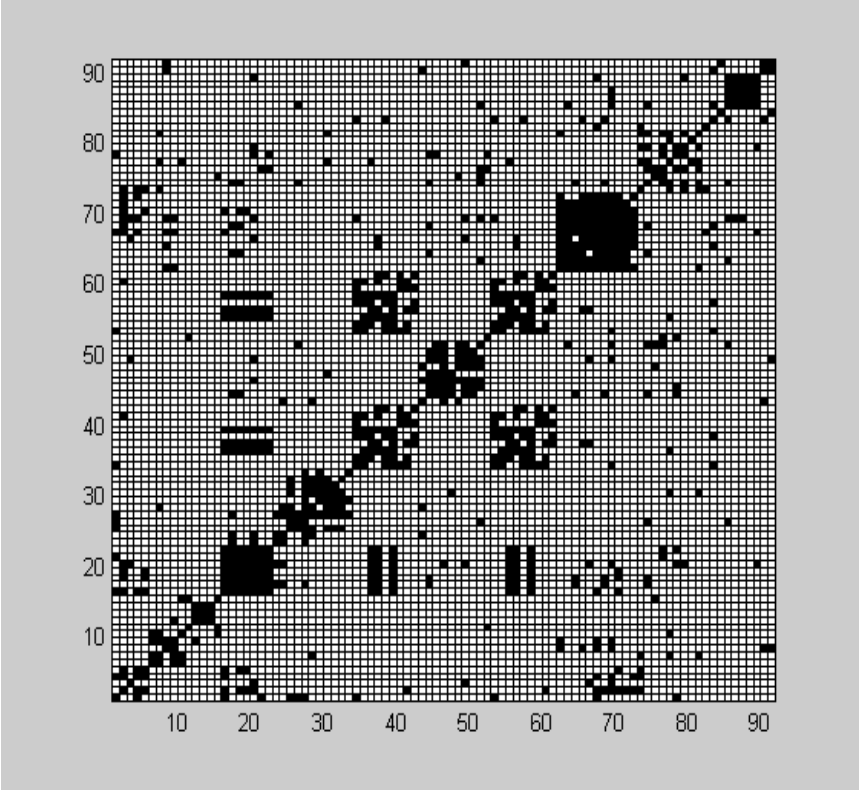

Figure 1

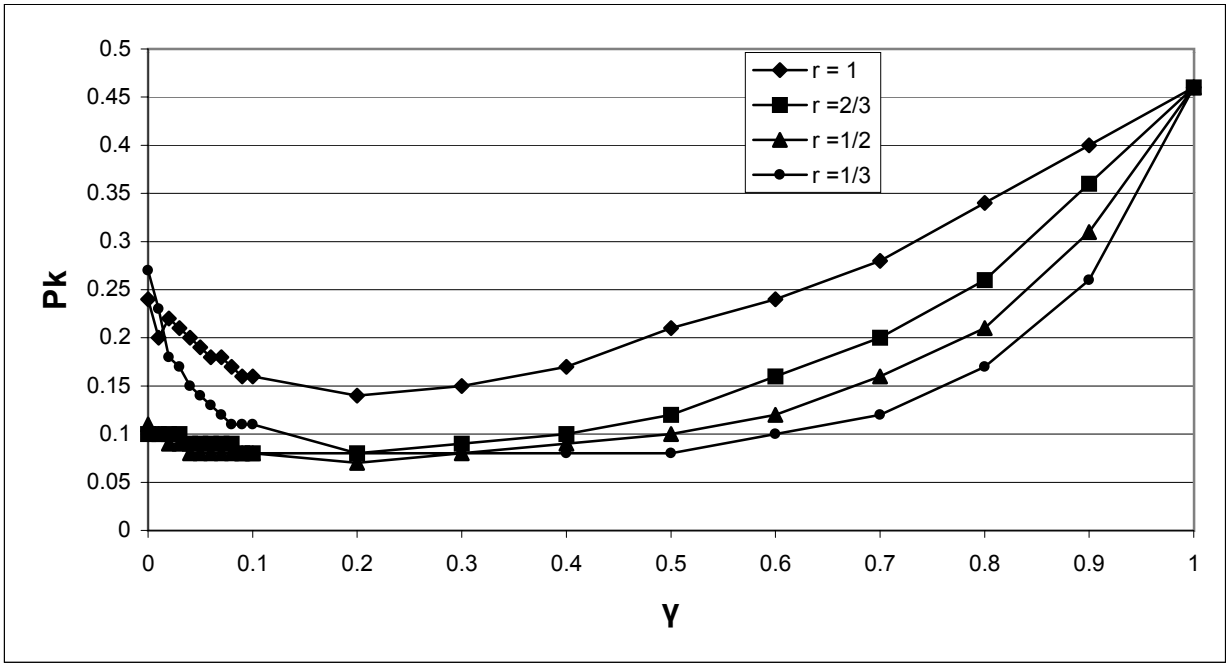

Figure 2:

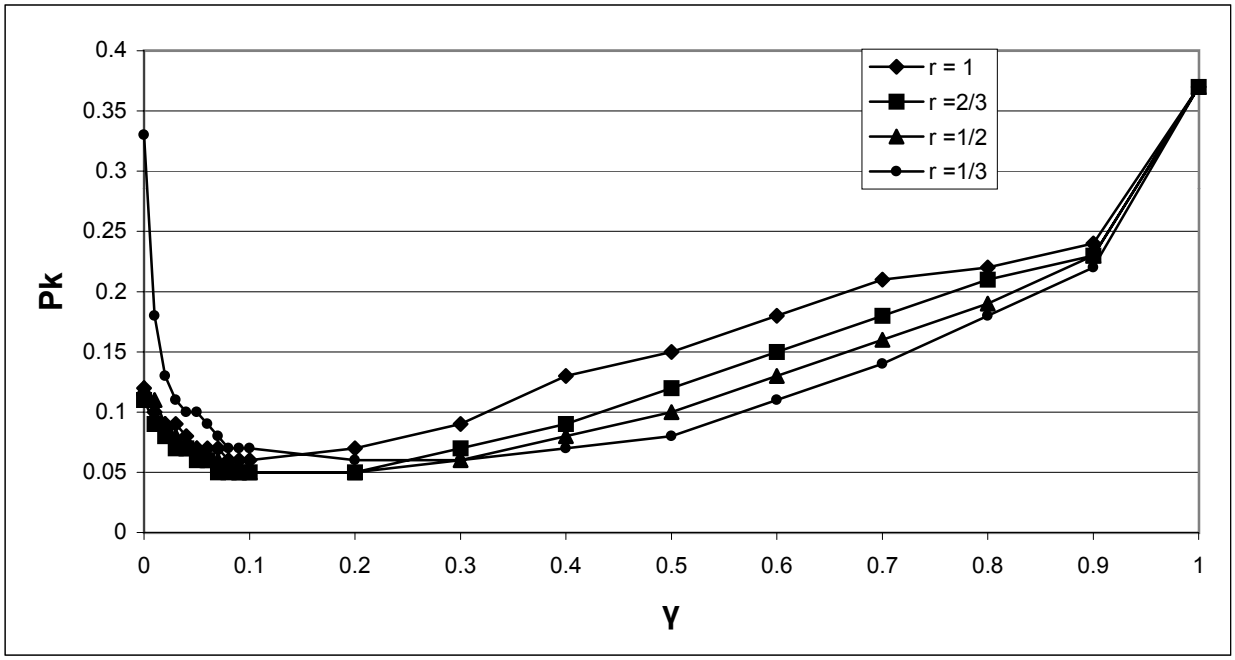

Figure 3

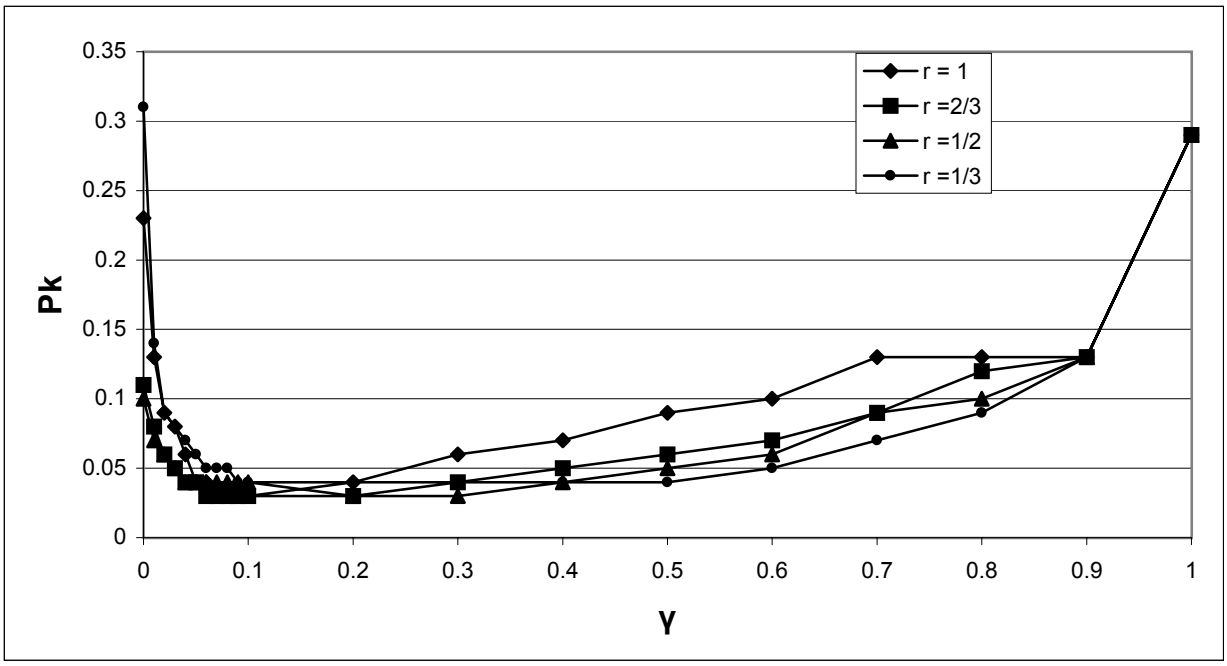

Figure 4

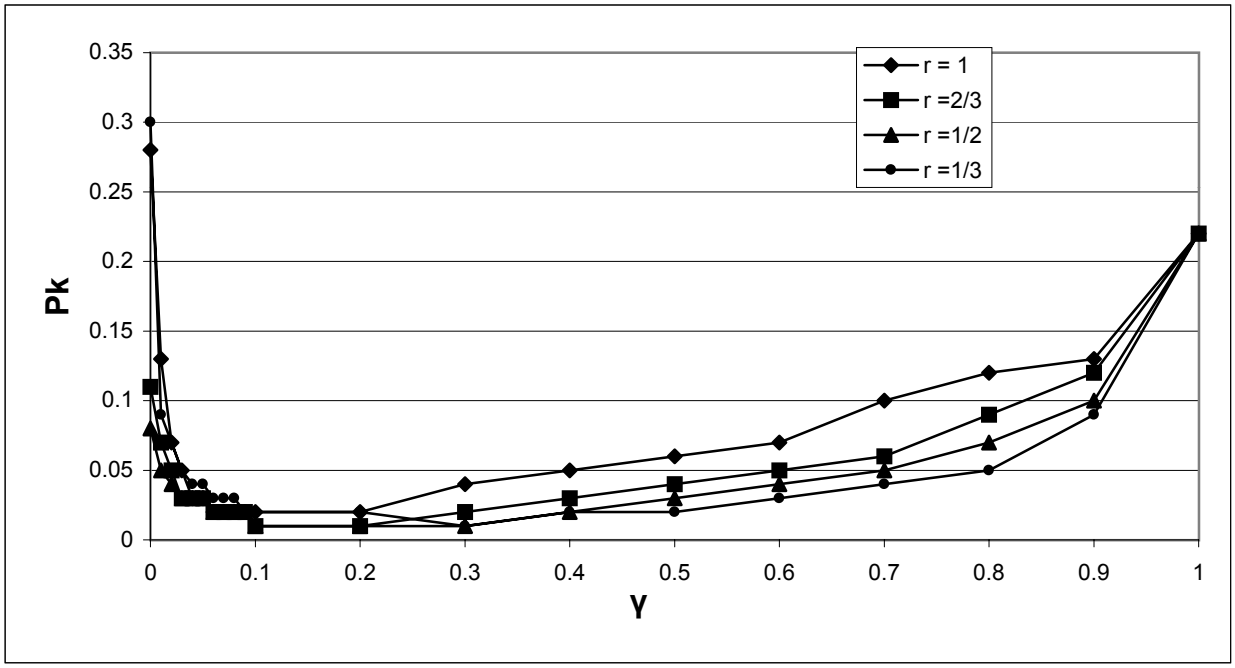

Figure 5

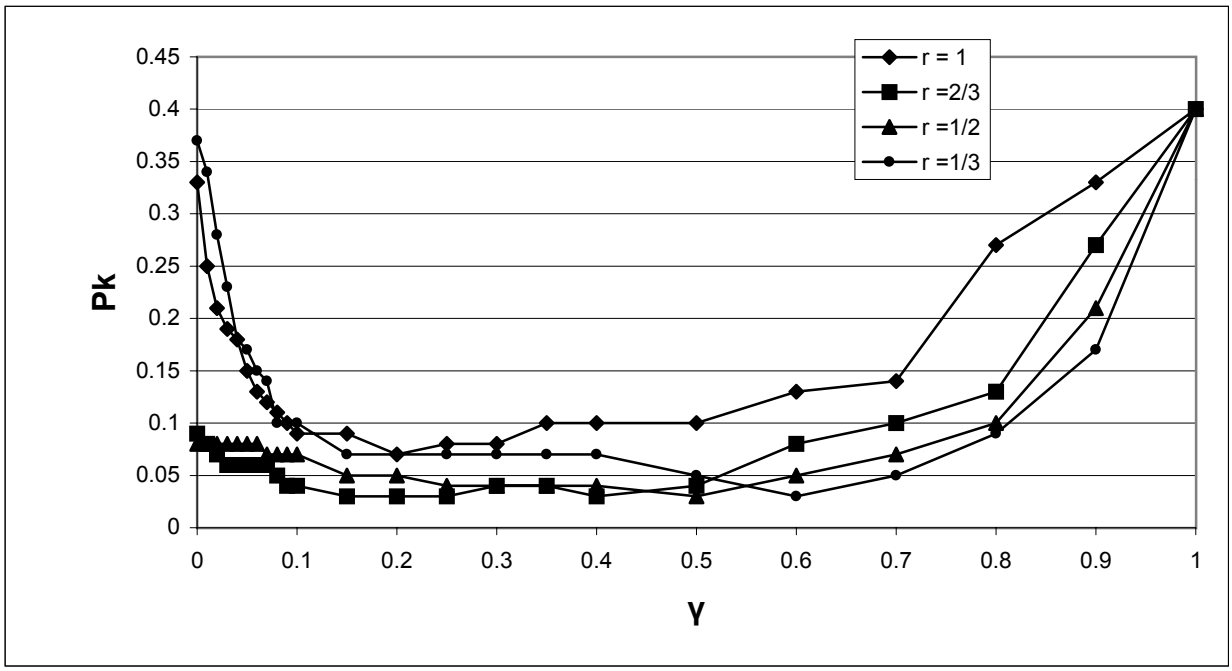

Figure 6

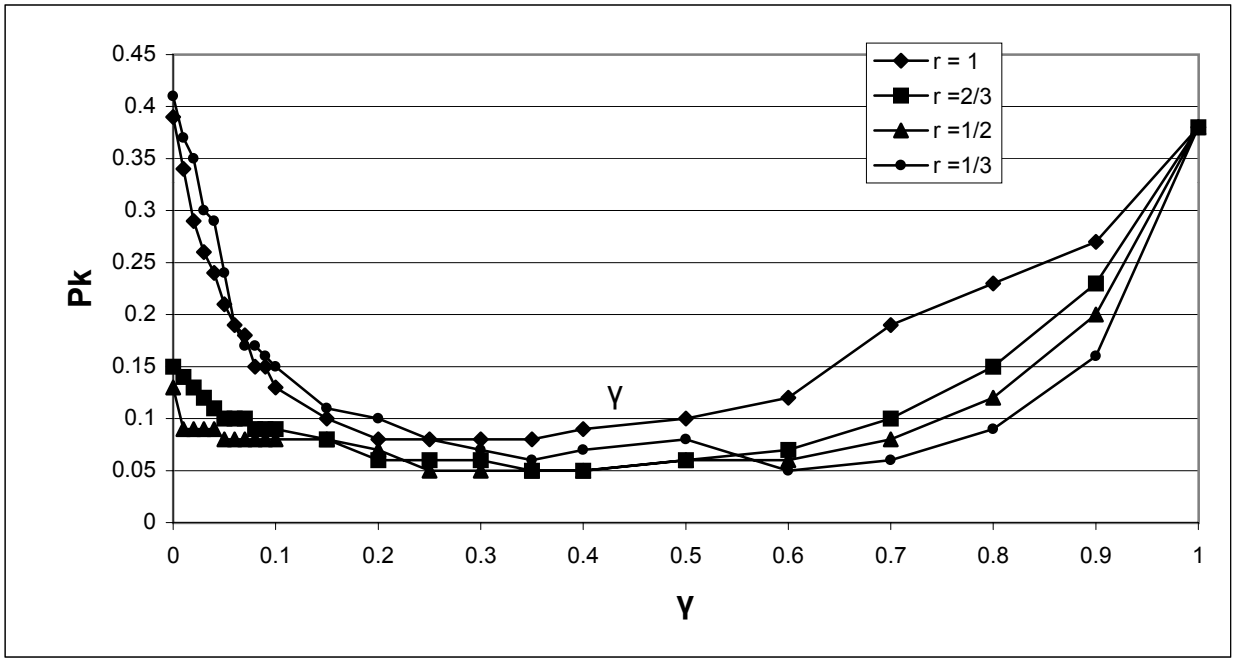

Figure 7

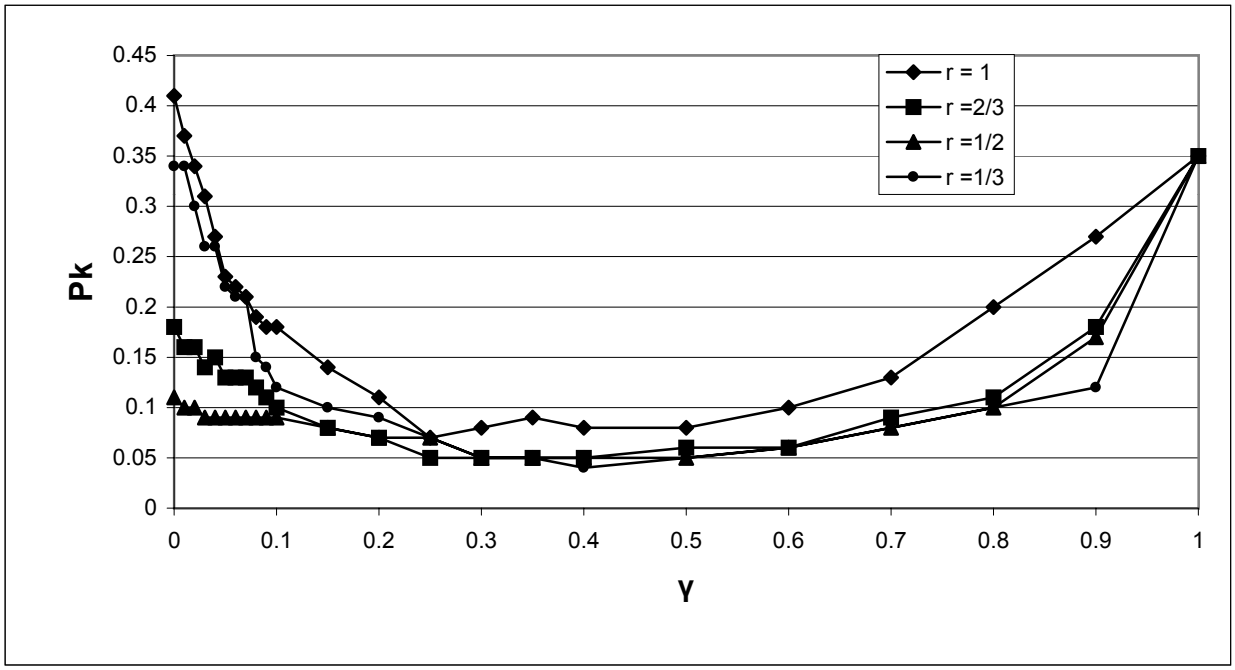

Figure 8

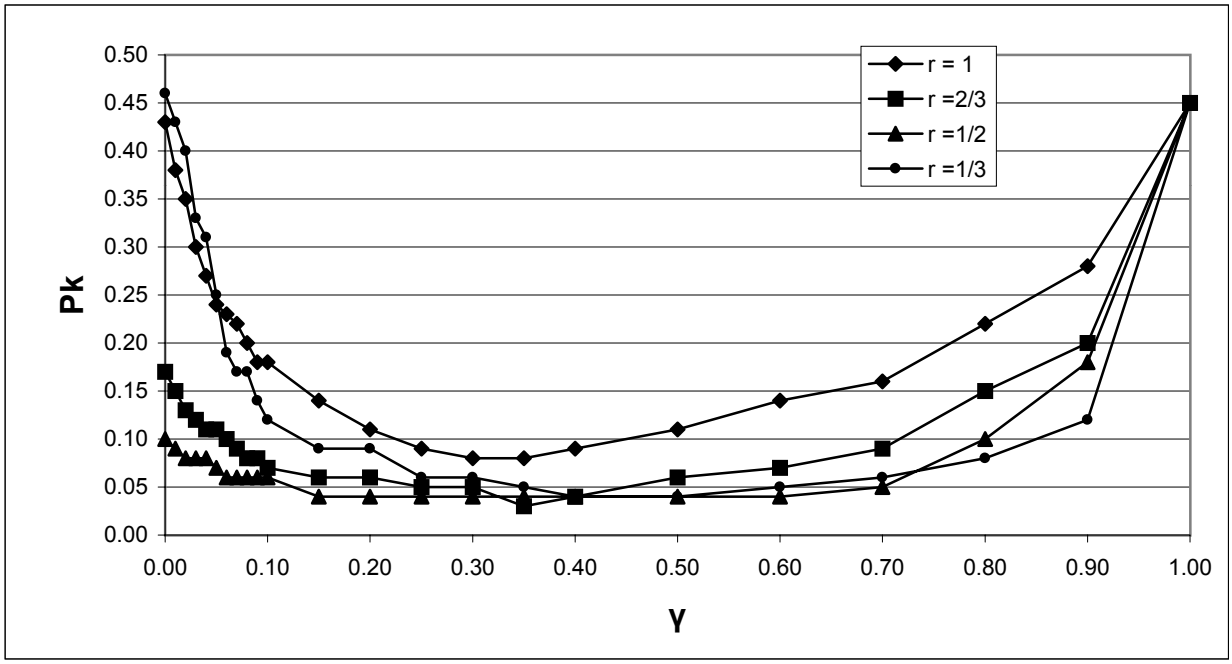
r = 1
r =2/3
r =1/2
r =1/3
Pk
γ
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
0.50
0.00
0.10
0.20
0.30
0.40
0.50
0.60
0.70
0.80
0.90
1.00

Figure 9