leghist: Automated legislative history analysis for R*

Mark Huberty† and Hillary Sanders‡

December 28, 2012

Abstract

We describe an integrated workflow for automated analysis of legislative history in the R language. Legislative history analysis often requires laborious hand-matching of proposed amendments and their authors to their destination in a final bill. `leghist` deploys natural language and textual analysis methods to automate matching, estimate match accuracy, model the semantic content of amendments, tabulate the rate of acceptance and rejection of amendments by legislative actor and topic, and visualize the resulting flow of information. We provide a theoretical overview of the workflow and suggest this model could be extended to other analogous forms of legal and political document analysis. We finally provide a vignette based on the European Union's Third Renewable Energy Directive, adopted in 2008.

Contents

1 Introduction **2**

2 How a bill becomes a law: a model **2**

2.1 The simple model: a pure legislative process 3

2.2 Extensions: fuzzy matches 4

2.3 Extensions: negotiated content and revisions 5

2.4 Extensions: influence estimates 5

*Original version: 30 January 2012. Thanks to Adrienne Hosek for discussions on the comparative legislative processes for the United States Congress and the European Parliament.

†Travers Department of Political Science, University of California, Berkeley. Contact: markhuberty@berkeley.edu.

‡University of California, Berkeley.

3 The technical implementation **6**
3.1 Distance matching and estimation 7
3.2 Learning the distance threshold 8
3.3 Bill synthesis 9
3.4 Semantic clustering and impact analysis 9
3.5 Visualization 9

4 Vignette: the 2009 European Renewable Energy law **9**
4.1 Distance estimation and matching 10
4.2 Learning the similarity threshold 12
4.3 Generating and visualizing the synthetic bill 12
4.4 Impact analysis 13
4.5 Visualizing output 20

5 Accuracy trials **20**

6 Conclusions **20**

1 Introduction

Analysis of legislative history plays a central role in political science. The data often present barriers to making straightforward, or quantifiable progress. Few legislatures provide marked-up versions of final bills indicating when and how proposed legislation was amended, or by whom. Particularly for very large bills with many amendments, identifying these important sources of information on legislative influence and process constitute a very large search problem.

The `leghist` package for R is designed to help analysts address these barriers. `Leghist` provides computational tools for processing text from proposed legislation and amendments, matching sections of the final bill to their most likely origin in either the proposed bill or subsequent amendments, and quantifying the degree and form of variation among both successful and unsuccessful amendments. We show, through hand-coded trials, that the methods used here generate valid matches at a high degree of accuracy in terms of both human-identified textual similarity and evidence from the formal legislative record. Finally, we emphasize that the tools provided here should work for other document history processes analogous to the legislative history model we present below.

2 How a bill becomes a law: a model

This section presents an abstracted view of how a legislative proposal evolves into a final bill through three three, increasingly complex, processes. The biggest change imposed

by the added complexity concerns whether we should use supervised or unsupervised methods to identify the origin of the components of the final bill. While the most basic case permits a purely unsupervised approach, the more complex cases benefit from some additional user input. The tools provided in the `leghist` package assist the analyst in making progress on both the most basic case and each extension.

Analysts interested in legislative history may raise a set of questions about the history of piece of legislation:

- Which sections of the original bill survived to the final version?
- Which amendments were adopted, and which weren't?
- Who originated the adopted amendments, versus those not adopted?
- What was the content of the amendments, and does that content categorize in systematic ways (i.e., did amendments focus on particular topics within the policy domain)?
- Were there aspects of the final bill that have no good match, suggesting changes made behind the scenes (i.e., in the reconciliation process between the House and Senate in the United States Congress)

2.1 The simple model: a pure legislative process

Formally, we treat a bill as a document D composed of I sections d_i. The legislative process begins with an initial bill D_1, and concludes with a successful bill D_2. During the legislative process, J amendments $a_j \in A_J$ are proposed by some set of legislative actors L, of which $A_K, 0 \leq K \leq J$ are adopted. For D_1, D_2, and A_J, we assume that the sections $d_{1,i}, d_{2,i}$, and a_j are all at the same level of dis-aggregation (e.g. sentence, paragraph, article). The tools presented here are agnostic to that level of dis-aggregation, but for the rest of this paper we assume that sections are analogous to paragraphs. This is consistent with the structure of amendments for the European Parliamentary bills considered in section 4.

For the simple case, then, the final bill is a combination of the proposed bill and the proposed amendments. We can represent that process as a functional form of $D_2 = f(D_1, A_J)$. Each section $d_{2,i}$ contains one and only one best match in D_1, A_j. Given that, the challenge is to identify the sections of the original bill and the amendments that comprise the final bill.

To do so, we can thus define a similarity measure S to permit comparison between each section $d_{2,i}$ of the final bill and its potential origins. For simplicity, we assume that $S \in [0, 1]$. For each final section $d_{2,i}$ we construct a similarity vector s_i between all possible matches in D_1, A_J. The best match is thus $d \in arg\max_{s_i} S(D_1, A_J)$. Under the assumption

that S is well-ordered, we can use nearest-neighbor matching, since for each $d_{2,i}$ there is one and only one best match.[1]

To construct the distance metric, we represent each section of D_2, D_1, and A_j, as a term-frequency vector, where each element in the vector represents the frequency with which one of the set of unique words in the entire corpus occurs in this particular section. Given the set of all bill sections in a term-frequency matrix, we can measure the similarity between any two bill sections using any continuously-valued similarity or distance metric.

2.2 Extensions: fuzzy matches

The first extension to this stylized legislative process involves a set of fuzzy potential matches. Two possible developments will lead to fuzzy matches. First, specific language in a bill, such as cross-references to other bill sections, dates, or formal legal language, may change without affecting the substantive semantic content of the bill section and its best match. Second, the drafting process may split apart and recombine sections prior to the final bill, without necessarily affecting the origin of the legislative language.[2] We would wish to identify both cases.

Nearest-neighbor matching with replacement handles both problems. Matching with replacement permits a single amendment to match to multiple sections of the final bill. This same approach also works for subtle changes in language introduced during final drafting of the bill. For instance, consider the section of a final bill (left) and its match in the initial bill (right) presented as below:

to develop renewable energies to meet the commitment of the Community to using 20% renewable energies by 2020, as well as to develop other technologies contributing to the transition to a safe and sustainable low-carbon economy and to help meet the commitment of the Community to increase energy efficiency by 20% by 2020;	to develop renewable energies to meet the commitment of the Community to using 20% renewable energies by 2020, and to meet the commitment of the Community to increase energy efficiency by 20% by 2020;

Here, the use of nearest-neighbor matching does not require the exact match between the final bill section and its candidate sources. Rather, so long as the similarity measure produces a well-ordered similarity vector, the match will return the correct value.

[1] A distance metric can also be used, in which case the choice problem becomes $d \in arg\min_{s_i} S(D_1, A_J)$.

[2] We also note that this set of changes may be introduced by the process of producing machine-readable text. Given that much of this text must be scraped by the analyst, this form of variation may be introduced by the scraping algorithm itself. Validating the algorithm for each bill represents a significant effort that we would like the analyst to avoid.

It is important to note one situation in which this matching process breaks down. Introduction of the same amendment by multiple legislative actors, or at multiple stages of the legislative process, will produce more than one match, since the similarity vector is no longer well-ordered. Absent additional information, it is impossible to determine which is the "correct" match from this set. Thus the algorithm may return the correct *semantic* origin without necessarily returning the correct *procedural* origin. Presently, `leghist` breaks such ties by choosing the last version of the amendment submitted, on the supposition that an amendment would not be re-submitted if it had been accepted at an earlier stage of the legislative process. We will return to this issue when we discuss supervised processes.

2.3 Extensions: negotiated content and revisions

Finally, some legislative processes may not require that all changes to a bill be introduced as formal amendments. Examples here include the reconciliation process in the United States Congress, and the co-decision negotiation process between the European Parliament and Council. In this case, substantive changes to a bill may occur without any matching record in either the original bill or the proposed amendments.

Given that the nearest-neighbor process defined above will return some match even if no actual match exists, handling this case requires a filter for "poor" matches. We implement this process with a filter on the similarity measure itself: for matches whose similarity measure s is below a user-supplied threshold T, we reject the match and instead match the section to itself. This "false" match can aid the identification of elements in a bill that came out of negotiated rather than formal amendment processes.

Setting the threshold poses the precision/recall trade-off. A low threshold level will match 100% of the actual matches (high recall), while returning many false positives (low precision). A high threshold level will do the reverse. `Leghist` treats this trade-off as a supervised learning problem in which the optimal T is derived from a set of user-coded data. We provide tools to automate the hand-matching of a random subset of the final bill to its potential sources. Based on that hand coding, the proper threshold value can be chosen to maximize the accuracy of the matching algorithm. We provide the capability to estimate the optimal T based on both overall accuracy and the precision/recall trade-off.

2.4 Extensions: influence estimates

Tsebelis et al. (2001) suggest two measures of actor influence in the legislative process. We can describe these as:

1. *Semantic impact* measures whether the actors are successful at introducing substantively new content to bills (as opposed to simply tweaking existing content). An example here would be expanding an agriculture subsidy bill to cover humane treatment of animals.

2. *Semantic completeness* measures whether proposed amendments were adopted in whole or only in part, regardless of whether they introduced new substance or modified existing content.

Semantic impact can be treated as a cluster analysis problem. We treat the entire corpus of bills and amendments (a corpus of length $D_1 + D_2 + A_J$) as representing K latent topics. Each section d_i, a_j can be assigned to one of these K topics. Substantively new content is thus a set of topics $K^* \in D_2, K^* \notin D_1$. Legislative actors whose contributions constitute a higher proportion of K^* are identified as having greater substantive impact on outcomes.

Semantic completeness can be derived directly from the distribution of the similarity or distance measures for the accepted amendments from each legislative actor. Higher similarity metric values indicate amendments adopted almost verbatim; lower values suggest partially-adopted amendments.

Finally, we may also wish to identify *contested* policy domains. To the extent that contestation appears in the written legislative record, we could abstract it as follows: A section of the final bill $d_{2,i}$, is considered more contested if:

1. The distribution of its similarity metric S is bimodal, with a small cluster of very close matches and a large body of poor matches
2. The small cluster of close matches comes from a more diverse set of legislative actors

Condition (1) suggests how we can programatically identify contestation from the distribution of the similarity metric. Condition (2) then eliminates those cases where one legislative actor continually submitted the same amendment despite having little actual success. Together, these suggest a mechanism for inferring contestation from data.

3 The technical implementation

In `leghist`, we implement this model in five stages:[3]

1. Nearest-neighbor matching of the final bill to candidate origins
2. Semi-supervised learning of the right threshold values
3. Synthetic bill construction
4. Semantic clustering and impact analysis
5. Visualization and presentation

[3]Code implementing `leghist` is available at https://github.com/markhuberty/leghist.

3.1 Distance matching and estimation

Given the model presented in section 2, the analytic problem thus becomes identifying which amendments and portions of the initial bill become the final bill. We treat this as a document retrieval problem, wherein we wish to query for a document $D_{2,i}$ and retrieve its most likely match from D_2, A_J. [4]

We implement a generic comparison strategy in four steps:

1. Transform each section of D_2, D_1, and A_J into a bag-of-words vector-space representation in which features are n-grams of potentially heterogeneous length

2. Construct the pairwise similarity, using some similarity metric, between each section of D_2 and all sections in D_1 and A_J.

3. Use nearest-neighbor matching with replacement to establish the closest match between each section of D_2 and a section in D_1 and a candidate amendment in A_J

4. Pick the closest match of the pair of potential matches d_i and a_j

Implementation of step (1) occurs via the `tm` package for R (Meyer et al., 2008), and uses the Weka tokenizer for term tokenization (Hall et al., 2009). This infrastructure provides users the ability to specify whether words should be stemmed; whether punctuation, excess whitespace or stopwords should be removed; the minimum and maximum n-gram length used for constructing the vector space representation of text, and whether the term list should be filtered on the basis of one of several metrics including term-frequency and term frequency / inverse document frequency measures.

The `leghist` package comes with several alternative distance metrics for use in step (2). For token-based distance metrics, we implement a standard cosine similarity, a length-weighted cosine similarity, and a length-weighted joint information distance measure as described by Hoad and Zobel (2003). Each is implemented as matrix operation, permitting rapid pairwise distance calculations even for relatively large (∼1000+ amendment) corpora. In plagiarism applications, length-sensitive distance metrics have performed well in identifying correct documents from otherwise quite similar candidate matches. However, the matching process here appears to favor the fuzzy matches provided by the length-insensitive cosine similarity. This may, as we will discuss, be due to the somewhat arbitrary choice of what constitutes a document section, and its instability over time. Finally, the function interfaces allow the user to supply any distance function so long as it returns a correctly-formatted distance matrix. We also implement an interface to string-based metrics like the Levenshtein edit distance (Elmagarmid et al., 2007), though these

[4] This can, alternatively, be treated as a subset of the plagiarism problem, wherein we know with probability approaching one that D_2 has plagiarized content from D_1 and A_J. As with other versions of plagiarism detection, the problem then becomes establishing an effective and efficient comparison strategy.

metrics are far more computationally-intensive than the cosine similarity or set-based measures like the Jaccard index.

Step (3) uses simple nearest neighbor matching with replacement to identify the most likely candidate matches from both the initial document and the set of proposed amendments. Both matches are returned in an intermediate step to permit users to identify the connection to the initial bill and to the potential amendments. Finally, under the assumption that the similarity metric is consistent and well-ordered, we can then in step (4) compare this pair of potential matches to determine the best potential match from all candidate matches.

3.2 Learning the distance threshold

Given the distance matrix, construction of the composite matched bill requires a threshold value for accepting or rejecting matches. Selection of an optimum threshold value will depend on the specifics of the legislative process behind each bill. For instance, the right threshold value for a bill that was modified only via the amendment process may be much lower than a threshold value valid for a bill that went through a negotiation or reconciliation process. While our experiments suggest that values on the order of 0.3 perform reasonably well, we recommend that users base threshold selection on the performance of the matching algorithm in their particular case. For instance, legislative procedures known to have significant negotiated content may benefit from a higher threshold value, while those with very little negotiated content might require a threshold value close to 0.

Identification of the context-specific threshold value constitutes a supervised learning problems. To facilitate identification of context-specific threshold values, we provide tools that allow the user to hand-code a subset of a given bill and derive the optimum threshold value based on the algorithm's performance against that hand-coded data. In practice, coding of 30% of a bill appears sufficient to derive acceptable threshold values.

Figure 1 provides an example of the performance of the document retrieval process as established by this supervised approach. For each of four different bills, 30% of the final bill was hand-matched to the most semantically appropriate sources in either the original draft of the bill or subsequently-proposed amendments. Based on this sub-sample, we established the optimum threshold based on both overall accuracy and the precision-recall trade off between Type 1 (precision) and Type 2 (recall) errors. In all cases, the optimum threshold lay between 0.3-0.45. The variation in the optimum threshold setting appears to derive from two primary sources: the degree of negotiation (as opposed to formal amendment) in the legislative process itself; and the amount of structural change that occurred when amendments were incorporated into formal legislative language.

3.3 Bill synthesis

Based on the pairwise matches between sections in the final bill and the set of potential sources for the final version of those sections, we can construct a synthetic bill for comparison purposes. The package provides tools to construct a side-by-side comparison of the final bill sections to their candidate matches. If the user supplies a non-zero threshold value as described in section 2.3, that value is used to determine whether the actual match should be used. If the threshold value is not met, then the final document section is matched to itself.

3.4 Semantic clustering and impact analysis

Finally, the legislative process rarely amends bills at random. Rather, specific sections or issue areas within bills will often prove more contentious than others. The user may wish to identify these areas on the basis of accepted and rejected amendments, sections of the original bill that persist to the final bill, and sections of the final bill for which no match was identified.

We provide a ready interface between the matched bill and the `topicmodels` package for R that enables topic modeling of these different slices of the matched bill. As described by Blei et al. (2003), topic modeling assumes a latent structure within a set of documents. Each document–in this case, a bill section–is assumed to encompass one or more of an unobserved set of latent topics that underpin the entire document collection. Each topic, in turn, is represented by distribution over a set of terms. Latent Dirichlet Allocation provides a Bayesian method for inferring topics from empirical word distributions, and assigning documents to those topics on the basis of document-specific distributions.

3.5 Visualization

Finally, `leghist` provides, via the `WriteSideBySide` functionality, for side-by-side comparison of the actual final bill and its synthetic match in a two-column LaTeX document, which can be directly compiled into a PDF. Within that document, each paragraph of the final bill is shown side-by-side with its match. Words differences between the paragraphs are highlighted as colored text. Margin notes indicate the origin of the matched segment and its similarity or distance measure compared with the section to which it is matched.

4 Vignette: the 2009 European Renewable Energy law

To illustrate each of these tools and their technical underpinnings, we now provide a vignette based on actual legislation as passed by the European Parliament. In 2008, the European Commission proposed a new round of reforms to renewable energy policy. This

reform followed on the 2001 Renewable Energy Directive. This series of legislation set Europe-wide and country targets for renewable energy adoption and established a European framework for renewable energy investment and subsidy. The reforms also played an important role in the multi-part bargain on energy and climate change introduced in the Third Climate and Energy Package.

The original bill in this case was a package of amendments to the 2001 legislation proposed by the European Commission as Commission document COM 2008 0019. Based on this document, the European Parliament submitted a range of amendments in the report tabled for the plenary session in Strasbourg. Subsequently, the Parliament and the Council agreed on a final bill, which passed both bodies. The Commission did not contest any of the agreed-upon changes. The final bill was adopted as Directive 2009 (28).[5]

This section demonstrates the operation of the `leghist` package in analyzing the amendment process for this legislation. We treat the initial bill as the 2001 legislation which the Commission proposed to change. Amendments then constitute the Commission proposal and the amendments contained in the First Reading Report.

We begin by reading in the bills and amendments as plain text. In each case, the bills had been parsed from original documents available from the European Legislative Observatory.

```
ep.first.2007 <- read.csv("./2007/txt/ep_first_reading_report.csv",
                          header=TRUE,
                          stringsAsFactors=FALSE
                          )

bill.initial.2007 <- readLines("./2001/orig/directive_2001_77_ec.txt")
commission.proposal.2007 <- readLines("./2007/txt/com_2008_0019.txt")
bill.final.2007 <- readLines("./2007/txt/directive_2009_28_ec.txt")

committees.2007 <- c(rep("Parliament 1st reading",
                         nrow(ep.first.2007)),
                     rep("Commission",
                         length(commission.proposal.2007))
                     )
```

4.1 Distance estimation and matching

The documents are then transformed into a single document-term matrix containing a bag-of-words representation of each section of the bills and proposed amendments. Each row of the matrix represents one section of one document. Columns represent unique

[5] A full procedural record can be found at http://www.europarl.europa.eu/oeil/popups/ficheprocedure.do?reference=2008/0016(COD)&l=en. Referenced 10 June 2012.

terms in those documents as term frequencies for those terms in each document section. In this case, we do not stem the terms, but do remove English stopwords, excess whitespace, and punctuation. We also use a range of n-grams, to preserve the semantic resolution of consecutive-word combinations.

```
doc.list.2007 <- CreateAllVectorSpaces(bill.initial.2007,
                                       bill.final.2007,
                                       c(ep.first.2007$text,
                                         commission.proposal.2007
                                         ),
                                       ngram.min=1,
                                       ngram.max=3,
                                       stem=FALSE,
                                       rm.stopwords=TRUE,
                                       rm.whitespace=TRUE,
                                       rm.punctuation=TRUE,
                                       filter=NULL,
                                       filter.thres=NULL,
                                       weighting=weightTf
                                       )
```

The `doc.list.2007` is a R `list` containing four components::

- `vs.out`, the document-term matrix represented as a sparse `dgCMatrix` object (Bates and Maechler, 2012)
- `idx.final`, an integer vector of row indices representing documents in the final bill
- `idx.initial`, an integer vector of row indices representing documents from the initial bill
- `idx.amendments`, an integer vector of row indices representing documents from the proposed amendments

Using the document-term matrix, the `MapBills` function generates a distance or similarity matrix between all sections of the final bill and their candidate origins in the initial bill and proposed amendments. Here, we use `CosineMat`, a version of the cosine similarity optimized for this use case by using matrix algebra to compute all distances simultaneously. However, `MapBills` will accept any function that returns a distance matrix of the $N_{final} \times N_{compare}$.

```
map.bills.2007.cos <- MapBills(doc.list.2007,
                               distance.fun="CosineMat"
                               )
```

`MapBills` returns an R data frame with five columns: the section index of the final bill, and the section index and distance measure of both the original bill match and the amendment match. We can see a sample of that data frame here:

```
  bill2.idx bill1.idx bill1.dist amend.idx amend.dist
1         1         1        NaN         1        NaN
2         2         6  0.2377064       981  0.7376436
3         3         3  0.7177070      1082  0.3880453
4         4         4  0.5805601      1083  0.6782620
5         5        24  0.1116871        94  0.8236878
```

4.2 Learning the similarity threshold

To select the optimum threshold value as discussed in section 3.2, `leghist` provides a 2-step workflow. First, the `RunEncoder` routine allows the user to encode a randomly-sampled portion of the final bill. It presents the user with a set of candidate matches, based on likely matches from both the initial bill and amendments. The user provides the index of the best match, or `NA` if no valid match is available.[6] Figure 4 provides a sample of the `RunEncoder` interface.

Second, based on the output of `RunEncoder`, `LearnThreshold` computes accuracy measures for a synthetic bill based on a vector of threshold values. Based on those accuracy measures, `LearnThreshold` returns optimum threshold values from the sequence. We recommend that the threshold sequence be specified at a fairly granular level, such as `seq(0, 0.5, 0.005)`. The optimum threshold value may be identified in two ways based on the `type` argument to `LearnThreshold`. With `type=overall`, the aggregate accuracy of the matched pairs is estimated, and the threshold value corresponding to the highest accuracy returned. With `type=tradeoff`, the threshold value at the intersection of the the precision/recall tradeoff curves.

Figure 2 shows the output of `LearnThreshold` for both `overall` and `tradeoff` settings.We can see that the optimum choice of threshold value is about 0.45.

4.3 Generating and visualizing the synthetic bill

`GetLikelyComposite` provides functionality for building up the match to the final bill from the output of `MapBills` and the learned distance threshold. It takes a series of arguments: the output from MapBills, the plain text used to create the original `doc.list` object, and labels that indicate who originated the amendments (in this case, the Parliament at First Reading, and the European Commission). For any section of the final

[6] We note that users who do not want to use the automated matching tools may still use the encoding tools to hand-match amendments. The tools reduce the set of possible matches to a set of likely matches based on the same vector space language representation used by the automated match algorithms, thereby reducing the search space for hand-coded matches.

bill, `GetLikelyComposite` will return the best match if the similarity threshold exceeds `dist.threshold`; otherwise, it will insert the same section of the "final" bill as the best match. This can be used to identify sections of the bill with uncertain origins, which perhaps arose outside of the formal amendment process.

```
rese.composite.2007.cos <- GetLikelyComposite(map.bills.2007.cos,
                                              bill.initial.2007,
                                              bill.final.2007,
                                              c(ep.first.2007$text,
                                                commission.proposal.2007
                                                ),
                                              committees.2007,
                                              filter="max",
                                              dist.threshold=0.45
                                              )
```

4.4 Impact analysis

To identify the semantic and substantive influence of the EU legislative actors, we make use of the topic modeling capabilities in the `topicmodels` package (Grün and Hornik, 2011). These, in turn, provide R interfaces to the Latent Dirichlet Allocation methods implemented in Blei et al. (2003) and Blei and Lafferty (2006). These methods infer a set of topics from the term distributions over a set of documents, on the assumption that topics are differentiated by different distributions over a set of terms. The `leghist` package provides a structured interface to cluster legislative text using these tools and tabulate the output.

We first identify new substantive areas introduced into the final bill during the legislative process. The `ModelTopics` function provides a means for modeling the topic distribution for the entire corpus. We recommend that users select the optimum k value on the basis of both formal model selection criteria and substantive topic coherence. Wallach et al. (2009) provide a range of formal methods; `topicmodels` also implements log-likelihood and perplexity measures.

```
## Subset the columns to include only 1-grams
addl.stopwords <- c("december", "paragraph", "article", "january",
                    "march", "june", "commission", "parliament", "community"
                    )
idx.to.keep <- leghist:::GetNgramIdx(colnames(doc.list.2007$vs.out),
                                     1
                                     )
rese.1gram.2007 <- doc.list.2007$vs.out[,idx.to.keep]
```

```
## Here k was chosen to maximize the log-likelihood of
## the model on a held-out 10% sample of the full
## document-term matrix
rese.full.model <- ModelTopics(rese.1gram.2007,
                          idx=c(doc.list.2007$idx.final,
                            doc.list.2007$idx.initial,
                            doc.list.2007$idx.amendments),
                          k=26,
                          sampling.method="VEM",
                          topic.method="LDA",
                          control=list(
                            var=list(tol=10^-4),
                            em=list(tol=10^-3)
                            ),
                          n.terms=10,
                          addl.stopwords=addl.stopwords,
                          na.rm=FALSE
                          )
```

With the full model, we can now match topics to individual document sections, their status as accepted or rejected in the final bill, and the legislative actor that proposed them.

```
rese.topics.2007 <- CollateTopicDtm(rese.composite.2007.cos,
                                    rese.full.model,
                                    doc.list.2007,
                                    committees.2007
                                    )

impact <- EstimateSourceImpact(rese.topics.2007)
```

`CollateTopicDtm` returns two matrices with topic and status information. The first, `doc.topic` maps the entire document-term matrix to the committee that originated each row, the status of that document section, and the topic assigned by the model. The second, `composite.topic`, is a matrix of the same form as `GetLikelyComposite`, augmented with the topic assigned to both the final bill section and its assigned match in the original bill or amendments. Sample output is shown below.

```
> rese.topics.2007$doc.topic[1:5,]
           source committee idx status topic
result.1 doc.final     final   1    acc  <NA>
result.2 doc.final     final   2 redund    22
result.3 doc.final     final   3 redund     6
result.4 doc.final     final   4 redund     3
```

```
result.5 doc.final     final   5 redund    16
```

`EstimateSourceImpact` then operates on these results to return tables identifying semantic additions to the original proposed legislation. It identifies which topics in the final bill are not present in the initial, and tabulates documents assigned to those topics by legislative actor. It also breaks down the quality of the match, estimated by the similarity metric, as a measure of how well For the 2007 renewable energy bill, we can easily see that the Parliament had substantially more influence over topics 1, 7, and 11. Looking at the terms returned from `ModelTopics`, it's clear that the Parliament's influence in new policy domains came primarily in areas surrounding the impact of biofuels on third countries and on land use; and secondarily on influence cam largely in amendments governing national renewable energy action plans, the impact of biofuels on third countries and agricultural land, and regulatory compliance procedures.

```
> round(impact$prop.origin.topic, 2)
## Proportion of bill in each topic by contributing
## legislative actor
                        Topic
Source                     1   11   12   14  16   18   19    2   23   26    5
  Commission             0.0 0.33 0.31 0.58 0.5 0.46 0.75 0.57 0.20 0.75 0.43
  Final                  0.5 0.27 0.56 0.27 0.2 0.15 0.00 0.14 0.67 0.08 0.36
  Parliament 1st reading 0.5 0.40 0.12 0.15 0.3 0.38 0.25 0.29 0.13 0.17 0.21

                        Topic
Source                      7 9
  Commission             0.20 1
  Final                  0.47 0
  Parliament 1st reading 0.33 0

      Topic 1      Topic 11      Topic 18         Topic 2         Topic 7
 [1,] "countries"  "land"        "training"       "materials"     "accordance"
 [2,] "third"      "raw"         "action"         "raw"           "regulatory"
 [3,] "country"    "purposes"    "national"       "emissions"     "adopted"
 [4,] "convention" "material"    "local"          "cultivation"   "measures"
 [5,] "impact"     "biofuels"    "plans"          "values"        "referred"
 [6,] "concerning" "obtained"    "regional"       "agricultural"  "procedure"
 [7,] "community"  "provided"    "administrative" "production"    "elements"
 [8,] "labour"     "forest"      "including"      "gas"           "essential"
 [9,] "biofuel"    "significant" "planning"       "greenhouse"    "infrastructur
[10,] "treaties"   "bioliquids"  "bodies"         "list"          "scrutiny"
```

We also provide a hierarchical interface, which first clusters document sections into

semantically-related groups, and then clusters the documents inside each cluster into semantic sub-groups. This functionality implements the following intuition: consider a group of amendments and bill sections classified as "procedural" and identified because of the predominance of terms like "audit", "budget", "monitor", and "oversight". We would expect that, within that cluster, we would observe that different amendments focus on different aspects of these procedural elements of legislation. Hence sub-clustering provides a means of breaking down bills and amendments into a sensible 2-level semantic hierarchy. Analysts could, of course, iterate this process for any desired level of hierarchy.

We also provide a means of tabulating the committee contributions by both primary and secondary topic areas. The `ctab.amend.hierarchy` function takes the output from `model.amend.hierarchy` and returns the nested cross-tabulation of committee contributions to each semantic cluster and sub-cluster.

```
## Define the number of primary topics
k.main <- 8
rese.topics.2007 <-
    ModelAmendHierarchy(doc.list.2007,
                        rese.composite.2007,
                        k=c(k.main, rep(5, k.main)),
                        addl.stopwords=c("article",
                          "paragraph", "follow", "replace",
                          "insert",
                          "ensure", "regulation",
                          as.character(0:9),
                          "commission",
                          "directive", "annex",
                          "parliament",
                          "council",
                          "alia", "whereas",
                          "january", "december", "july",
                          "deleted", "added", "amend",
                          "draft", "http", "mso",
                          "allowincell", "textbox",
                          "endiftextbox", "div"),
                        n.terms=10,
                        ngram=1,
                        sampling.method="VEM",
                        topic.method="CTM",
                        control=list(
                          var=list(tol=10^-4),
                          em=list(tol=10^-3),
```

```
                          seed=2342),
                        sparseness.probs=c(0.01, 0.999)
                        )


rese.topic.proportions.2007 <-
  CtabAmendHierarchy(rese.topics.2007,
                     rese.composite.2007,
                     committees.2007,
                     doc.list.2007,
                     tab.idx=2
                     )
```

Example output is shown below. We can quickly observe that the Commission had, for instance, far more influence on amendments related to topic 5, on specification and definition of renewable liquid fuels, than the Parliament. Likewise, the Parliament exercised greater influence over amendments in topic 6, governing the transfer of renewable energy production credits among EU countries and between EU countries and third parties. Likewise, we can see that within the general set of regulations governing grid operators (topic 7), different groups of amendments governed connection requirements, administrative licensing and planning, emissions reduction and renewable energy accommodation, and energy efficiency. These inferences proceed in straightforward fashion from the `leghist` workflow combining an automated approach to legislative history tracking, and readily available tools for semantic clustering of text.

```
## Top ten words by influence for the 8-topic model
      [,1]             [,2]            [,3]            [,4]
 [1,] "bioliquids"     "training"      "consumption"   "raw"
 [2,] "countries"      "solar"         "share"         "support"
 [3,] "significant"    "accordance"    "heat"          "materials"
 [4,] "third"          "benefits"      "final"         "sustainability"
 [5,] "provided"       "adopted"       "target"        "schemes"
 [6,] "sustainability" "share"         "overall"       "development"
 [7,] "environmental"  "procedure"     "means"         "bioliquids"
 [8,] "food"           "regulatory"    "mandatory"     "material"
 [9,] "protection"     "certification" "set"           "environmental"
[10,] "sustainable"    "appropriate"   "installations" "report"
      [,5]        [,6]           [,7]           [,8]
 [1,] "oil"       "origin"       "operators"    "fuels"
 [2,] "values"    "transfer"     "table"        "emission"
 [3,] "ethanol"   "guarantees"   "grid"         "resources"
 [4,] "waste"     "competent"    "carbon"       "specific"
```

```
 [5,] "process"   "accounting"   "require"      "savings"
 [6,] "wood"      "body"         "distribution" "based"
 [7,] "default"   "guarantee"    "procedures"   "efficiency"
 [8,] "vegetable" "certificates" "transmission" "cogeneration"
 [9,] "biogas"    "issued"       "producers"    "information"
[10,] "biofuel"   "compliance"   "rules"        "changes"
```

```
## Distribution of amendment acceptance and rejection
## by legislative actor and topic number
                            acc   rej

1 Commission                50.0   5.5
  Parliament 1st reading    50.0  94.5
2 Commission                61.8   8.0
  Parliament 1st reading    38.2  92.0
3 Commission                56.5   6.2
  Parliament 1st reading    43.5  93.8
4 Commission                54.5   4.5
  Parliament 1st reading    45.5  95.5
5 Commission                77.8  18.4
  Parliament 1st reading    22.2  81.6
6 Commission                36.1  21.2
  Parliament 1st reading    63.9  78.8
7 Commission                54.5   6.3
  Parliament 1st reading    45.5  93.7
8 Commission               100.0   6.7
  Parliament 1st reading     0.0  93.3

## Primary and secondary topic features for topic 7
[[7]]
[[7]]$terms.primary
 [1] "operators"    "table"        "grid"         "carbon"       "require"
 [6] "distribution" "procedures"   "transmission" "producers"    "rules"

[[7]]$terms.secondary%$
      Topic 1        Topic 2     Topic 3      Topic 4       Topic 5
 [1,] "operators"    "table"     "procedures" "carbon"      "administrative"
 [2,] "grid"         "filled"    "equipment"  "wind"        "certification"
 [3,] "costs"        "stroked"   "systems"    "economic"    "authorisation"
 [4,] "transmission" "rectrect"  "conversion" "operators"   "licensing"
 [5,] "distribution" "carbon"    "promote"    "information" "local"
 [6,] "producers"    "unit"      "achieve"    "roads"       "planning"
 [7,] "require"      "measured"  "efficiency" "offshore"    "applications"
 [8,] "connection"   "stock"     "stock"      "development" "regional"
 [9,] "regions"      "mass"      "carbon"     "impact"      "body"
[10,] "rules"        "molecular" "annualised" "priority"    "established"
```

4.5 Visualizing output

`leghist` provides a visualization framework for assessing the quality of the composite bill and the flow of text from legislative actors to final status. Based on the `GetLikelyComposite` output, `WriteSideBySide` provides an interface for building a 2-column LaTeX document showing the final bill and its matched origins. Margin notes indicate the source of the matched paragraph, its index number, and the distance or similarity measure for its match to the target paragraph in the final bill. Sample PDF output is shown in figure 4.5.

```
rese.sbs.2007 <- WriteSideBySide(rese.composite.2007.cos,
                                 bill.final.2007,
                                 cavs.out=doc.list.2007,
                                 file.out="ep_2007_rese_cos.tex",
                                 dir.out="./2007/tex/",
                                 pdflatex=TRUE
                                 )
```

5 Accuracy trials

The analyses that `leghist` makes available are only as good as the quality of the matches. We assess the accuracy of the matching algorithm against a human-matched version of the same bill. Using the `RunEncoder` interface provided in `leghist` and shown in figure 4, we had-coded matches for all 380 sections in the final bill. We then compared the algorithmic match to the manual match using the `LearnThreshold` tools. Figure 5 shows the results of those comparisons. For a threshold value of 0.45, the algorithm correctly identified and matched amendments in 84% of cases. The rate of correct source + index matching was lower. The discrepancy between identifying something as an amendment, versus *which* amendment, is largely due to the presence of ties. If more than one committee submits an amendment with identical language, the algorithm breaks ties based on order in the amendment data. Manual matching may return the same substantive match, but a different index.

6 Conclusions

The `leghist` package provides a range of utilities for identifying the origin of sections of a final piece of legislation given a set of possible sources. It further provides tools to analyze, cluster, and visualize these matches. We hope that these tools will prove useful to analysts of legislation and other documents in which the process of document preparation is as important as the result.

We anticipate several useful extensions. A more elaborate document retrieval process would perhaps improve on the quality of the matches. Likewise, we might imagine a

further set of algorithms that help determine the optimal level at which to compare the final and source documents, on the basis of some loss function. Furthermore, we may be able to exploit the distribution of the distance metric for each match to assess match quality, inferring the right threshold level T from data rather than from manual user input; that would improve `leghist`'s utility when hand-coding was unfeasible, as with very large document corpora. Finally, the process presented here makes no use of positional information; the fact that amendments are proposed to specific sections of a document gets lost in the parsing stage. While the process of drafting legislation destroys the value of a specific index, it does not necessarily mean that a paragraph once at the center of a bill now appears at the very end. Exploiting that information to restrict the set of potential matches may improve overall match quality. Finally, we are interested in, and hope to implement in future versions, measures of semantic, rather than textual, similarity to quantify differences in the degree of change implied by competing amendments to the same section of a bill.

References

Bates, D. and Maechler, M. (2012). *Matrix: Sparse and Dense Matrix Classes and Methods*, r package version 1.0-3 edition.

Blei, D. and Lafferty, J. (2006). Correlated topic models. *Advances in neural information processing systems*, 18:147.

Blei, D., Ng, A., and Jordan, M. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022.

Elmagarmid, A. K., Ipeirotis, P. G., and Verykios, V. S. (2007). Duplicate record detection: a survey. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 19(1).

Grün, B. and Hornik, K. (2011). topicmodels: An r package for fitting topic models. *Journal of Statistical Software*, 40(13):1–30.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. (2009). The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Hoad, T. and Zobel, J. (2003). Methods for identifying versioned and plagiarized documents. *Journal of the American society for information science and technology*, 54(3):203–215.

Meyer, D., Hornik, K., and Feinerer, I. (2008). Text mining infrastructure in r. *Journal of Statistical Software*, 25(i05).

Tsebelis, G., Jensen, C., Kalandrakis, A., and Kreppel, A. (2001). *Legislative procedures in the European Union: An empirical analysis*. Cambridge Univ Press.

Wallach, H., Murray, I., Salakhutdinov, R., and Mimno, D. (2009). Evaluation methods for topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1105–1112. ACM.

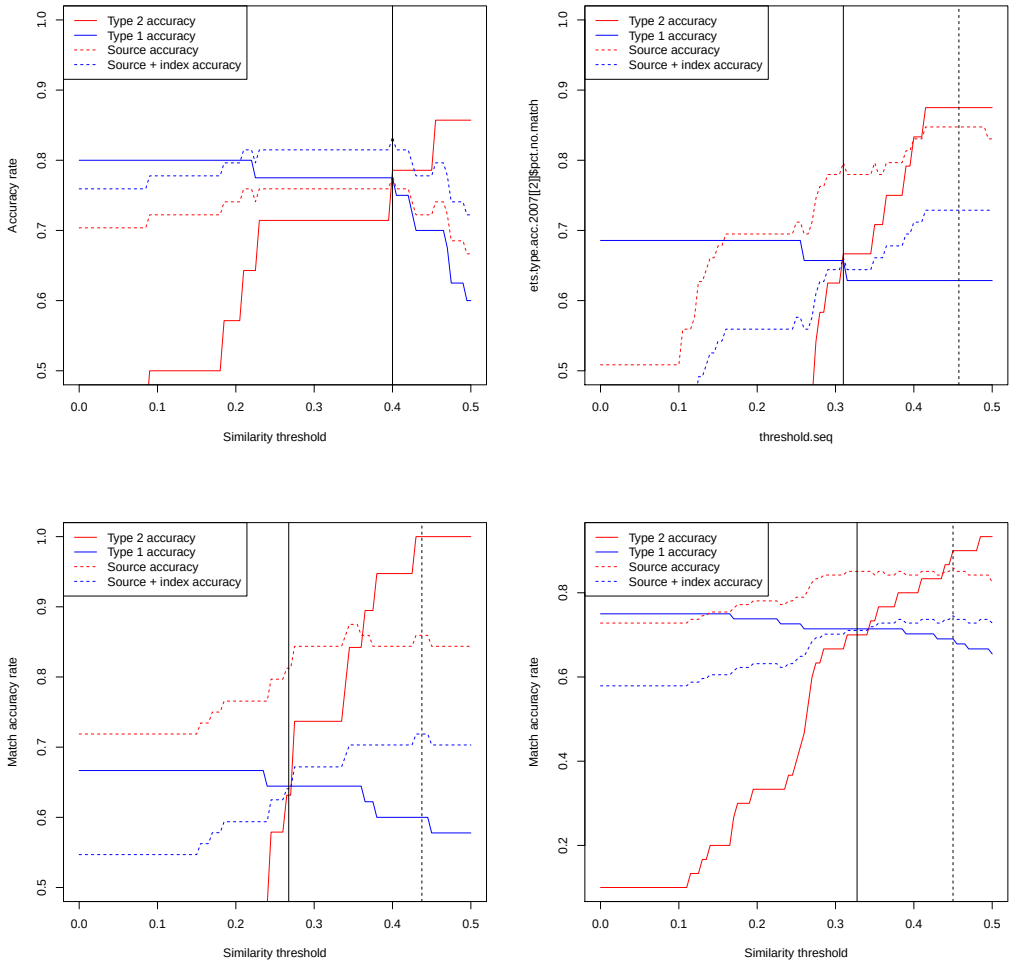


Figure 1: Accuracy assessment curves for 30% training samples in each of four different bills. Bills were drawn from the European Union in years 2001, 2003, and 2007. Optimum threshold values ranged from 0.3-0.4 in all cases.

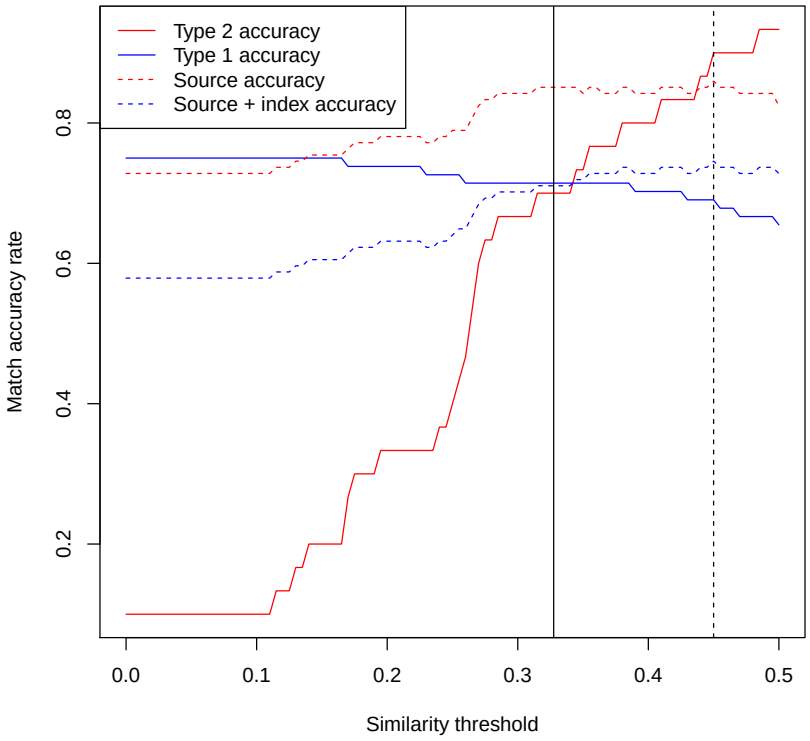
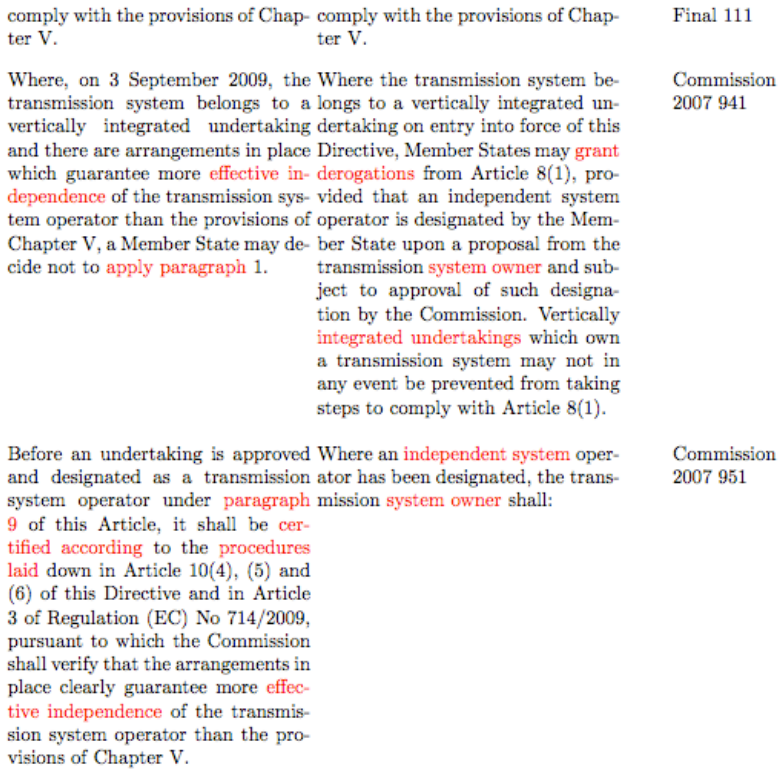


Figure 2: Accuracy curves as returned from the `LearnThreshold` function. Notice that the optimum threshold value is closer to the accuracy curve than the type 1/2 tradeoff curve intersection, owing to the relative flatness of the tradeoff curves.



comply with the provisions of Chapter V.	comply with the provisions of Chapter V.	Final 111
Where, on 3 September 2009, the transmission system belongs to a vertically integrated undertaking and there are arrangements in place which guarantee more effective independence of the transmission system operator than the provisions of Chapter V, a Member State may decide not to apply paragraph 1.	Where the transmission system belongs to a vertically integrated undertaking on entry into force of this Directive, Member States may grant derogations from Article 8(1), provided that an independent system operator is designated by the Member State upon a proposal from the transmission system owner and subject to approval of such designation by the Commission. Vertically integrated undertakings which own a transmission system may not in any event be prevented from taking steps to comply with Article 8(1).	Commission 2007 941
Before an undertaking is approved and designated as a transmission system operator under paragraph 9 of this Article, it shall be certified according to the procedures laid down in Article 10(4), (5) and (6) of this Directive and in Article 3 of Regulation (EC) No 714/2009, pursuant to which the Commission shall verify that the arrangements in place clearly guarantee more effective independence of the transmission system operator than the provisions of Chapter V.	Where an independent system operator has been designated, the transmission system owner shall:	Commission 2007 951

Figure 3: Sample PDF output from the `WriteSideBySide` function, for the 2009 European energy market reform legislation.

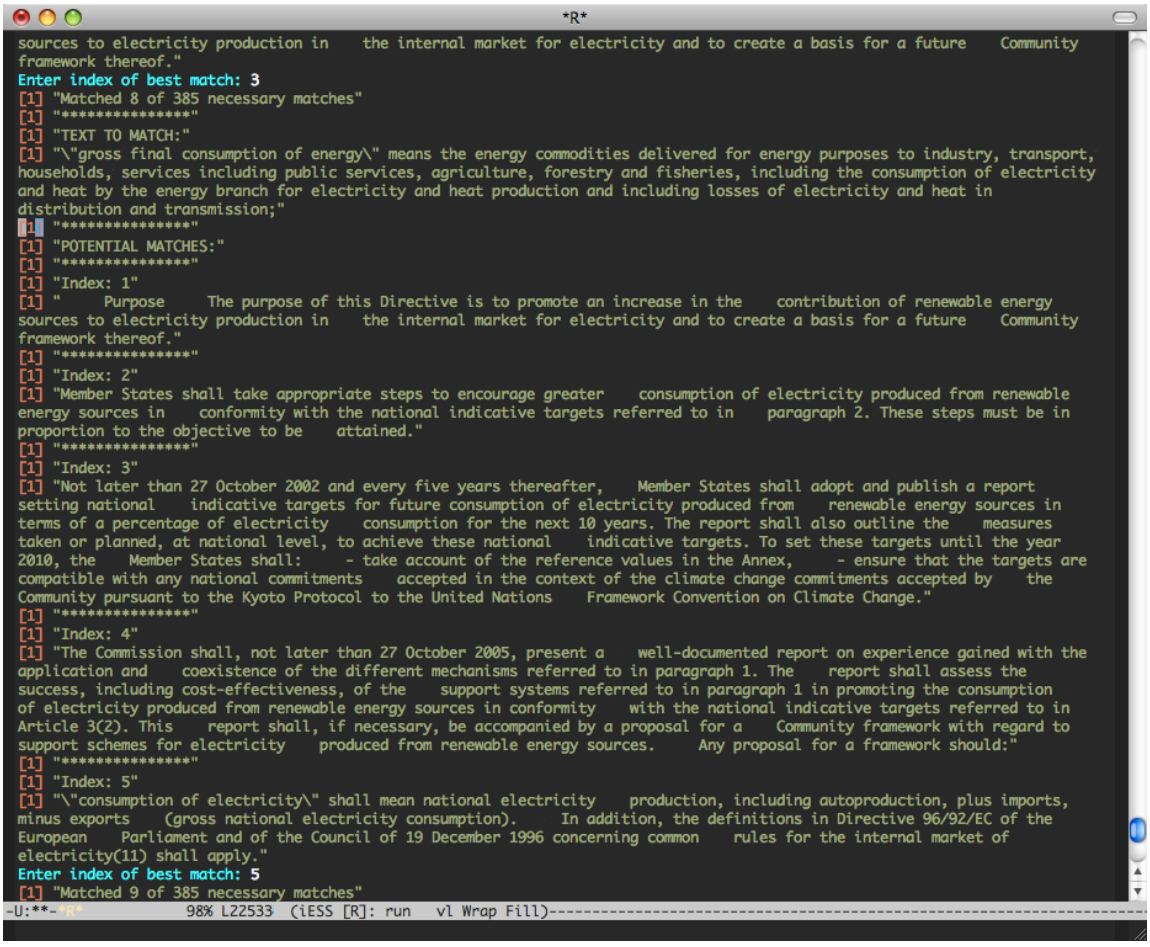


Figure 4: Sample of the `RunEncoder` interface, running in the R console.

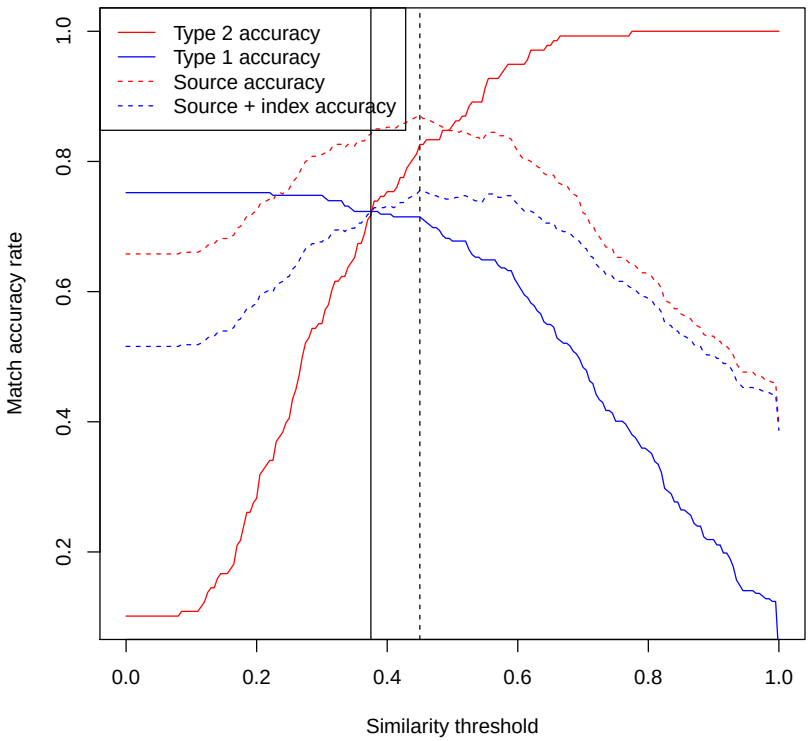


Figure 5: Algorithm accuracy as a function of the similarity threshold value. Baseline for comparison is a fully human-encoded bill. Source accuracy indicates where the algorithm identified the same source (amendment, original bill, or final bill) as the human coder. Source + index accuracy indicates where the algorithm found both the same source and the same index value (e.g. Amendment 24) as identified by the human coder.