Information Bottleneck Inspired Method For Chat Text Segmentation

S Vishal[1,⋆], **Mohit Yadav**[2,⋆,$], **Lovekesh Vig**[1] and **Gautam Shroff**[1]
[1]TCS Research New Delhi, India
[2]University of Massachusetts, Amherst
s.vishal3@tcs.com, ymohit@cs.umass.edu, lovekesh.vig@tcs.com, gautam.shroff@tcs.com

Abstract

We present a novel technique for segmenting chat conversations using the information bottleneck method (Tishby et al., 2000), augmented with sequential continuity constraints. Furthermore, we utilize critical non-textual clues such as time between two consecutive posts and people mentions within the posts. To ascertain the effectiveness of the proposed method, we have collected data from public Slack conversations and Fresco, a proprietary platform deployed inside our organization. Experiments demonstrate that the proposed method yields an absolute (relative) improvement of as high as 3.23% (11.25%). To facilitate future research, we are releasing manual annotations for segmentation on public Slack conversations.

1 Introduction

The prolific upsurge in the amount of chat conversations has notably influenced the way people wield languages for conversations. Moreover, conversation platforms have now become prevalent for both personal and professional usage. For instance, in a large enterprise scenario, project managers can utilize these platforms for various tasks such as decision auditing and dynamic responsibility allocation (Joty et al., 2013). Logs of such conversations offer potentially valuable information for various other applications such as automatic assessment of possible collaborative work among people (Rebedea et al., 2011).

It is thus vital to invent effective segmentation methods that can seperate discussions into small granules of independent conversational snippets. By 'independent', we meant a segment should as much as possible be self-contained and discussing the same topic, such that a segment can be suggested if any similar conversation occurs again. As an outcome of this, various short text similarity methods can be employed directly. Segmentation can also potentially act as an empowering preprocessing step for various down-streaming tasks such as automatic summarization (Dias et al., 2007), text generation (Barzilay and Lee, 2004), information extraction (Allan, 2012), and conversation visualization (Liu et al., 2012). It is worth noting that chat segmentation presents a number of gruelling challenges such as, the informal nature of the text, the frequently short length of the posts and a significant proportion of irrelevant interspersed text (Schmidt and Stone).

Research in text segmentation has a long history going back to the earliest attempt of Kozima (1993). Since then many methods, including but not limited to, Texttiling (Hearst, 1997), Choi's segmentation (Choi, 2000), representation learning based on semantic embeddings (Alemi and Ginsparg, 2015), and topic models (Du et al., 2015a) have been presented. Albeit, very little research effort has been proposed for segmenting informal chat text. For instance, Schmidt and Stone have attempted to highlight the challenges with chat text segmentation, though they have not presented any algorithm specific to chat text.

The Information Bottleneck (IB) method has been successfully applied to clustering in the NLP domain (Slonim and Tishby, 2000). Specifically, IB attempts to balance the trade-off between accuracy and compression (or complexity) while clustering the target variable, given a joint probability distribution between the target variable and an ob-

⋆ indicates that both authors contributed equally. $ indicates that the author was at TCS Research New-Delhi during the course of this work.

Proceedings of the The 8th International Joint Conference on Natural Language Processing, pages 194–203,
Taipei, Taiwan, November 27 – December 1, 2017 ©2017 AFNLP

served relevant variable. Similar to clustering, this paper interprets the task of text segmentation as a compression task with a constraint that allows only contiguous text snippets to be in a group.

The focus of this paper is to develop text segmentation methods for chat text utilizing the IB framework. In the process, this paper makes the following major contributions:

(i) We introduce an IB inspired objective function for the task of text segmentation.

(ii) We develop an agglomerative algorithm to optimize the proposed objective function that also respects the necessary sequential continuity constraint for text segmentation.

(iii) To the best of our knowledge, this paper is a first attempt that addresses segmentation for chat text and incorporates non-textual clues.

(iv) We have created a chat text segmentation dataset and releasing it for future research.

The remainder of this paper is organized as follows: we present a review of related literature in Section 2. Then, we formulate the text segmentation problem and define necessary notations in Section 3. Following this, we explain the proposed methodology in Section 4. Section 5 presents experiments and provides details on the dataset, experimental set-up, baselines, results, and effect of parameters. Finally, conclusions and potential directions for future work are outlined in Section 6.

2 Related Work

The IB method (Tishby et al., 2000) was originally introduced as a generalization of rate distortion theory which balances the tradeoff between the preservation of information about a relevance variable and the distortion of the target variable. Later on, similar to this work, a greedy bottom-up (agglomerative) IB based approach (Slonim and Tishby, 1999, 2000) has been successfully applied to NLP tasks such as document clustering.

Furthermore, the IB method has been widely studied for multiple machine learning tasks, including but not limited to, speech diarization (Vijayasenan et al., 2009), image segmentation (Bardera et al., 2009), image clustering (Gordon et al., 2003), and visualization (Kamimura, 2010). Particularly, similar to this paper, image segmentation has considered segmentation as the compression part of the IB based method. But, image segmentation does not involve continuity constraints as their application can abolish the exploitation of similarity within the image. Yet another similar attempt that utilizes information theoretic terms as an objective (only the first term of the IB approach) has been made for the task of text segmentation and alignment (Sun et al., 2006).

Broadly stating, a typical text segmentation method comprises of a method that: (a) consumes text representations for every independent text snippet, and (b) applies a search procedure for segmentation boundaries while optimizing objectives for segmentation. Here, we review literature of text segmentation by organizing them into 3 categories based on their focus: *Category1* - (a), *Category2* - (b), and *Category3* - both (a) and (b).

Category1 approaches utilize or benefit from a great amount of effort put in developing robust topic models that can model discourse in natural language texts (Brants et al., 2002). Recently, Du et al. (2013, 2015b) have proposed a hierarchical Bayesian model for unsupervised topic segmentation that integrates a point-wise boundary sampling algorithm used in Bayesian segmentation into a structured (ordering-based) topic model. For a more comprehensive view of classical work on topic models for text segmentation, we refer to Misra et al. (2009); Riedl and Biemann (2012). This work does not explore topic models and is left as a direction for future research.

Category2 approaches comprise of different search procedures proposed for the task of text segmentation, including but not limited to, divisive hierarchical clustering (Choi, 2000), dynamic programming (Kehagias et al., 2003), and graph based clustering (Pourvali and Abadeh, 2012; Glavas et al., 2016; Utiyama and Isahara, 2001). This work proposes an agglomerative IB based hierarchical clustering algorithm - an addition to the arsenal of the approaches that falls in this category.

Similar to the proposed method, *Category3* cuts across both of the above introduced dimensions of segmentation. Alemi and Ginsparg (2015) have proposed the use of semantic word embeddings and a relaxed dynamic programming procedure. We have also argued to utilize chat clues and introduced an IB based approach augmented with sequential continuity constraints. Yet another similar attempt has been made by Joty et al. (2013) in which they use topical and conversational clues and introduce an unsupervised random walk model for the task of text segmentation.

Beyond the above mentioned categorization, a significant amount of research effort has been put up in studying the evaluation metric for text segmentation (Pevzner and Hearst, 2002; Scaiano and Inkpen, 2012). Here, we make use of the classical and most widely utilized metric introduced by Beeferman et al. (1999). Also, there have been attempts to track topic boundaries for thread discussions (Zhu et al., 2008; Wang et al., 2008). While these methods look similar to the proposed method, they differ as they attempt to recover thread structure with respect to the topic level view of the discussions within a thread community.

The most similar direction of research to this work is on conversation trees (Louis and Cohen, 2015) and disentangling chat conversations (Elsner and Charniak, 2010). Both of these directions cluster independent posts leading to topic labelling and segmentation of these posts simultaneously. It is important to note that these methods do not have a sequential continuity constraint and consider lexical similarity even between long distant posts (Elsner and Charniak, 2011). Moreover, if these methods are applied only for segmentation then they are very likely to produce segments with relatively very smaller durations; as reflected in the ground truth annotations of correspondingly released dataset (Elsner and Charniak, 2008). It is worth noting that Elsner and Charniak (2010) have also advocated to utilize time gap and people mentions similar to the proposed method of this work.

3 Problem Description And Notations

Let C be an input chat text sequence $C = \{c_1, ..., c_i, ..., c_{|t|}\}$ of length $|C|$, where c_i is a text snippet such as a sentence or a post from chat text. In a chat scenario, text post c_i will have a corresponding time-stamp c_i^t. A segment or a subsequence can be represented as $C_{a:b} = \{c_a, ..., c_b\}$. A segmentation of C is defined as a segment sequence $S = \{s_1, ..., s_p\}$, where $s_j = C_{a_j:b_j}$ and $b_j + 1 = a_{j+1}$. Given an input text sequence C, the segmentation is defined as the task of finding the most probable segment sequence S.

4 Proposed Methodology

This section firstly presents the proposed IB inspired method for text segmentation that conforms to the necessary constraint of sequential continuity, in Section 4.1. Next, in Section 4.2, the proposed IB inspired method is augmented to incorporate important non-textual clues that arise in a chat scenario. More specifically, the time between two consecutive posts and people mentions within the posts are integrated into the proposed IB inspired approach for the text segmentation task.

4.1 IB Inspired Text Segmentation Algorithm

The IB introduces a set of relevance variables R which encapsulate meaningful information about C while compressing the data points (Slonim and Tishby, 2000). Similarly, we propose that a segment sequence S should also contain as much information as possible about R (i.e., maximize $I(R, S)$), constrained by mutual information between S and C (i.e., minimize $I(S, C)$). Here, C is a chat text sequence, following the notation introduced in the previous section. The IB objective can be achieved by maximizing the following:

$$F = I(R, S) - \frac{1}{\beta} \times I(S, C) \tag{1}$$

In other words, the above IB objective function attempts to balance a trade-off between the most informative segmentation of R and the most compact representation of C; where β is a constant parameter to control the relative importance.

Similar to Tishby et al. (2000), we model R as word clusters and optimize F in an agglomerative fashion, as explained in Algorithm 1. In simple words, the maximization of F boils down to agglomeratively merging an adjacent pair of posts that correspond to least value of d. In Algorithm 1, $p(\overline{s})$ is equal to $p(s_i) + p(s_{i+1})$ and $d(s_i, s_{i+1})$ is computed using the following definition:

$$\begin{aligned} d(s_i, s_{i+1}) = JSD[p(R|s_i), p(R|s_{i+1})] - \\ \frac{1}{\beta} \times JSD[p(C|s_i), p(C|s_{i+1})] \end{aligned} \tag{2}$$

Here, JSD indicates Jensen-Shannon-Divergence. The computation of R and $p(R, C)$ is explained later in Section 5.2. Stopping criterion for Algorithm 1 is $SC > \theta$, where SC is computed as follows:

$$SC = \frac{I(R, S)}{I(R, C)} \tag{3}$$

The value of SC is expected to decrease due to a relatively large dip in the value of $I(R, S)$ when

Algorithm 1: IB inspired text segmentation

```
Input          : Joint distribution: $p(R, C)$,
                 Tradeoff parameter: $\beta$
Output         : Segmentation sequence: $S$
Initialization: $S \leftarrow C$
                 Calculate $\Delta F(s_i, s_{i+1}) =$
$p(\overline{s}) \times\ d(s_i, s_{i+1})\ \forall\ s_i \in S$
1 while Stopping criterion is false do
2     $\{i\} = argmin_{i'} \Delta F(s_{i'}, s_{i'+1})$;
3     Merge $\{s_i, s_{i+1}\} \Rightarrow \overline{s} \in S$;
4     Update $\Delta F(\overline{s}, s_{i-1})$ and $\Delta F(\overline{s}, s_{i+2})$;
5 end
```

more dissimilar clusters are merged. Therefore, SC provides strong clues to terminate the proposed IB approach. The inspiration behind this specific computation of SC has come from the fact that it has produced stable results when experimented with a similar task of speaker diarization (Vijayasenan et al., 2009). The value of θ is tuned by optimizing the performance over a validation dataset just like other hyper-parameters.

The IB inspired text segmentation algorithm (Algorithm 1) respects the sequential continuity constraint, as it considers merging only adjacent pairs (see step 2, 3, and 4 of Algorithm 1) while optimizing F; unlike the agglomerative IB clustering (Slonim and Tishby, 2000). As a result of this, the proposed IB based approach requires a limited number of involved computations, more precisely, linear in terms of number of text snippets.

4.2 Incorporating Non-Textual Clues

As mentioned above, we submit that non-textual clues (such as time between two consecutive posts and people mentions within the posts) are critical for segmenting chat text. To incorporate these two important clues, we augment Algorithm 1, developed in the previous section. More precisely, we modify d of Equation 2 to $\overline{d}$ as follows:

$$\overline{d}(s_i, s_{i+1}) = w_1 \times d(s_i, s_{i+1}) + w_2 \times (c^t_{a_{i+1}} - c^t_{b_i}) + w_3 \times ||s^p_i - s^p_{i+1}|| \tag{4}$$

Here, $c^t_{a_{i+1}}$, $c^t_{b_i}$ and s^p_i represent time-stamp of the first post of segment s_{i+1}, time-stamp of last post of segment s_i, and representation for poster information embedded in segment s_i, respectively. The s^p_i representation is computed as a bag of posters counting all the people mentioned in the posts and posters themselves in a segment. w_1, w_2, w_3 are weights indicating the relative importance of distance terms computed for all three different clues. $||.||$ in Equation 4 indicates euclidean norm.

It is important to note that Algorithm 1 utilizes d of Equation 2 to represent textual dissimilarity between a pair of posts in order to achieve the optimal segment sequence S. Following the same intuition, $\overline{d}$ in Equation 4 measures weighted distances based not only on textual similarity but also based on information in time-stamps, posters and people mentioned. The intuition behind the second distance term in $\overline{d}$ is that if the time difference between two posts is small then they are likely to be in the same segment. Additionally, the third distance term in $\overline{d}$ is intended to merge segments that involve a higher number of common posters and people mentions. Following the same intuition, in addition to the changes in $\overline{d}$, we modify the stopping criterion as well while the rest stays the same as in Algorithm 1. The stopping criterion is defined as $SC > \theta$, where SC is as follows:

$$SC = w_1 \times \frac{I(R,S)}{I(R,C)} + w_2 \times (1 - \frac{G(S)}{G_{max}}) + w_3 \times \frac{H(S)}{H_{max}} \tag{5}$$

Here, the $G(S)$ and $H(S)$ mentioned in Equation 5 are computed as follows:

$$G(S) = \sum_{s_i \in S} c^t_{b_i} - c^t_{a_i} \tag{6}$$

$$H(S) = \sum_{i=1}^{|S|} ||s^p_i - s^p_{i+1}|| \tag{7}$$

The first term in SC in Equation 5 is taken from the stopping criterion of Algorithm 1 and the remaining second and third terms are similarly derived. Both the second and third terms decrease as the cardinality of S is decreased and reflect analogous behaviour to the two introduced important clues. The first term computes the fraction of information contained in S about R, normalized by the information contained in C about R; similarly, the second term computes the fraction of time duration between segments normalized by total duration of chat text sequence (i.e. 1 - fraction of durations of all segments normalized by total duration), and the third term computes the sum of

	Slack	*Fresco*
# Threads	4	46
# Posts	9000	5000
# Segments	900	800
# Documents	73	73

Table 1: Statistics of the chat datasets.

inter segment distances in terms of poster information normalized by the maximum distance of similar terms (i.e. when each post is a segment).

5 Experiments

This section starts with a description of the datasets collected from the real world conversation platforms in Subsection 5.1. Later in Subsection 5.2, we explain the evaluation metric utilized in our experiments. Subsection 5.3 describes the meaningful baselines developed for a fair comparison with the proposed IB approach. Next in Subsections 5.4 and 5.5, we discuss the performance accomplished by the proposed approach on both of the collected datasets. Lastly, we analyse the stability of the proposed IB approach with respect to parameters β and θ in Subsection 5.6.

5.1 Dataset Description

We have collected chat text datasets, namely, *Slack* and *Fresco*, respectively from http://slackarchive.io/ and http://talk.fresco.me/. After that, we have manually annotated them for the text segmentation task. We have utilized the annotations done by 3 workers with problematic cases resolved by consensus. Datasets' statistics is mentioned in Table 1. The collected raw data was in the form of threads, which was later divided into segments. Further, we have created multiple documents where each document contains N continuous segments from the original threads. N was selected randomly between 5 and 15. 60% of these documents were used for tuning hyper-parameters which include weights (w_1, w_2, w_3), θ, and β; and the remaining were used for testing.

A small portion of one of the documents from the *Slack* dataset is depicted in Figure 1(a). Here, manual annotations are marked by a bold black horizontal line, and also enumerated as 1), 2), and 3). Every text line is a post made by one of the users on the *Slack* platform during conversations. As mentioned above, in a chat scenario, every post has following three integral components:

1) poster (indicated by corresponding identity in Figure 1, from beginning till ‘-=[*_says_’),
2) time-stamp (between ‘-=[*’ and ‘*]=-)’, and
3) textual content (after ‘*]=-::: ’till end).
One must also notice that some of the posts also have people mentions within the posts (indicated as ‘<@USERID>’ in Figure 1).

To validate the differences between the collected chat datasets and traditional datasets such as Choi's dataset (Choi, 2000), we computed the fraction of words occurring with a frequency less than a given word frequency, as shown in Figure 2. It is clearly evident from the Figure 2 that chat segmentation datasets have a significantly high proportion of less frequent words in comparison to the traditional text segmentation datasets. The presence of large infrequent words makes it hard for textual similarity methods to succeed as it will increase the proportion of out of vocabulary words (Gulcehre et al., 2016). Therefore, it becomes even more critical to utilize the non-textual clues while processing chat text.

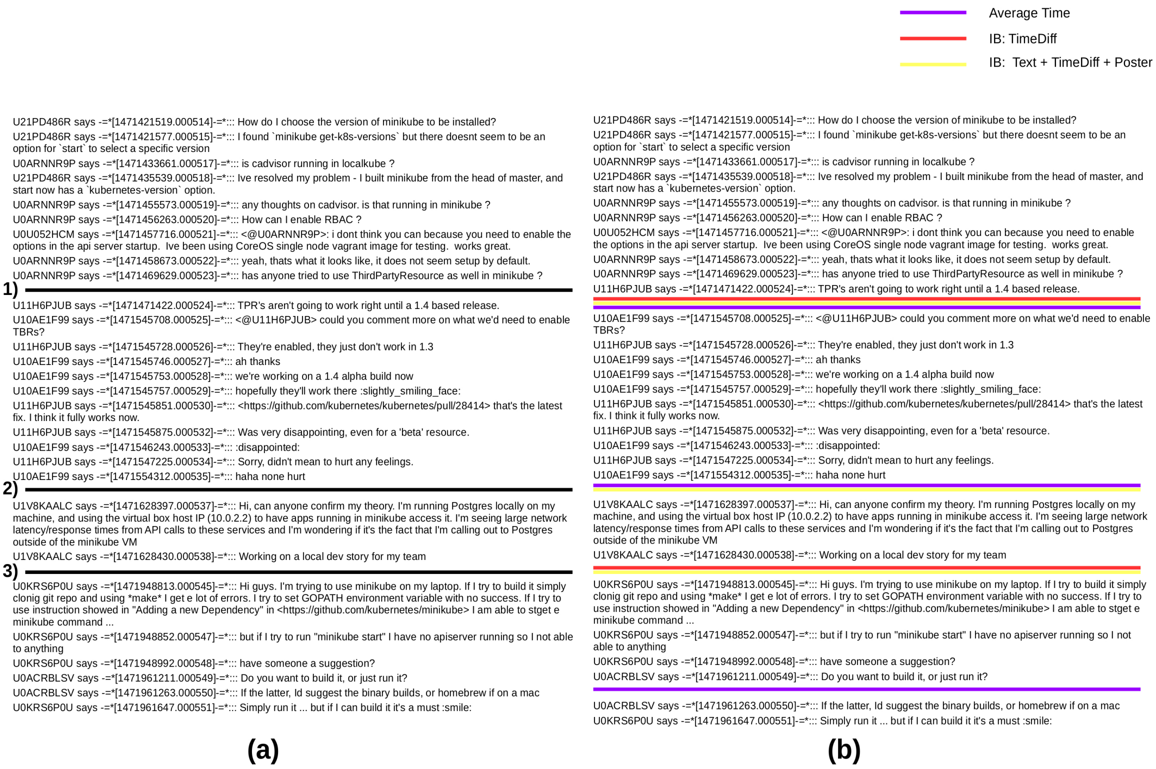
5.2 Evaluation and Setup

For performance evaluation, we have employed P_k metric (Beeferman et al., 1999) which is widely utilized for evaluating the text segmentation task. A sliding window of fixed size k (usually half of the average of length of all the segments in the document) slides over the entire document from top to bottom. Both inter and intra segment errors for all posts k apart is calculated by comparing inferred and annotated boundaries.

We model the set of relevance variables R as word clusters estimated by utilizing agglomerative IB based document clustering (Slonim and Tishby, 2000) where posts are treated as relevance variables. Consequently, R comprises of informative word clusters about posts. Thus, each entry $p(r_i, c_j)$ in matrix $p(R, C)$ represents the joint probability of getting a word cluster r_i in post c_j. We calculate $p(r_i, c_j)$ simply by counting the common words in r_i and c_j and then normalizing.

5.3 Baseline Approaches

For comparisons, we have developed multiple baselines. In *Random*, 5 to 15 boundaries are inserted randomly. In case of *No Boundary*, the entire document is labelled as one segment. Next, we implemented *C-99* and *Dynamic Programming*, which are classical benchmarks for the text segmentation task. Another very simple and yet effec-



```
U21PD486R says -=*[1471421519.000514]-=*::: How do I choose the version of minikube to be installed?
U21PD486R says -=*[1471421577.000515]-=*::: I found `minikube get-k8s-versions` but there doesnt seem to be an option for `start` to select a specific version
U0ARNNR9P says -=*[1471433661.000517]-=*::: is cadvisor running in localkube ?
U21PD486R says -=*[1471435539.000518]-=*::: Ive resolved my problem - I built minikube from the head of master, and start now has a `kubernetes-version` option.
U0ARNNR9P says -=*[1471455573.000519]-=*::: any thoughts on cadvisor. is that running in minikube ?
U0ARNNR9P says -=*[1471456263.000520]-=*::: How can I enable RBAC ?
U0U052HCM says -=*[1471457716.000521]-=*::: <@U0ARNNR9P>: i dont think you can because you need to enable the options in the api server startup.  Ive been using CoreOS single node vagrant image for testing.  works great.
U0ARNNR9P says -=*[1471458673.000522]-=*::: yeah, thats what it looks like, it does not seem setup by default.
U0ARNNR9P says -=*[1471469629.000523]-=*::: has anyone tried to use ThirdPartyResource as well in minikube ?
U11H6PJUB says -=*[1471471422.000524]-=*::: TPR's aren't going to work right until a 1.4 based release.
U10AE1F99 says -=*[1471545708.000525]-=*::: <@U11H6PJUB> could you comment more on what we'd need to enable TBRs?
U11H6PJUB says -=*[1471545728.000526]-=*::: They're enabled, they just don't work in 1.3
U10AE1F99 says -=*[1471545746.000527]-=*::: ah thanks
U10AE1F99 says -=*[1471545753.000528]-=*::: we're working on a 1.4 alpha build now
U10AE1F99 says -=*[1471545757.000529]-=*::: hopefully they'll work there :slightly_smiling_face:
U11H6PJUB says -=*[1471545851.000530]-=*::: <https://github.com/kubernetes/kubernetes/pull/28414> that's the latest fix. I think it fully works now.
U11H6PJUB says -=*[1471545875.000532]-=*::: Was very disappointing, even for a 'beta' resource.
U10AE1F99 says -=*[1471546243.000533]-=*::: :disappointed:
U11H6PJUB says -=*[1471547225.000534]-=*::: Sorry, didn't mean to hurt any feelings.
U10AE1F99 says -=*[1471554312.000535]-=*::: haha none hurt
U1V8KAALC says -=*[1471628397.000537]-=*::: Hi, can anyone confirm my theory. I'm running Postgres locally on my machine, and using the virtual box host IP (10.0.2.2) to have apps running in minikube access it. I'm seeing large network latency/response times from API calls to these services and I'm wondering if it's the fact that I'm calling out to Postgres outside of the minikube VM
U1V8KAALC says -=*[1471628430.000538]-=*::: Working on a local dev story for my team
U0KRS6P0U says -=*[1471948813.000545]-=*::: Hi guys. I'm trying to use minikube on my laptop. If I try to build it simply clonig git repo and using *make* I get e lot of errors. I try to set GOPATH environment variable with no success. If I try to use instruction showed in "Adding a new Dependency" in <https://github.com/kubernetes/minikube> I am able to stget e minikube command ...
U0KRS6P0U says -=*[1471948852.000547]-=*::: but if I try to run "minikube start" I have no apiserver running so I not able to anything
U0KRS6P0U says -=*[1471948992.000548]-=*::: have someone a suggestion?
U0ACRBLSV says -=*[1471961211.000549]-=*::: Do you want to build it, or just run it?
U0ACRBLSV says -=*[1471961263.000550]-=*::: If the latter, Id suggest the binary builds, or homebrew if on a mac
U0KRS6P0U says -=*[1471961647.000551]-=*::: Simply run it ... but if I can build it it's a must :smile:
```

```
U21PD486R says -=*[1471421519.000514]-=*::: How do I choose the version of minikube to be installed?
U21PD486R says -=*[1471421577.000515]-=*::: I found `minikube get-k8s-versions` but there doesnt seem to be an option for `start` to select a specific version
U0ARNNR9P says -=*[1471433661.000517]-=*::: is cadvisor running in localkube ?
U21PD486R says -=*[1471435539.000518]-=*::: Ive resolved my problem - I built minikube from the head of master, and start now has a `kubernetes-version` option.
U0ARNNR9P says -=*[1471455573.000519]-=*::: any thoughts on cadvisor. is that running in minikube ?
U0ARNNR9P says -=*[1471456263.000520]-=*::: How can I enable RBAC ?
U0U052HCM says -=*[1471457716.000521]-=*::: <@U0ARNNR9P>: i dont think you can because you need to enable the options in the api server startup.  Ive been using CoreOS single node vagrant image for testing.  works great.
U0ARNNR9P says -=*[1471458673.000522]-=*::: yeah, thats what it looks like, it does not seem setup by default.
U0ARNNR9P says -=*[1471469629.000523]-=*::: has anyone tried to use ThirdPartyResource as well in minikube ?
U11H6PJUB says -=*[1471471422.000524]-=*::: TPR's aren't going to work right until a 1.4 based release.
U10AE1F99 says -=*[1471545708.000525]-=*::: <@U11H6PJUB> could you comment more on what we'd need to enable TBRs?
U11H6PJUB says -=*[1471545728.000526]-=*::: They're enabled, they just don't work in 1.3
U10AE1F99 says -=*[1471545746.000527]-=*::: ah thanks
U10AE1F99 says -=*[1471545753.000528]-=*::: we're working on a 1.4 alpha build now
U10AE1F99 says -=*[1471545757.000529]-=*::: hopefully they'll work there :slightly_smiling_face:
U11H6PJUB says -=*[1471545851.000530]-=*::: <https://github.com/kubernetes/kubernetes/pull/28414> that's the latest fix. I think it fully works now.
U11H6PJUB says -=*[1471545875.000532]-=*::: Was very disappointing, even for a 'beta' resource.
U10AE1F99 says -=*[1471546243.000533]-=*::: :disappointed:
U11H6PJUB says -=*[1471547225.000534]-=*::: Sorry, didn't mean to hurt any feelings.
U10AE1F99 says -=*[1471554312.000535]-=*::: haha none hurt
U1V8KAALC says -=*[1471628397.000537]-=*::: Hi, can anyone confirm my theory. I'm running Postgres locally on my machine, and using the virtual box host IP (10.0.2.2) to have apps running in minikube access it. I'm seeing large network latency/response times from API calls to these services and I'm wondering if it's the fact that I'm calling out to Postgres outside of the minikube VM
U1V8KAALC says -=*[1471628430.000538]-=*::: Working on a local dev story for my team
U0KRS6P0U says -=*[1471948813.000545]-=*::: Hi guys. I'm trying to use minikube on my laptop. If I try to build it simply clonig git repo and using *make* I get e lot of errors. I try to set GOPATH environment variable with no success. If I try to use instruction showed in "Adding a new Dependency" in <https://github.com/kubernetes/minikube> I am able to stget e minikube command ...
U0KRS6P0U says -=*[1471948852.000547]-=*::: but if I try to run "minikube start" I have no apiserver running so I not able to anything
U0KRS6P0U says -=*[1471948992.000548]-=*::: have someone a suggestion?
U0ACRBLSV says -=*[1471961211.000549]-=*::: Do you want to build it, or just run it?
U0ACRBLSV says -=*[1471961263.000550]-=*::: If the latter, Id suggest the binary builds, or homebrew if on a mac
U0KRS6P0U says -=*[1471961647.000551]-=*::: Simply run it ... but if I can build it it's a must :smile:
```

Figure 1: (a) Manually created ground truth for *Slack* public conversations. Black color lines represents segmentation boundaries. (b) Results obtained for multiple approaches. Text best read magnified.

Methods	Span of Weights	*Slack*	*Fresco*
Random	–	60.6	54
No Boundary	–	36.76	45
Average Time	–	32	35
C-99	–	35.18	37.75
Dynamic Programming	–	28.7	35
Encoder-Decoder Distance	–	29	38
LDA Distance	–	36	44
IB Variants:			
Text	$w_1 = 1, w_2 = 0, w_3 = 0$	33	42
TimeDiff	$w_1 = 0, w_2 = 1, w_3 = 0$	**26.75**	**34.25**
Poster	$w_1 = 0, w_2 = 0, w_3 = 1$	34.52	41.50
Text + TimeDiff	$\forall w \in \{w_1, w_2\}, w \in (0,1); w_3 = 0; w_1 + w_2 = 1$	**26.47**	**34.68**
Text + Poster	$\forall w \in \{w_1, w_3\}, w \in (0,1); w_2 = 0; w_1 + w_3 = 1$	**28.57**	38.21
Text+TimeDiff+Poster	$\forall w \in \{w_1, w_2, w_3\}, w \in (0,1); w_1 + w_2 + w_3 = 1$	**25.47**	**34.80**

Table 2: Performance evaluation: P_k metric [in terms of % error] for various methods. Lower is better.

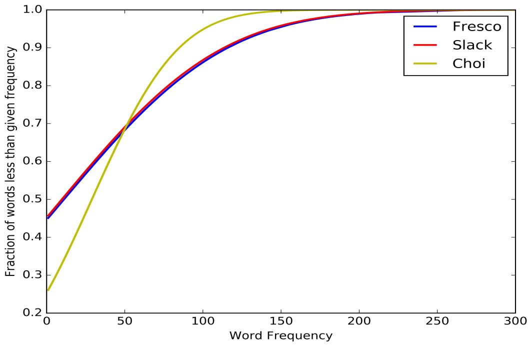


Figure 2: Fraction of words less than a given word frequency.

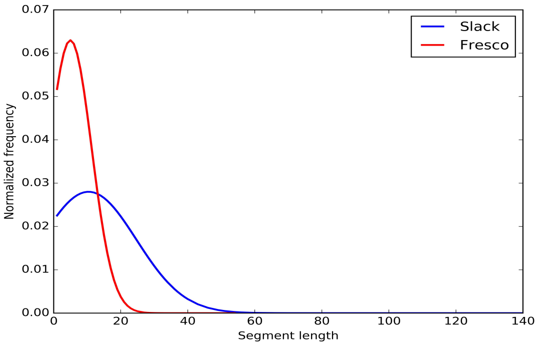


Figure 3: Normalized frequency distribution of segment length for both the collected chat datasets.

tive baseline *Average Time* is prepared, in which boundaries are inserted after a fixed amount of time has elapsed. Fixed time is calculated from a certain separate portion of our annotated dataset.

Next baseline utilized in our experiments is *Encoder-Decoder Distance*. In this approach, we have trained a sequence to sequence RNN encoder-decoder (Sutskever et al., 2014) utilizing 1.5 million posts from the publicly available *Slack* dataset excluding the labelled portion. The network comprises of 2 hidden layers and the hidden state dimension was set to 256 for each. The encoded representation was utilized and greedily merged in an agglomerative fashion using Euclidean distance. The stopping criterion for this approach was similar to the third term in Equation 5 corresponding to poster information. The optimization of hidden state dimension was computationally demanding hence left for further exploration in future. Similar to *Encoder-Decoder Distance*, we have developed *LDA Distance* where representations have come from a topic model (Blei et al., 2003) having 100 topics.

5.4 Quantitative Results

The results for all prepared baselines and variants of IB on both *Slack* and *Fresco* datasets are mentioned in Table 2. For both *Slack* and *Fresco* datasets, multiple variants of IB yield superior performance when compared against all the developed baselines. More precisely, for *Slack* dataset, 4 different variants of the proposed IB based method achieve higher performance with an absolute improvement of as high as $\mathbf{3.23}\%$ and a relative improvement of $\mathbf{11.25}\%$, when compared against the baselines. In case of *Fresco* dataset, 3 different variants of the proposed method achieve superior performance but not as significantly in terms of absolute P_k value, as they do for the *Slack* dataset. We hypothesize that such a behaviour is potentially because of the lesser value of posts per segment for *Fresco* (5000/800=6.25) in comparison to *Slack* (9000/900=10). Also, note that just the time clue in IB framework performs best on *Fresco* dataset indicating that the relative importance of time clue will be higher for a dataset with smaller lengths of segments (i.e. low value of posts per segment). To validate our hypothesize further, we estimate the normalized frequency distribution of segment length (number of posts per segment) for both datasets, as shown in Figure 3.

It is worth noting that the obtained empirical results support the major hypothesis of this work. As variants of IB yield superior performance on both the datasets. Also, on incorporation of individual non-textual clues, superior improvements of 3.23% and 7.32% are observed from *Text* to *Text+TimeDiff* for *Slack* and *Fresco*, respectively; similarly, from *Text* to *Text+Poster* improvements of 4.43% and 3.79% are observed for *Slack* and *Fresco*, respectively. Further, the best performance is achieved for both the datasets on fusing both the non-textual clues indicating that clues are complementary as well.

5.5 Qualitative Results

Results obtained for multiple approaches, namely, *Average Time*, *IB:TimeDiff*, and *IB:Text+TimeDiff+Poster*, corresponding to a small portion of chat text placed in part (a) of Figure 1 are presented in part (b) of Figure 1. *Average Time* baseline (indicated by purple) managed to find three boundaries, albeit one of the boundary is significantly off, potentially due to the constraint of fixed time duration.

Similarly, the next *IB:TimeDiff* approach also manages to find first two boundaries correctly but fails to recover the third boundary. Results seem to indicate that the time clue is not very effective to reconstruct segmentation boundaries when segment length varies a lot within the document. Interestingly, the combination of all three clues as happens in the *IB:Text+TimeDiff+Poster* approach, yielded the best results as all of three segmentation boundaries in ground truth are recovered with high precision. Therefore, we submit that the incorporation of non-textual clues is critical to achieve superior results to segment chat text.

5.6 Effect Of Parameters

To analyse the behaviour of the proposed IB based methods, we compute the average performance metric P_k of *IB:Text* with respect to β and θ, over the test set of *Slack* dataset. Also, to facilitate the reproduction of results, we mention optimal values of all the parameters for all the variants of the proposed IB approach in Table 5.5.

Figure 4 shows the behaviour of the average of performance evaluation metric P_k over the test set of *Slack* dataset with respect to hyper-parameter β. As mentioned above also, the parameter β represents a trade-off between the preserved amount of information and the level of compression. It is clearly observable that the optimal value of β does not lie on extremes indicating the importance of both the terms (as in Equation 1) of the proposed IB method. The coefficient of the second term (i.e. $\frac{1}{\beta}$ equals to 10^{-3}) is smaller. One could interpret the behaviour of thr second term as a regularization term because $\frac{1}{\beta}$ controls the complexity of the learnt segment sequence S. Furthermore, optimal values in Table 5.5 for variants with fusion of two or more clues indicate complementary and relative importance of the studied non-textual clues.

The average performance evaluation metric P_k over test set of the *Slack* dataset with respect to hyper-parameter θ is depicted in Figure 5. Figure 5 makes the appropriateness of the stopping criterion clearly evident. Initially, the average of P_k value decreases as more coherent posts are merged and continues to decrease till it is less than a particular value of θ. After that, the average of P_k value starts increasing potentially due to the merging of more dissimilar segments. The optimal values of θ varies significantly from one variant to another requiring a mandatory tuning over the validation dataset, as mentioned in Table 5.5, for all *IB* variants proposed in this work.

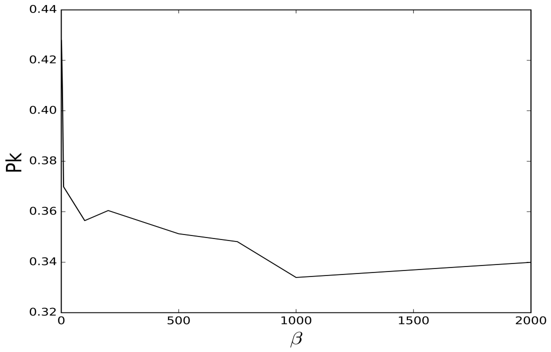


Figure 4: Average evaluation metric P_k over *Slack* dataset with respect to hyper-parameter β.

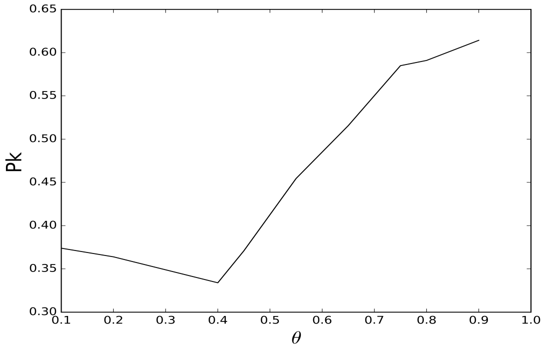


Figure 5: Average evaluation metric P_k over *Slack* dataset with respect to hyper-parameter θ.

6 Discussion And Future Work

We started by highlighting the increasing importance of efficient methods to process chat text, in particular for text segmentation. We have collected and introduced datasets for the same. Our introduction of chat text datasets has enabled us to explore segmentation approaches that are specific to chat text. Further, our results demonstrate that the proposed IB method yields an absolute improvement of as high as $\mathbf{3.23\%}$. Also, a significant boost ($\mathbf{3.79\%}$-$\mathbf{7.32\%}$) in performance is observed on incorporation of non-textual clues indicating their criticality. In future, it will be interesting to investigate the possibility of incorporat-

IB Variants:	Slack			Fresco		
	β	(w_1, w_2, w_3)	θ	β	(w_1, w_2, w_3)	θ
Text	1000	(1,0,0)	0.4	1000	(1,0,0)	0.5
TimeDiff	750	(0,1,0)	0.9	750	(0,1,0)	0.9
Poster	750	(0,0,1)	0.09	750	(0,0,1)	0.1
Text+TimeDiff	750	(0.3,0.7,0)	0.75	750	(0.3,0.7,0)	0.75
Text+Poster	750	(0.1,0,0.9)	0.2	∞	(0.3,0,0.7)	0.2
Text+TimeDiff+Poster	750	(0.24,0.58,0.18)	0.65	750	(0.10,0.63,0.27)	0.65

Table 3: Optimal values of parameters corresponding to results obtained by IB variants in Table 2.

ing semantic word embeddings in the proposed IB method (Alemi and Ginsparg, 2015).

References

Alexander A Alemi and Paul Ginsparg. 2015. Text segmentation based on semantic word embeddings. *arXiv preprint arXiv:1503.05543*.

James Allan. 2012. *Topic detection and tracking: event-based information organization*, volume 12. Springer Science & Business Media.

Anton Bardera, Jaume Rigau, Imma Boada, Miquel Feixas, and Mateu Sbert. 2009. Image segmentation using information bottleneck method. *IEEE Transactions on Image Processing*, 18(7):1601–1612.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. *arXiv preprint arXiv: 0405039*.

Doug Beeferman, Adam Berger, and John Lafferty. 1999. Statistical models for text segmentation. *Machine Learning*, 34(1-3):177–210.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Thorsten Brants, Francine Chen, and Ioannis Tsochantaridis. 2002. Topic-based document segmentation with probabilistic latent semantic analysis. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, CIKM ’02, pages 211–218.

Freddy Y. Y. Choi. 2000. Advances in domain independent linear text segmentation. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, pages 26–33.

Gaël Dias, Elsa Alves, and José Gabriel Pereira Lopes. 2007. Topic segmentation algorithms for text summarization and passage retrieval: An exhaustive evaluation. In *Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 2*, AAAI’07, pages 1334–1339. AAAI Press.

Lan Du, Wray L. Buntine, and Mark Johnson. 2015a. Topic segmentation with a structured topic model. In *HLT-NAACL*. The Association for Computational Linguistics.

Lan Du, John K Pate, and Mark Johnson. 2013. Topic segmentation with a structured topic model. In *Proceedings of the North American Chapter of the Association for Computational Linguistics Conference*, pages 190–200.

Lan Du, John K Pate, and Mark Johnson. 2015b. Topic segmentation with an ordering-based topic model. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2232–2238.

Micha Elsner and Eugene Charniak. 2008. You talking to me? a corpus and algorithm for conversation disentanglement. pages 834–842.

Micha Elsner and Eugene Charniak. 2010. Disentangling chat. *Computation Linguistics*, 36:389–409.

Micha Elsner and Eugene Charniak. 2011. Disentangling chat with local coherence models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 1179–1189.

Goran Glavas, Federico Nanni, and Simone Paolo Ponzetto. 2016. Unsupervised text segmentation using semantic relatedness graphs. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics (*SEM 2016)*, pages 125–130.

Shiri Gordon, Hayit Greenspan, and Jacob Goldberger. 2003. Applying the information bottleneck principle to unsupervised clustering of discrete and continuous image representations. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV ’03, pages 370–, Washington, DC, USA. IEEE Computer Society.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. *arXiv preprint arXiv:1603.08148*.

Marti A Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics*, 23(1):33–64.

Shafiq Joty, Giuseppe Carenini, and Raymond T Ng. 2013. Topic segmentation and labeling in asynchronous conversations. *Journal of Artificial Intelligence Research*, 47:521–573.

Ryotaro Kamimura. 2010. Information-theoretic enhancement learning and its application to visualization of self-organizing maps. *Neurocomputing*, 73(13):2642–2664.

Athanasios Kehagias, Fragkou Pavlina, and Vassilios Petridis. 2003. Linear text segmentation using a dynamic programming algorithm. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 171–178. Association for Computational Linguistics.

Hideki Kozima. 1993. Text segmentation based on similarity between words. In *Proceedings of the 31st Annual Meeting on Association for Computational Linguistics*, ACL '93, pages 286–288.

Shixia Liu, Michelle X Zhou, Shimei Pan, Yangqiu Song, Weihong Qian, Weijia Cai, and Xiaoxiao Lian. 2012. Tiara: Interactive, topic-based visual text summarization and analysis. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(2):25.

Annie Louis and Shay B. Cohen. 2015. Conversation trees: A grammar model for topic structure in forums. In *Proceedings of the Empirical Methods on Natural Language Processing*, pages 1543–1553. The Association for Computational Linguistics.

Hemant Misra, François Yvon, Joemon M. Jose, and Olivier Cappe. 2009. Text segmentation via topic modeling: An analytical study. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 1553–1556.

Lev Pevzner and Marti A Hearst. 2002. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28:19–36.

Mohsen Pourvali and Ph D Mohammad Saniee Abadeh. 2012. A new graph based text segmentation using wikipedia for automatic text summarization. *Editorial Preface*, 3(1).

Traian Rebedea, Mihai Dascalu, Stefan Trausan-Matu, Gillian Armitt, and Costin Chiru. 2011. Automatic assessment of collaborative chat conversations with polycafe. In *European Conference on Technology Enhanced Learning*, pages 299–312. Springer.

Martin Riedl and Chris Biemann. 2012. Text segmentation with topic models. In *Journal for Language Technology and Computational Linguistics*, volume 27.1, pages 47–69.

Martin Scaiano and Diana Inkpen. 2012. Getting more from segmentation evaluation. In *Proceedings of the 2012 conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 362–366. Association for Computational Linguistics.

Alan P Schmidt and Trevor KM Stone. . Detection of topic change in irc chat logs. *http://www.trevorstone.org/school/ircsegmentation.pdf*.

Noam Slonim and Naftali Tishby. 1999. Agglomerative information bottleneck. In *Advances in Neural Information Processing Systems*, pages 617–623.

Noam Slonim and Naftali Tishby. 2000. Document clustering using word clusters via the information bottleneck method. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 208–215. ACM.

Bingjun Sun, Ding Zhou, Hongyuan Zha, and John Yen. 2006. Multi-task text segmentation and alignment based on weighted mutual information. In *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, CIKM '06, pages 846–847.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Naftali Tishby, Fernando C. N. Pereira, and William Bialek. 2000. The information bottleneck method. *arXiv preprint arXiv: 0004057*.

Masao Utiyama and Hitoshi Isahara. 2001. A statistical model for domain-independent text segmentation. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ACL '01, pages 499–506.

Deepu Vijayasenan, Fabio Valente, and Hervé Bourlard. 2009. An information theoretic approach to speaker diarization of meeting data. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(7):1382–1393.

Yi-Chia Wang, Mahesh Joshi, William W. Cohen, and Carolyn Penstein Ros. 2008. Recovering implicit thread structure in newsgroup style conversations. In *Proceedings of The International Conference on Weblogs and Social Media*, pages 152–160. The AAAI Press.

Mingliang Zhu, Weiming Hu, and Ou Wu. 2008. Topic detection and tracking for threaded discussion communities. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on*, volume 1, pages 77–83. IEEE.