An Improved Algorithm for Unsupervised Decomposition of a Multi-Author Document

Chris Giannella

The MITRE Corporation

7515 Colshire Dr., McLean VA 22102 USA

cgiannel@acm.org

Abstract

This paper addresses the problem of unsupervised decomposition of a multi-author text document: identifying the sentences that were written by each author assuming the number of authors is unknown. An approach, BayesAD, is developed for solving this problem: apply a Bayesian segmentation algorithm, followed by a segment clustering algorithm. Results are presented from an empirical comparison between BayesAD and AK, a modified version of an approach published by Akiva and Koppel in 2013.

BayesAD exhibited greater accuracy than AK in all experiments. However, BayesAD has a parameter that needs to be set and which had a non-trivial impact on accuracy. Developing an effective method for eliminating this need would be a fruitful direction for future work. When controlling for topic, the accuracy of BayesAD and AK were, in all but one case, worse than a baseline approach wherein one author was assumed to write all sentences in the input text document. Hence, room for improved solutions exists.

1. Introduction

Authorship analysis is a field of study which aims to infer authorship information from a document or corpus of documents. The field can be divided into several sub-fields depending upon the type of authorship information to be inferred. One sub-field is authorship distinction (or similarity detection), "…the task of grouping documents by authorship when the author of none of those documents is known…" (Layton, Watters, & Dazeley, 2011, p. 98). An interesting variation on problems in this sub-field has received relatively little attention, namely, unsupervised decomposition of a single

multi-author text document: identifying the sentences that were written by each author assuming the number of authors is unknown. Akiva and Koppel argue that an effective solution to this problem, when the number of authors is assumed known, would be of practical interest in a variety of contexts including "…commercial or legal interest, as in the case of contemporary documents, or of academic or cultural interest, as in the case of important historical documents…" (Akiva & Koppel, 2012, p. 205). This argument applies also to the more general version of the problem where the number of authors is unknown.

Formally stated, the unsupervised decomposition of a single multi-author document problem is defined as follows. Given D a multi-author document consisting of |D| sentences: d[1],…,d[|D|], produce a partition[1] $\{C_1,\ldots,C_m\}$ of the sentences matching authorship: d[i] and d[j] are in the same part if and only if d[i] and d[j] were written by the same author. The number of parts, m, is not specified and must be automatically determined. Furthermore, neither writing samples from the authors of D, nor ground truth of any kind is available. For brevity, in the remainder of this paper, "unsupervised decomposition of a single multi-author document" is shortened to "unsupervised authorship decomposition" or "authorship decomposition". The same problem is addressed in (Akiva & Koppel, 2013), except there, the number of authors is assumed known.

This paper develops an approach, BayesAD, for solving the authorship decomposition problem. The approach is summarized in Section 2: first divide D into sub-sequences of consecutive sentences (segments); second cluster the segments to produce the final authorship decomposition. The segmentation[2] algorithm is described in Section 3 and adopts a Bayesian approach combining ideas from (Eisenstein & Barzilay, 2008) and (Utiyama & Isahara, 2001). The segment clustering algorithm is described in Section 4 (with details in the appendix, Section 9) and uses a modified version of the spectral clustering algorithm in (Zelnik-Manor & Perona, 2004).

BayesAD was empirically compared with AK, a modified version of the approach in (Akiva & Koppel, 2013). The details of AK are contained in Section 5 along with the definition of the accuracy metric used to quantify the comparison. The details of the data used and experimental procedure are contained in Section 6. The results of the experiments are contained in Section 7. Related work is discussed in Section 8.

---

[1] A partition $\{C_1,..,C_m\}$ of the sentences in D is defined as follows. Each part C_i is a non-empty subset of sentences in D. Each pair of parts C_i and C_j ($i \neq j$) is disjoint. Every sentence in D is contained in some part.

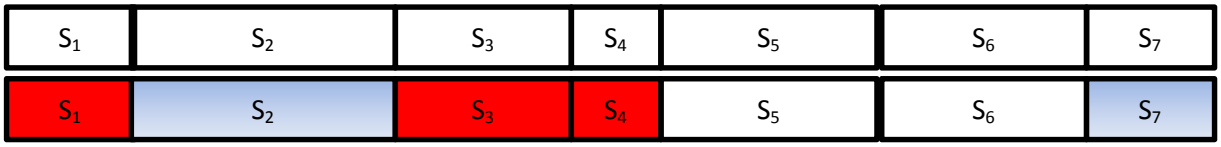
[2] Throughout this paper, the term "segmentation" is used to mean "linear segmentation", as opposed to "hierarchical segmentation".

2. Summary of BayesAD

(i) Apply a segmentation algorithm to divide the sequence of sentences in D into sub-sequences (segments) $S_1,...,S_q$ each consisting of consecutive sentences. The number of segments q is not specified and must be automatically determined. The goal of the algorithm is to produce as few segments as possible while respecting authorship: all sentences in the same segment are written by the same author.

(ii) Apply a clustering algorithm to the segments with the goal of grouping together those whose sentences were written by the same author. Let $SegC_1,...,SegC_m$ denote the resulting segment clusters; the number of segment clusters m must be automatically determined. The final partition $\{C_1,...,C_m\}$ is formed in the natural way: C_i is the set of all sentences that appear in a segment in $SegC_i$.

The figure below illustrates the overall approach. In this example, step (i) produces q=7 segments (top part of the figure). Step (ii) produces m=3 segment clusters as illustrated in solid red $\{S_1,S_3,S_4\}$, shaded blue $\{S_2,S_7\}$, and white $\{S_5,S_6\}$.



3. The Bayesian Segmentation Algorithm

D is assumed to arise according to a stochastic generative model described next. The model is a combination of the models described in (Eisenstein & Barzilay, 2008) and (Utiyama & Isahara, 2001). An algorithm is developed for choosing a segmentation of the sentences in D maximizing the log-joint-likelihood.

3.1 The Stochastic Generative Model

Let |d[i]| denote the number of words[3] in sentence d[i] and W denote the set of all words that appear in any sentence in D. For simplicity, the words in all sentences are denoted by integers 1 to |W|. Let Z denote the set of all segmentations of the sentences in D without restriction on the number of segments present. Given a segmentation z in Z, |z| denotes the number of segments in z. The segmentation is represented as a sequence of integers $0=z(0)<z(1)<...<z(|z|)=|D|$. The j^{th} segment is

[3] A word is simply any consecutive sequence of non-whitespace characters as defined by the Java character class "\s": space, tab, newline, form feed, carriage-return, and vertical tab.

{d[z(j−1)+1],d[z(j−1)+2],…,d[z(j)]}. Let $\theta_0 \in \mathbb{R}_+^{|W|}$ denote a hyper-parameter vector specifying a Dirichlet prior in the following generative model.

First, a segmentation $z \in Z$ is drawn from the prior distribution defined in (Utiyama & Isahara, 2001) which assigns less probability to segmentations with more segments. Next, for each segment (the j^{th}), the parameters θ_j for the word distribution are drawn from a Dirichlet prior with parameters θ_0. The sentences in the j^{th} segment are generated independently. For each, the words are generated independently by drawing each from a categorical distribution with parameters θ_j. More precisely, the generative model proceeds as follows.

1. Choose z with probability $|D|^{-|z|}/\sigma(Z,|D|)$ where $\sigma(Z,|D|)$ is a normalization constant only depending on |D| and Z. This prior distribution is based on the minimum description length principal as discussed in (Utiyama & Isahara, 2001).
2. For j = 1 to |z| do
    a. Choose θ_j according to[4] $Dir(.;\theta_0)$.
    b. For i = z(j−1)+1 to z(j) do
        I. For h = 1 to |d[i]| do
            1. Generate the h^{th} word of the i^{th} sentence of the j^{th} segment according to[5] $Cat(.;\theta_j)$.

The desired segmentation is $z^* = \text{argmax}_{z \in Z}[\log(\Pr(D,z|\theta_0))]$. An algorithm for computing z*, assuming fixed θ_0, is described next.

3.2 Maximizing the Log-Joint-Likelihood, $\log\Pr(D,z|\theta_0)$

Let S(W) denote the set of all probability distributions over W. It follows that:

$$\Pr(D,z|\theta_0) = \Pr(z) \prod_{j=1}^{|z|} \int_{\theta_j \in S(W)} \Pr\left(\{d[i]:i=z(j\text{-}1)+1 \text{ to } z(j)\}|\theta_j\right) \Pr\left(\theta_j|\theta_0\right) d\theta_j .$$

$\Pr(\{d[i]:i=z(j-1)+1 \text{ to } z(j)\}|\theta_j\})$ denotes the probability of generating the j^{th} segment {d[i]:i=z(j−1)+1 to z(j)} given word categorical distribution parameters θ_j.

From the generative model priors, $\Pr(z) = |D|^{-k(z)}/\sigma(Z,|D|)$ and $\Pr(\theta_j|\theta_0) = Dir(\theta_j;\theta_0)$, it follows that (with C not depending on z):

---

[4] $Dir(.;\theta_0)$ denotes a Dirichlet distribution with parameters θ_0. See http://en.wikipedia.org/wiki/Dirichlet_distribution for details.

[5] $Cat(.;\theta_j)$ denotes a categorical distribution with parameters θ_j. See http://en.wikipedia.org/wiki/Categorical_distribution for details.

$$\log\big(\Pr(D,z|\theta_0)\big)=\sum_{j=1}^{|z|}\left[-\log(|D|)+\log\left(\int_{\theta_j\in S(W)}\Pr\big(\{d[i]:i=z(j-1)+1\text{ to }z(j)\}|\theta_j\big)\mathrm{Dir}\big(\theta_j;\theta_0\big)\,d\theta_j\right)\right]+C.$$

Since each term in the above sum corresponds to one and only one segment in z, then $\mathrm{argmax}_{z\in Z}[\log(\Pr(D,z|\theta_0))]$ can be computed using a weighted graph longest path approach like that in (Utiyama & Isahara, 2001). To see how, consider the directed graph with vertices {0,1,..,|D|} and an edge from u to v for all $0\le u<v\le|D|$. Utiyama and Isahara illustrated how each segmentation z in Z corresponds to a path from vertex 0 to |D|, and vice versa. Let Term(u,v) denote the weight on the edge from u to v and be defined as the term in the above sum corresponding segment {d[u+1],…,d[v]}:

$$\mathrm{Term}(u,v)\stackrel{\mathrm{def}}{=}-\log(|D|)+\log\left(\int_{\theta_j\in S(W)}\Pr\big(\{d[i]:i=u+1\text{ to }v\}|\theta_j\big)\,\mathrm{Dir}\big(\theta_j;\theta_0\big)d\theta_j\right).$$

Hence, $\mathrm{argmax}_{z\in Z}[\log(\Pr(D,z|\theta_0))]$ can be computed by applying Dijkstra's algorithm to find largest weight path from vertex 0 to |D|.

3.3 Computing Term(u,v)

Let c([u,v],w) denote the number of times word w appears in the sentences in the segment defined by edge [u,v]: {d[u+1],d[u+2],…,d[v]}. Let θ[u,v] denote the parameters of the word categorical distribution for the segment defined by edge [u,v]. From the generative model independence assumptions, it follows that:

$$\mathrm{Term}(u,v)=-\log(|D|)+\log\left(\int_{\theta_{[u,v]}\in S(W)}\prod_{w=1}^{|W|}\theta_{[u,v]}[w]^{c([u,v],w)}\mathrm{Dir}\big(\theta_{[u,v]};\theta_0\big)d\theta_{[u,v]}\right).$$

Let s(c[u,v]) denote $\Sigma_w c([u,v],w)$. From the definition of the Dirichlet distribution and a property of the Gamma function,[6] it follows that:

$$\mathrm{Term}(u,v)=-\log(|D|)+\sum_{w=1}^{|W|}\sum_{i=0}^{c([u,v],w)-1}\log(\theta_0[w]+i)-\sum_{i=0}^{s(c[u,v])-1}\log(\sigma(\theta_0)+i).$$

4. Segment Clustering

In this section, an algorithm is described for clustering a segmentation $S_1,\ldots,S_q$ of the sentences in D. The clustering algorithm does not require the number of clusters to be

[6] For real x > 0 and integer k ≥ 0, $\log(\Gamma(x)/\Gamma(x+k))=-\sum_{i=0}^{k-1}\log(x+i)$ and $\log(\Gamma(x+k)/\Gamma(x))=\sum_{i=0}^{k-1}\log(x+i)$ where Γ(.) denotes the standard Gamma function, http://en.wikipedia.org/wiki/Gamma_function.

specified. The clustering algorithm is very similar to the one in (Zelnik-Manor & Perona, 2004).

1. Compute qxq similarity matrix, M, whose (i,j) entry is 0 if i = j; otherwise, one minus the Jensen-Shannon divergence[7] between the word frequency distributions over S_i and S_j.
2. Compute qxq normalized similarity matrix M' = $Deg(M)^{-0.5}MDeg(M)^{-0.5}$. Deg(M) is the degree matrix for M: the diagonal matrix whose i^{th} diagonal entry is the sum of the entries in the i^{th} row of M.
3. Compute K*, the desired number of clusters, using an approach based on that in Section 3 of (Zelnik-Manor & Perona, 2004), but with non-trivial differences; see the appendix for details.
4. Compute the dominant K* eigenvectors of M'. Let V denote the qxK* matrix whose i^{th} column is the i^{th} eigenvector. Normalize the rows of V to have Euclidean length one.
5. Apply K*-means clustering to the normalized rows of V with initial centroids chosen according to (Arthur & Vassilvitskii, 2007). S_i and S_j are put into a cluster if and only if the i^{th} and j^{th} normalized rows of V end up in the same K*-means cluster.

5. Experiments: Decomposition Approaches and Accuracy Quantification

5.1 Authorship Decomposition Approaches

BayesAD was compared to a baseline approach, denoted "1Author", which assigns a single author to D. BayesAD was also compared to a modified version of the approach in (Akiva & Koppel, 2013), denoted AK and described below.

1. Divide the sentences in D into sub-sequences (segments) of a fixed number, AK_Segment_Size, of consecutive sentences.[8] The setting of AK_Segment_Size is explained later.
2. For each word in D, count the number of different segments from step 1 in which the word appears. Sort the words in D in decreasing order by this count and identify the top 500 words in the sorted list; call these the 500 most common words. For each segment, build a length 500 binary vector whose i^{th} entry is one (zero) if the i^{th} most common word is (is not) in the segment.
3. Build a segment similarity matrix, M, whose (i,j) entry is zero if i=j, else is the cosine similarity between the binary vectors for the i^{th} and j^{th} segments.

[7] http://en.wikipedia.org/wiki/Jensen%E2%80%93Shannon_divergence
[8] The last segment may have fewer sentences – whatever remains in D.

4. Cluster the segments as done in steps 2-5 of Section 4. The clustering algorithm is different than the one used by Akiva and Koppel. The primary difference is that the number of clusters is unknown and must be automatically determined (Akiva and Koppel assumed the number of authors was known).
5. For each segment cluster, compute the "core" segments as follows. Let Cent denote the cluster centroid. Given segment S, let Row(S) denote the corresponding row for S in the eigenvector matrix V in step 4 of Section 4. Compute $Cent_2(S)$ the second closest centroid to Row(S) in terms of Euclidean distance.[9] Compute CentDist(S) = EuclideanDistance[Cent,row(S)] and CentGap(S) = EuclideanDistance[$Cent_2$(S),row(S)] - EuclideanDistance[Cent,Row(S)]. Compute BestCentDist, the top 80% (rounding down) of the segments in the cluster according to smallest CentDist(.). Compute BestCentGap, the top 80% (rounding down) of the segments in the cluster according to largest CentGap(.). Compute the core segments in the cluster as the intersection between BestCentDist and bestCentGap. If the intersection is empty, then the top segment by CentDist(.) is assigned as the only core segment.
6. Compute FT, the set of all words that appear at least five times in D. Assign a label to each segment, the number of the cluster containing the segment. Represent each segment as a length |FT| feature vector whose i^{th} entry is the number of times the i^{th} word in FT appears in the segment. Using version 2.0.7 of the MALLET open-source library (McCallum, 2002), train a maximum entropy classifier on the labeled feature vectors. Akiva and Koppel used a SVM, but for AK a maximum entropy classifier was used since, unlike an SVM, it directly provides label probability distributions.
7. For each sentence in D, compute its length |FT| feature vector, then apply the maximum entropy classifier and compute the largest label probability. For each sentence whose largest label probability was among the top 25%, assign the sentence its largest probability label.
8. For each sentence s in D which was not assigned a label in step 7, find s', the closest sentence[10] to s in D which was assigned a label in step 7, and assign that label to s. Here, "closest" is based on the number of sentences in D between s and s'. Akiva and Koppel use a different procedure to assign labels to sentences. Their procedure utilizes properties of the boundary of the SVM produced during their step 6.

5.2 Quantifying Authorship Decomposition Accuracy

[9] By virtue of the way K-means clustering works, Cent is the closest centroid to Row(S).

[10] If there are two sentences each assigned a label in step 5, each equally close to s, and with no closer sentence with an assigned label, then the label assigned to s is randomly chosen between the labels of the two sentences.

An authorship decomposition, $\{C_1,\ldots,C_m\}$, is a partition of the sentences in D. Ground truth authorship forms another partition, denoted $\{\Gamma_1,\ldots,\Gamma_t\}$. Let n_{ij} denote $|C_i \cap \Gamma_j|$. In (Akiva & Koppel, 2013), the accuracy of $\{C_1,\ldots,C_m\}$ was quantified by using purity, a common extrinsic clustering validation index.

$$|D|^{-1} \sum_{i=1}^{m} \max_{1 \le j \le t} \{n_{ij}\}$$

A weakness of this index was pointed out on (Manning, Raghavan, & Schutze, 2008, p. 357): "High purity is easy to achieve when the number of clusters is large – in particular, purity is 1 if each document gets its own cluster." This is not a problem in Akiva and Koppel's setting where the number of authors (t) is assumed known. However, with this assumption dropped, this weakness is a problem, hence, purity is not used in this paper. Instead, the extrinsic clustering validation index in (Akiva & Koppel, 2012) is used and is referred to as matching accuracy.

To motivate the definition of matching accuracy, it is useful to first consider a simpler index that quantifies the accuracy of $\{C_1,\ldots,C_m\}$ as follows. Assign each sentence in D proposed label i and ground truth label j if the sentence is in C_i and Γ_j. Compute standard classification accuracy:

$$|D|^{-1} \sum_{i=1}^{\min\{m,t\}} n_{ii} .$$

This index is flawed, however, as a renumbering of the parts in $\{C_1,\ldots,C_m\}$ or $\{\Gamma_1,\ldots,\Gamma_t\}$ could cause the standard classification accuracy to change. The accuracy should not be affected by any such renumbering. To remedy this flaw, maximum classification accuracy is chosen over all one-to-one mappings between part numbers.

If $t \ge m$, then let $\Delta(t \ge m)$ denote the set of all one-to-one mappings from $\{1,\ldots,m\}$ into $\{1,\ldots,t\}$ and, given δ in $\Delta(t \ge m)$, let

$$\mathrm{MatchAcc}_{t \ge m}(\delta) \stackrel{\mathrm{def}}{=} |D|^{-1} \sum_{i=1}^{m} n_{i\delta(i)} .$$

If $m > t$, then let $\Delta(m > t)$ denote the set of all one-to-one mappings from $\{1,\ldots,t\}$ into $\{1,\ldots,m\}$ and, given δ in $\Delta(m > t)$, let

$$\mathrm{MatchAcc}_{m > t}(\delta) \stackrel{\mathrm{def}}{=} |D|^{-1} \sum_{i=1}^{t} n_{\delta(i)i} .$$

The matching accuracy of $\{C_1,\ldots,C_m\}$, is defined as

$$\begin{cases} \max_{\delta\in\Delta(t\geq m)}\{\text{MatchAcc}_{t\geq m}(\delta)\} & \text{if } t\geq m \\ \max_{\delta\in\Delta(m>t)}\{\text{MatchAcc}_{m>t}(\delta)\} & \text{if } m>t. \end{cases}$$

6. Experiments – Data and Procedures

6.1 Data

Upon request, Navot Akiva provided two corpora which appeared to be the same as ones used in (Akiva & Koppel, 2013): BP-Blog, NYT-Columnists.

The BP-Blog corpus is a portion of the "Becker-Posner Blog"[11] which consists of blog posts by Gary Becker and Richard Posner. This corpus was preprocessed as follows. Blog posts pertaining to six topics were manually selected, sentence segmented, their title lines (*e.g.* "Comment on Tort Reform-BECKER") removed, and concatenated to form six multi-author documents. Each multi-author document pertaining to one topic and has alternating authorship; see Table 1.

Topic	Author Order and Number of Sentences per Post
Tort Reform (TR)	Posner (29), Becker (31), Posner (24)
Profiling (Pro)	Becker (35), Posner (19), Becker (21)
Tenure (Ten)	Posner (73), Becker (36), Posner (33), Becker (19)
Traffic Congestion (TC)	Becker (57), Posner (33), Becker (20)
Microfinance (Mic)	Posner (51), Becker (37), Posner (44), Becker (33)
Senate Filibuster (SF)	Posner (39), Becker (26), Posner (28), Becker (24)

Table 1: The seven multi-author documents created from the BP-Blog corpus.

Many of the posts are direct responses to previous posts and contain sentences directly indicating the authorship of the previous post, *e.g.* "As Becker explains, a driver does not consider the effect of his driving on the other users of the road, but only on himself." In these sentences, 14 in total, "Becker" or "Posner" was replaced with "XXXX". Finally, the six multi-author documents were tokenized using the Stanford English core NLP tokenizer (version 1.3.4) with the default settings.[12]

Six experiments were performed, one for each multi-author document. In each experiment: 1Author was run once and its matching accuracy was computed; BayesAD and AK were run 500 times and their mean matching accuracies and 0.95 confidence

[11] http://www.becker-posner-blog.com
[12] http://nlp.stanford.edu/software/corenlp.shtml

intervals[13] were computed. BayesAD and AK were run multiple times because they are non-deterministic, unlike 1Author. The clustering algorithm used by BayesAD and AK involves the non-deterministic setting of initial K*-means centroids using the technique in (Arthur & Vassilvitskii, 2007).

In all experiments, the AK_Segment_Size parameter was set to 15 to be slightly smaller than all author run lengths (an author run is a consecutive sequence of sentences written by the same author).

The NYT-Columnists corpus is a collection of opinion pieces written by four New York Times columnists, see Table 2. The corpus appeared to be sentence segmented and tokenized. This was confirmed by N. Akiva via email.

Columnist Name	Number of Opinion Pieces	Total Number of Sentences
Gail Collins	273	11327
Maureen Dowd	299	11660
Paul Krugman	331	12634
Thomas Friedman	279	11230

Table 2: Statistics regarding the NYT-Columnists corpus.

For each columnist, sequence of sentences was formed by concatenating the columnist's pieces in the order they appeared in the original corpus. Each of these sequences is referred to as a "columnist's sequence". Unlike the BP-Blog corpus, the experiments performed using the NYT-Columnists corpus are designed to examine the impact of author run length and number of runs per author on decomposition approach accuracy. As such, a more complex experimental procedure is used.

6.2 Experimental Procedure Using the NYT-Columnists Corpus

An experiment involved two parameters meanARL and numRperA: the mean author run length and number of runs per author. An experiment consisted of 500 trials, during each: a multi-author document D is produced, the authorship decomposition approaches are applied, and matching accuracy is computed for each approach. The procedure for producing multi-author documents guarantees that each contains exactly numRperA runs of consecutive sentences from each columnist's sequence, concatenated in random order (possibly concatenating two runs from the same columnist). In detail, each trial proceeds as follows.

[13] The t-test was used to compute the 0.95 confidence intervals.

For i in {0,1,2,3}, let numSentences(i) denote the number of sentences in the i^{th} columnist's sequence (the "Total Number of Sentences" in Table 2).

1. For trial = 1 to 500, do set D to empty and:
    a. For i = 0 to 3 do
        A. Choose startSentence(i) uniformly from {0,1,..,(numSentences(i)-4*numRperA*meanARL)}. Discard the first startSentence(i) sentences from the i^{th} column's sequence.
    b. Choose a random permutation P of {0,1,2,...,(4*numRperA-1)}.[14]
    c. For j = 0 to (4*numRperA-1) do
        A. Choose a number from an exponential distribution[15] with mean meanARL and round to the nearest integer, denoted by ChunkSize(j).
        B. Compute i=[P(j)mod4] and choose the first ChunkSize(j) sentences (or as many as possible) from the i^{th} columnist's sequence; append these to the end of D; discard these from the columnist's sequence.
    d. Apply BayesAD, AK, 1Author to D computing the matching accuracy for each.
2. For each authorship decomposition approach, compute the mean and 0.95 confidence interval[16] over the 500 accuracies for the approach.

In all experiments and all trials, the AK_Segment_Size parameter was set to min{40,ln(2)meanARL}, based on the following statement on (Akiva & Koppel, 2012, p. 207):[17] "Results aren't very sensitive to chunk size, as long as chunks are smaller than the median single-author run."

The meanARL parameter controls the typical number of consecutive sentences from the same author (run length) in D. The numRperA parameter controls the number of runs from each author appearing in D. A smaller value of meanARL or larger value of numRperA tends to produce a more difficult authorship decomposition problem.

Since the opinion pieces cover a wide variety of topics, D tends to contain multiple topics with each author transition tending to occur simultaneously with a topic transition. This is in marked contrast with the six multi-author documents produced from the BP-Blog corpus that are guaranteed to each pertain to a single topic.

7. Results

[14] The "Knuth Shuffle" was used: http://en.wikipedia.org/wiki/Fisher%E2%80%93Yates_shuffle
[15] http://en.wikipedia.org/wiki/Exponential_distribution
[16] The t-test was used to compute the 0.95 confidence interval.
[17] AK_Segment_Size was set to 40 in all experiments reported in (Akiva & Koppel, 2012). Each author chunk was drawn from an exponential distribution with median ln(2)meanARL.

7.1 Setting θ_0

A series of experiments were run on the NYT-Columnist corpus with meanARL set to 25, numRperA set to 2, and θ_0 varied from 0.025 to 1. The mean matching accuracy of BayesAD ranged from 0.416 to 0.6, achieving its maximum near θ_0=0.1. An "EM-style" approach similar[18] to that in Section 3.4 of (Eisenstein & Barzilay, 2008) was implemented: alternately choose z* and θ_0, each maximizing the log-joint-likelihood while keeping the other fixed. However the accuracies produced were considerably lower than those observed for θ_0=0.1.

In all subsequent experiments, involving the NYT-Corpus and the BP-Blog corpus, θ_0 was fixed at 0.1.

7.2 Controlling for Topic

[18] As implemented for this paper, all components of θ_0 were forced to be the same; Eisenstein and Barzilay's approach does not require this.

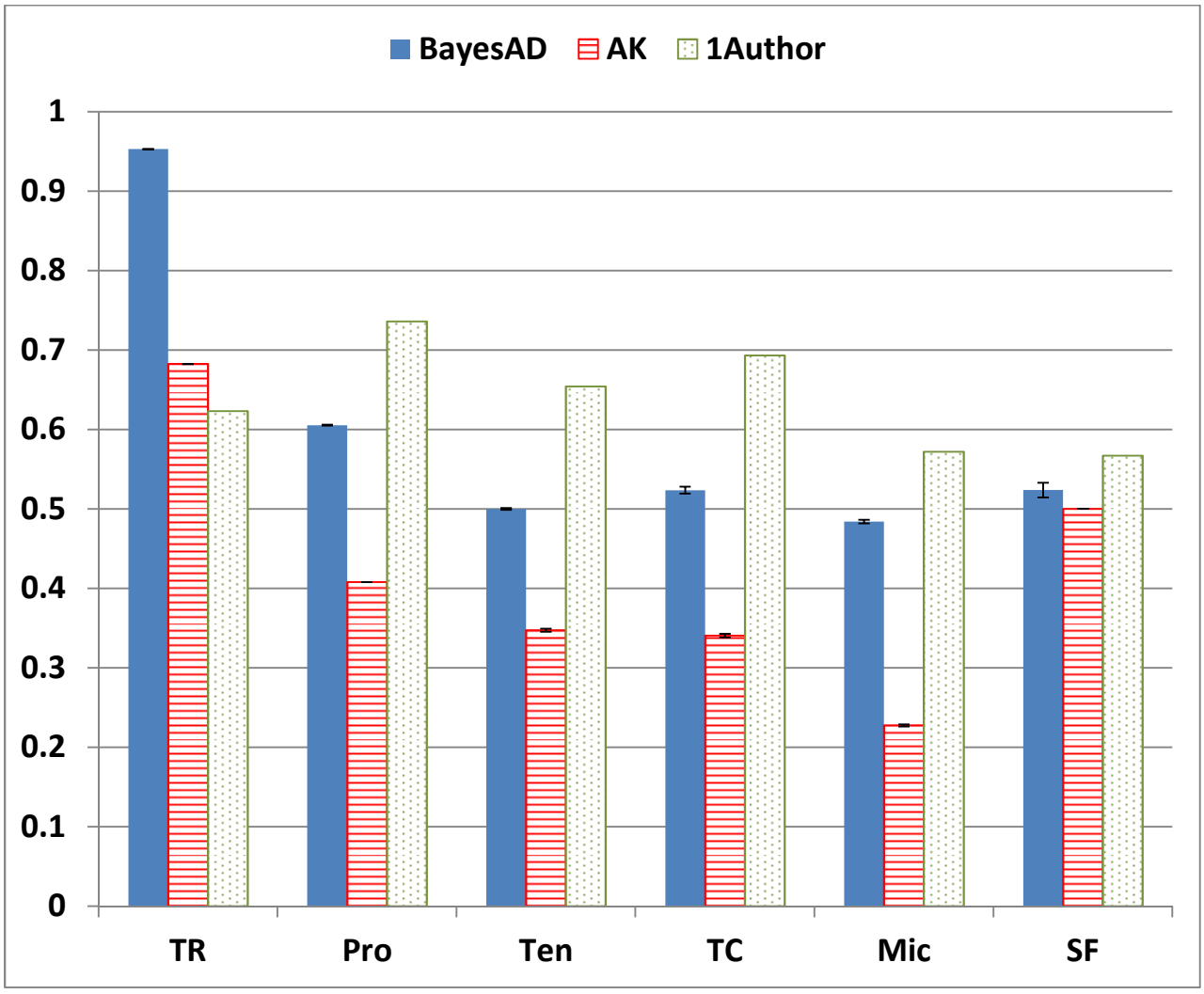


Figure 1: Results of the experiments using the BP-Blog corpus. The x-axis depicts topic (see Table 1), the y-axis depicts matching accuracy, the data bars depict mean matching accuracies, and the error bars for BayesAD and AK depict 0.95 confidence intervals. In many cases the confidence intervals are quite small and are not easily seen in the figure.

7.3 Controlling for Author Run Length and Number of Runs Per Author

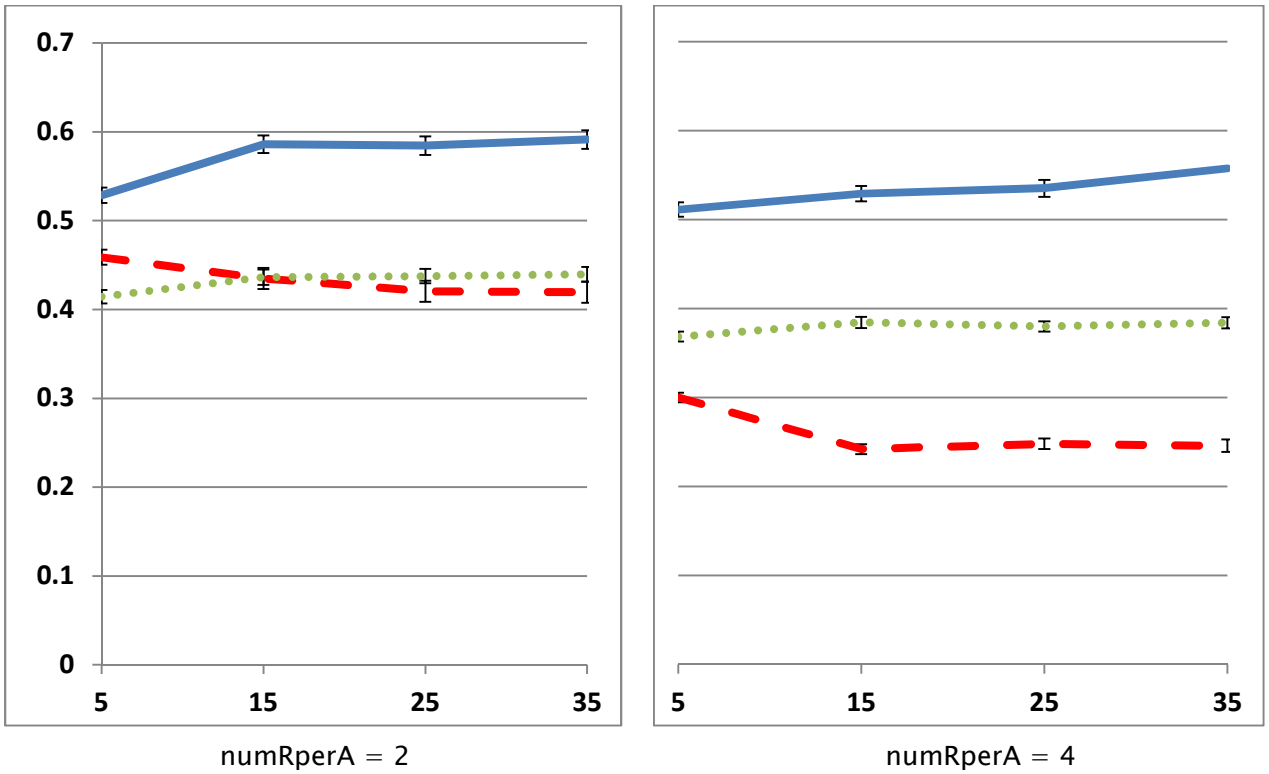


Figure 2: Results of the experiments using the NYT-Columnists corpus. In both charts: the x-axes depict meanARL, the y-axes depict matching accuracy, the graph data points depict mean matching accuracies, and the error bars depict 0.95 confidence intervals.

7.4 Discussion

As seen in Figures 1 and 2, BayesAD had greater accuracy that AK for all topics and all examined values of mean author run length (meanARL) and number of runs per author (numRperA). For example:

- in the experiment involving the "Tort Reform" document, BayesAD produced a 39% larger mean accuracy than AK;
- in the NYT-Columnist corpus experiment with meanARL=35 and numRperA=4, BayesAD produced a 126% larger mean accuracy than AK.

The overall difficulty of the authorship decomposition problem, when topic is controlled, is evident. As seen in Figure 1, the one author baseline approach, 1Author, had greater accuracy than BayesAD and AK on five of six topics.

To better understand the observed accuracy advantage of BayesAD over AK, both approaches were modified to work when the number of authors is assumed known. Step 3 was modified in the segment clustering algorithm (Section 4) to set K* to the known number of authors. The same experiments were run using the BP-Blog corpus

and the accuracy advantage of BayesAD was found to largely go away. This is due to the fact that AK relies more heavily on the clustering algorithm than BayesAD, hence, choosing the wrong number of clusters effects AK more.

Conclusions:

- The author-decomposition approach developed in this paper, BayesAD, exhibited greater accuracy than its leading competitor, AK, in all experiments.
- However, the accuracy of BayesAD was sensitive to the setting of its parameter, θ_0. A method for eliminating the need to manually set this parameter reported in the literature was implemented and yielded no success. Developing an effective method for eliminating this need would be a fruitful direction for future work.
- The authorship decomposition problem is challenging and much room for improved solutions exists. Indeed, when controlling for topic, the accuracy of BayesAD (and AK) was, in all but one case, worse than a simple, one author, baseline approach.

8. Related Work

8.1 Unsupervised Text Segmentation by Topic

Many researchers have addressed the problem of dividing, in unsupervised fashion, a text document into sub-sequences of consecutive sentences or paragraphs (segments) with the goal of producing as few segments as possible while respecting topic. Step (i) in Section 2 addresses the analogous problem with authorship.

Some researchers have adopted a probabilistic viewpoint and developed algorithms for choosing a maximum-likelihood segmentation based on various modeling assumptions. The Bayesian segmentation algorithm in Section 3 directly applies some of these modeling ideas to text segmentation by authorship. Specifically, the algorithm extends the approach in (Eisenstein & Barzilay, 2008) allowing the number of segments to be unspecified. The algorithm combines the segmentation probability model of Eisenstein with the non-uniform prior on segmentations from (Utiyama & Isahara, 2001). Misra *et al.* (Misra, Yvon, Cappe, & Jose, 2011) adopt a similar approach and use a segment prior similar to that of Utiyama, but consider segmentation probabilities based on latent Dirichlet allocation and multinomial mixture models. The Bayesian segmentation algorithm in Section 3 could be replaced with Misra's algorithm. Examining this idea is left for future work.

Other researchers have adopted a variety of other approaches, for example: peak finding in a lexical cohesion curve (Hearst, 1997), minimization of an ad-hoc segmentation cost function (Kehagias, Pavlina, & Petridis, 2003), converting the text segmentation problem to one of image segmentation then applying techniques from image processing (Ji & Zha, 2003), and using affinity propagation in factor graphs (Kazantseva & Szpakowicz, 2011).

8.2 Intrinsic Plagiarism Detection

Some researchers have focused on developing computational approaches to detect plagiarism in a given text document when no reference documents are provided. Their key supposition was that changes in writing style are indicative of a change in authorship. Stamatatos (Stamatatos, 2009) developed an approach centered on a "style change function" based on character n-grams which quantified the stylistic difference between the writing in a window of fixed size to the left and right of a fixed point in a document. The approach built a style curve by sliding the style function across the document, then, based on variance and the presence of peaks in the curve, determined if the document contained plagiarized passages and identified them. A modest modification of this approach could be used in place of the Bayesian segmentation algorithm in Section 3. Examining this idea is left to future work.

Stein *et al.* (Stein, Lipka, & Prettenhofer, 2011) developed an approach which started by decomposing the given document in sections of uniform length and identifying outliers based on a variety of stylometric features. The non-outlying sections were presumed to have been written by a single author and "unmasking" (Koppel, Schler, & Bonchek-Dokow, 2007) was applied to decide if that author also wrote all the outlying sections.

8.3 Authorship Analysis

The application of statistical and computational methods to problems in authorship analysis has been the focus of much study. Koppel et al. (Koppel, Schler, & Argamon, 2009) surveyed this line of work,[19] focused on three specific types of problems, and discussed how machine learning methods can be applied to those problems.

Layton *et al.* (Layton, Watters, & Dazeley, 2011) addressed the problem of authorship distinction: cluster a batch of documents (the number of clusters is not specified) with the goal that for any pair of documents, the documents are in the same cluster if and only if the documents were written by the same author. The document clustering

[19] Koppel *et al.* refer to authorship analysis as "authorship attribution".

algorithm Layton developed could be used in place of the segment clustering algorithm in Step (ii) of Section 2. Doing so is left to future work.

Graham *et al.* (Graham, Hirst, & Marthi, 2005) developed an approach for segmenting text documents by identifying paragraph breaks where the writing style changes significantly. Their approach is supervised in that it used a training set of documents in which the significant style change points are known. A neural network was trained to classify consecutive paragraphs in terms of whether or not the first is significantly different in style than the second. Non-training documents were segmented by applying the classifier to each pair of consecutive paragraphs in the documents. Due to its fundamentally supervised nature, Graham et al.'s approach is not applicable to the unsupervised author segmentation problem addressed in this paper.

Akiva and Koppel (Akiva & Koppel, 2013) define a close variant of unsupervised author segmentation problem. In their definition, the number of authors is assumed known, but that assumption is dropped in this paper. Akiva and Koppel's approach was modified to work after dropping this assumption, details are contained in Section 5. The modified approach was compared to BayesAD, results are contained in Section 7. To the author's knowledge, (Akiva & Koppel, 2013) is the most closely related work to this paper.

9. Appendix – Computing K*, the Desired Number of Clusters

Some notation is needed before discussing the computation of K*. Given an arbitrary matrix B, B_{ij} denotes the (i,j) entry in B. $B_{i.}$ denotes the i^{th} row of B and $||B_{i.}||$ denotes the two-norm of that row. B^T denotes the transpose matrix of B, *i.e.* $(B^T)_{ij} = B_{ji}$ for all i,j. B is orthogonal if B^TB is the identity matrix. $||B||_F$ denotes the Frobenius norm.[20] Assume B is square, with the same number of rows as columns. Then B^{-1} denotes the inverse matrix of B, *i.e.* B^{-1} is the square matrix such that $B^{-1}B$ is the identity matrix. The reader is referred to (Strang, 2005) for a discussion of the fundamental definitions and properties of the eigenvectors and eigenvalues.

To understand how K* can be computed from the qxq normalized similarity matrix M', it is useful to consider the following hypothetical assumption.

9.1 Hypothetical Assumption

Assume the segments cluster cleanly with respect to their similarities in M: $M_{ij}=0$ if S_i and S_j are in a different cluster, otherwise $M_{ij}=1$. The columns and their corresponding

[20] http://en.wikipedia.org/wiki/Matrix_norm

rows of M can be reordered to produce a block-diagonal matrix where each block corresponds to a cluster. The same column and row reordering of the normalized matrix M' produces the same block-diagonal structure.

Example: K*=3 and each cluster contains three segments (so q=9). M', after column and row reordering, is

M'_{11}	M'_{12}	M'_{13}	0	0	0	0	0	0
M'_{12}	M'_{22}	M'_{23}	0	0	0	0	0	0
M'_{13}	M'_{23}	M'_{33}	0	0	0	0	0	0
0	0	0	M'_{44}	M'_{45}	M'_{46}	0	0	0
0	0	0	M'_{45}	M'_{55}	M'_{56}	0	0	0
0	0	0	M'_{46}	M'_{56}	M'_{66}	0	0	0
0	0	0	0	0	0	M'_{77}	M'_{78}	M'_{79}
0	0	0	0	0	0	M'_{78}	M'_{88}	M'_{89}
0	0	0	0	0	0	M'_{79}	M'_{89}	M'_{99}

Key to recovering K* is the following fact, similar to Proposition 2 in (von Luxburg, 2007). There exists a qxK* orthogonal matrix X whose columns form a basis for the eigenspace associated with the largest eigenvalue of M', and each row of X contains all zeros except a single entry.[21]

For a given K', let X' denote a qxK' matrix X' produced by a standard eigensolver: X' is orthogonal and its columns are the K' dominant eigenvectors of M'. If K'=K*, then, like X, the columns of X' form a basis for the eigenspace associated with the largest eigenvalue of M'. However, there is no guarantee that X' will equal X. Nonetheless, some K'xK' orthogonal matrix O must exist such that X'O equals X, hence all rows of X'O contain all zeros except a single entry. Moreover, if K'>K*, then for any O, X'O will contain a row with more than one non-zero entry.

Let Θ denote an error function mapping K'xK' orthogonal matrices, O, to a number quantifying the extent to which all rows of X'O contain all zeros except a single entry. Formally[22]

$$\Theta(O) \stackrel{\text{def}}{=} \sum_{i=1}^{q} \begin{cases} 1 - \text{SimpsonIndex}\left[\frac{(X'O)_{i1}^{\,2}}{\sum_{j=1}^{K'}(X'O)_{ij}^{\,2}}, \ldots, \frac{(X'O)_{iK'}^{\,2}}{\sum_{j=1}^{K'}(X'O)_{ij}^{\,2}}\right] & \text{if } \sum_{j=1}^{K'}(X'O)_{ij}^{\,2} \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

[21] More specifically: segment S_j is in the i^{th} cluster if and only if the j^{th} row of X contains all zeros except in the entry for column i.

[22] The Simpson Index is defined here: http://en.wikipedia.org/wiki/Diversity_index

$$= \sum_{i=1}^{q} \begin{cases} 1 - \left[\frac{1}{\|X'_{i.}\|^4}\right] \sum_{h=1}^{K'} \left(\sum_{j=1}^{K'} X'_{ij} O_{jh} \right)^4 & \text{if } \|X'_{i.}\|^4 \neq 0 \\ 0 & \text{otherwise.} \end{cases}$$

The last equality follows from the definition of the Simpson Index and the fact that O is orthogonal. Θ(O) equals its minimum of zero exactly when all rows of X'O contain all zeros except at most a single entry. Let $\min_{K'}(\Theta)$ denote the minimum of Θ over all K'xK' orthogonal matrices. Reasoning in the previous paragraph implies the approach to recovering K*, namely, find the largest value of K' such that minimizes $\min_{K'}(\Theta)$.

Computing $\min_{K'}(\Theta)$ amounts to solving a constrained optimization problem.

$$\text{minimize}\{\Theta(O): O \text{ any } K'xK' \text{ real-valued matrix}\}$$
$$\text{subject to: } O \text{ is orthogonal.}$$

Constrained optimization problems with such constraints have been addressed in the literature, in particular (Wen & Yin, 2013). The gradient-descent search therein will find a critical point of Θ and guarantee that the matrix produced at each step is orthogonal. However, a critical point of Θ is not guaranteed to be a local minimum, let alone a global minimum. A simple way to deal with this problem is by repeating the gradient-descent search ten times from randomly chosen starting points and picking the search termination matrix, O(terminate), which minimizes Θ over all termination matrices.

9.2 The General Case

If the hypothetical assumption is dropped that the segments cluster cleanly with respect to their similarities, then M' will enjoy only approximately the block structure described earlier. Nonetheless, the approach described in the hypothetical case is still applied. To be clear, the algorithm for finding K* in the general case is as follows.

I. For K' = 2 to q-1, do
   a. Use a standard eigensolver to produce qxK' orthogonal matrix X' whose columns are the K' dominant eigenvectors of M'.
   b. Generate 10 matrices randomly from the space of all K'xK' orthogonal matrices. Each of these will serve as a starting point to a gradient-descent search as described next. Each search will find an orthogonal matrix which is approximately a critical point of Θ. Let err(K') denote the minimum value of Θ over all these 10 approximate critical points.

II. Return the largest $2 \leq K^* \leq q-1$ for which $err(K^*) = \min\{err(K'): 2 \leq K' \leq q-1\}$.

9.3 Gradient-Descent on Θ Under Orthogonality Constraints

Let B denote a K'xK' matrix and G[B] denote a K'xK' matrix whose (a,b) entry is the partial derivative of Θ with respect to O_{ab} evaluated at B:

$$\frac{\partial \Theta}{\partial O_{ab}}[B] = \sum_{i=1}^{q} \begin{cases} -\left[\frac{4}{\|X'_{i.}\|^4}\right] \sum_{h=1}^{K'} \left(\sum_{j=1}^{K'} X'_{ij} O_{jh} \right)^3 \sum_{j=1}^{K'} X'_{ij} \frac{\partial O_{jh}}{\partial O_{ab}}[B] & \text{if } \|X'_{i.}\|^4 \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

$$= \sum_{i=1}^{q} \begin{cases} -\left[\frac{4X'_{ia}}{\|X'_{i.}\|^4}\right] \left(\sum_{j=1}^{K'} X'_{ij} B_{jb} \right)^3 & \text{if } \|X'_{i.}\|^4 \neq 0 \\ 0 & \text{otherwise.} \end{cases}$$

A gradient-descent approach is described in Algorithm 1 in (Wen & Yin, 2013). This algorithm takes inputs: X' the matrix of orthogonal eigenvectors of M', X(0)' an orthogonal K'xK' matrix that serves as the starting point of the search and parameters $0<c$, $\rho<1$, $0<\epsilon$, $0<$maxNumSteps. The i^{th} step of the search will produce a new K'xK' orthogonal matrix X(i+1)'.

I. Set the step number i to 0 and compute gradient matrix G[X(0)'] and matrix A(0) = G[X(0)']X(0)'^{T} – X(0)'G[X(0)']^T.
II. For any $\tau>0$, let $Y(\tau)$ denote $[I+A(i)(\tau/2)]^{-1}[I-A(i)(\tau/2)]X(i)'$. [Lemma 3 in (Wen & Yin, 2013) shows that $Y(\tau)$ is defined, orthogonal, and represents a descent path for Θ from X(i)'.] Use a line search algorithm to determine τ' the step size taken along path $Y(\tau)$. The line search algorithm uses parameters ρ and c and is described later.
III. Set I to i+1 and X(i)' to Y(τ'). Compute gradient matrix G[X(i)'] and matrix A(i) = G[X(i)']X(i)'^{T} – X(i)'G[X(i)']^T.
IV. If i≥maxNumSteps or $||A(i)X(i)'||_F<\epsilon$, then return X(i)' and terminate, otherwise go to step II. The second condition checks whether the first-order Lagrange optimality condition is close enough to being satisfied at X(i)' -- as discussed in Lemma 1 of (Wen & Yin, 2013). If so, then X(i)' is regarded as an approximate critical point of Θ and the algorithm terminates.

For the stopping parameters ϵ and maxNumSteps, values of 0.00001 and 1000 were used, the same ones used by (Wen & Yin, 2013) in their experiments. The line search algorithm is called "backtracking-Armijo" and is described by Procedure 3.1 in (Nocedal & Wright, 1999). The algorithm takes inputs: X(i)' a K'xK' orthogonal matrix which is the starting point of the line search, *i.e.* Y(0) = X(i)', A(i) the matrix G[X(i)']X(i)'^{T} – X(i)'G[X(i)']^T, and parameters $0<\rho$, $0<c$.

I. Set τ' to 1.
II. Repeat until $\Theta([I+A(i)(\tau'/2)]^{-1}[I-A(i)(\tau'/2)]X(i)') \leq \Theta(X(i)') - 0.5\tau' c(||A(i)||_F)^2$

a. Set $\tau' = \rho\tau'$

The "Repeat-until" condition is drawn from (26a) and Lemma 3 part 3 in (Wen & Yin, 2013). It is based on the Armijo condition which, if true, implies that the step length yields a sufficient decrease in Θ. The geometric reduction in the step length ensures that the step length will not be too small. Common settings were used for ρ and c, 0.5 and 0.0001, respectively. The setting for c is based on (Nocedal & Wright, 1999, p. 38).

Acknowledgements

The author would like to thank a few people for their assistance in conducting this research. Richard MacMillan produced the Java implementation of the Bayesian segmentation algorithm. Navot Akiva provided the data used in all experiments.

Bibliography

Akiva, N., & Koppel, M. (2012). Identifying Distinct Components of a Multi-Author Document. *Proceedings of the European Intelligence and Security Informatics Conference*, (pp. 205-209).

Akiva, N., & Koppel, M. (2013). A Generic Unsupervised Method for Decomposing Multi-Author Documents. *Journal of the American Society for Information Science and Technology, 64*(11), 2256-2264.

Arthur, D., & Vassilvitskii, S. (2007). K-means++: the Advantages of Careful Seeding. *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA).*

Eisenstein, J., & Barzilay, R. (2008). Bayesian Unsupervised Topic Segmentation. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (pp. 334-343).

Graham, N., Hirst, G., & Marthi, B. (2005). Segmenting Documents by Stylistic Character. *Natural Language Engineering, 11*(4), 397-415.

Hearst, M. (1997). TextTiling: Segmenting Text in Multi-paragraph Subtopic Passages. *Computational Linguistics, 23*(1), 33-64.

Ji, X., & Zha, H. (2003). Domain-Independent Text Segmentation Using Anisotropic Diffusion and Dynamic Programming. *Proceedings 26th ACM SIGIR Conference on Research and Development in Information Retrieval*, (pp. 322-329).

Kazantseva, A., & Szpakowicz, S. (2011). Linear Text Segmentation Using Affinity Propagation. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (pp. 284-293).

Kehagias, A., Pavlina, F., & Petridis, V. (2003). Linear Text Segmentation Using a Dynamic Programming Algorithm. *Proceedings of the 10th Conference of the European Association for Computational Linguistics (EACL)*, (pp. 171-178).

Koppel, M., Schler, J., & Argamon, S. (2009). Computational Methods in Authorship Attribution. *Journal of the American Society for Information Science and Technology, 60*(1).

Koppel, M., Schler, J., & Bonchek-Dokow, E. (2007). Measuring Differentiability: Unmasking Pseudonymous Authors. *Journal of Machine Learning Research, 8*, 1261-1276.

Layton, R., Watters, P., & Dazeley, R. (2011). Automated Unsupervised Authorship Analysis Using Evidence Accumulation Clustering. *Natural Language Engineering, 19*(1), 95-120.

Manning, C., Raghavan, P., & Schutze, H. (2008). *Introduction to Information Retrieval.* Cambridge University Press.

McCallum, A. (2002). Retrieved July 16, 2013, from MALLET: MAchine Learning for LanguagE Toolkit : http://mallet.cs.umass.edu

Misra, H., Yvon, F., Cappe, O., & Jose, J. (2011). Text Segmentation: a Topic Modeling Perspective. *Information Processing and Management, 47*, 528-544.

Nocedal, J., & Wright, S. (1999). *Numerical Optimization.* New York, New York, USA: Springer Science+Business Media Inc.

Stamatatos, E. (2009). Intrinsic Plagiarism Detection Using Character n-Gram Profiles. *Proceedings of the PAN Workshop as Part of the 25th Annual Conference of the Spanish Society for Natural Language Processing (SEPLN).*

Stein, B., Lipka, N., & Prettenhofer, P. (2011). Intrinsic Plagiarism Analysis. *Language Resources and Evaluation, 45*(1), 63-82.

Strang, G. (2005). *Linear Algebra and Its Application* (4th ed.). Brooks/Cole Publishing Co.

Utiyama, M., & Isahara, H. (2001). A Statistical Model for Domain-Independent Text Segmentation. *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*, (pp. 491-498).

von Luxburg, U. (2007). A Tutorial on Spectral Clustering. *Statistics and Computing, 17*(4).

Wen, Z., & Yin, W. (2013). A Feasible Method for Optimization with Orthogonality Constraints. *Mathematical Programming, 142*(1-2), 397-434.

Zelnik-Manor, L., & Perona, P. (2004). Self-Tuning Spectral Clustering. *Proceedings of the 18th Conference on Neural Information Processing Systems (NIPS).*