Information Bottleneck Inspired Method For Chat Text Segmentation

(IJCNLP, 2017)

Authors – S Vishal, Mohit Yadav, Lovekesh Vig, Gautam Shroff

TCS Research, New Delhi, India

Overview

- Motivation
- Problem Formalization
- IB inspired Text Segmentation Algorithm
- Methodology
- Dataset Description
- Results
- Analysis

Motivation

- Conversation platforms have now become prevalent for both personal and professional usage. (like Fresco and Slack)
- Logs of such conversations offer potentially valuable information for various other applications such as automatic assessment of possible collaborative work among people.
- It is thus vital to invent effective segmentation methods that can seperate discussions into small granules of ‘independent’ conversational snippets.
- By ‘independent’, we mean 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.

Problem Description

- A chat sequence is represented as: $C=\{c_1, \ldots, c_i, \ldots, c_{|t|}\}$ where c_i is a text snippet i.e a post.
- A segment or a subsequence can be represented as: $C_{a:b}=\{c_a, \ldots, c_b\}$
- A segmentation of C is defined as a segment sequence $S=\{s_1, \ldots, s_p\}$ 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.

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 ?

1) ———

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

2) ———

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

3) ———

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:

The Information Bottleneck

- The information bottleneck method is a technique in information theory introduced by Naftali Tishby, Fernando C. Pereira, and William Bialek. [1]
- It is designed for finding the best tradeoff between accuracy and complexity (compression) when summarizing (e.g. clustering) a random variable X, given a joint probability distribution p(X,Y) between X and an observed relevant variable Y.
- The IB objective can be achieved by maximizing:

$$F = \underbrace{I(T,Y)}_{\text{Relevance}} - \underbrace{(1/\beta)}_{\text{Trade-off}} * \underbrace{I(X,T)}_{\text{Compression}}$$

where,

T : The compressed variable after compressing X.

Y : The relevant variable which encapsulate meaningful information about X.

I(T, Y): The mutual Information between T and Y. This information needs to be preserved.

I(X, T): The mutual information between X and T. This needs to be minimum.

[1] http://www.cs.huji.ac.il/labs/learning/Papers/allerton.pdf

IB Inspired Text Segmentation Algorithm

- 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)).
- IB objective is to maximize:

where, $$F = I(R, S) - (1/\beta) * I(S, C)$$

C : Chat sequence

S : Segmentation of C

R : Relevance variable - comprises of informative word clusters (will be explained shortly)

- Precisely, we merge the posts (s_i,s_{i+1}) in a thread such that minimum loss in F occurs.
- The 'loss' in F when merging the posts (s_i,s_{i+1}) is defined as:

$$d(s_i, s_{i+1}) = JSD[p(R|s_i), p(R|s_{i+1})] - 1/\beta * JSD[p(C|s_i), p(C|s_{i+1})]$$

JSD is the Jensen-Shanon divergence between two probability distributions.

- $JSD(P, Q) = 1/2 * KLD(P, M) + 1/2 * KLD(Q, M)$

$$M = (P + Q)/2$$

Methodology

STEP 1 :

- We model the set of relevance variables R as word clusters estimated by utilizing agglomerative IB based document clustering where posts are treated as relevance variables.[2]
- Consequently, R comprises of informative word clusters about posts.

For a given thread, we calculate the following:

1) $P(word_i | post_j) = \dfrac{count(word_i) \text{ in post } j}{\# \text{ words in post } j}$

2) $P(word_i, post_j) = P(word_i | post_j) \times P(post_j)$

3) $P(post_j | word_i) = \dfrac{P(word_i, document_j)}{P(word_i)}$

[2] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.9.6579&rep=rep1&type=pdf

	$P(post_1\|word)$	$P(post_2\|word)$	....	$P(post_m\|word)$
$word_1$				
$word_2$				
$word_3$				
. .				
.				
.				
$word_n$				

Here every row i is a distribution $P(post|word_i)$

Form a matrix which gives *δ* for every combination of probability distribution P(post|word)

	1	2	......	n
1	0	$\delta I_{1\|\|2}$		$\delta I_{1\|\|n}$
2	$\delta I_{1\|\|2}$	0		$\delta I_{2\|\|n}$
3	$\delta I_{1\|\|3}$		0	
. .				
.				
n-1	$\delta I_{1\|\|n-1}$	$\delta I_{2\|\|n-1}$		
n	$\delta I_{1\|\|n}$	$\delta I_{2\|\|n}$		0

δ matrix

STEP 2 :

	$P(post_1 \mid word)$	$P(post_2 \mid word)$	....	$P(post_m \mid word)$
$word_1$				
$word_2$				
$word_3$				
. .				
.				
.				
$word_n$				

Merge words with lowest corresponding value in $\boldsymbol{\delta}$ matrix.

After merging $\{word_i, word_j\}$ and forming a cluster,

We update the following:

$P(word_i) = P(word_i) + P(word_j)$

$P(word_i, post) = P(word_i, post) + P(word_j, post)$

$$P(post \mid word_i) = \frac{P(word_i, post)}{P(word_i)}$$

We also update the entries corresponding to i and j row/column in $\boldsymbol{\delta}$ matrix

STEP 3 :

After getting K word clusters, we update:

$P(word\_cluster_i, post_j)$ = Sum of $P(word, post_j)$ for all words in cluster i.
$P(word\_cluster_i)$ = Sum of P(word) for all words in cluster i.

$$P(word\_cluster_i | post_j) = \frac{P(word\_cluster_i, post_j)}{P(post_j)}$$

STEP 4 :

Similar to step 1, we merge posts with lowest *δ* between their corresponding conditional word cluster distribution. But this time instead of ***δ*** matrix we use ***δ*** vector which gives distance between adjacent distributions.

$\delta l_1 \| \delta l_2$	$\delta l_2 \| \delta l_3$	$\delta l_3 \| \delta l_4$	....	$\delta l_{n-1} \| \delta l_n$

	P(word $clus_1$|post)	P(word $clus_2$|post)	....	P(word $clus_k$|post)
$post_1$				
$post_2$				
$post_3$				
. .				
.				
.				
$post_n$				

Merge docs with the lowest D_{JS} between corresponding rows (check only between adjacent rows) and keep doing it till the stopping criteria is met to get C word segments.

Stopping Criteria

- The stopping criteria is $SC < \theta$, where:

1) $SC = I(R,S) / I(R,C)$
2) The value of θ is tuned by optimizing the performance over a validation dataset

- The value of SC is expected to decrease due to a relatively large dip in the value of $I(R,S)$ when more dissimilar pairs are merged.
- 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. [3]

[3] https://infoscience.epfl.ch/record/146338/files/Vijayasenan_TASLP_2009.pdf

Non Textual Clues[4]

- We incorporate 2 non-textual clues which seem to be important in the chat scenario:
  - Time between two consecutive posts
  - People mentions within the posts
- The formula for d is then modified to:

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

where,

$c^t_{a_{i+1}}$: time-stamp of the first post of segment s_{i+1}

$c^t_{b_i}$: time-stamp of the last post of segment s_i

s^p_i : normalised bag of posters counting all the people mentioned in the posts and the posters themselves in a segment

[4] Disentangling Chat (Elsner and Charniak 2010)

New Stopping Criteria

- We update the stopping criteria for the new formulation as follows:

$$SC = w_1 * I(R,S)/I(R,C) + w_2 * (1 - G(S)/G_{max}) + w_3 * H(S)/H_{max}$$

where,

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

$$H(S) = \sum_{i=1}^{|S|} \|s_i^p - s_{i+1}^p\|$$

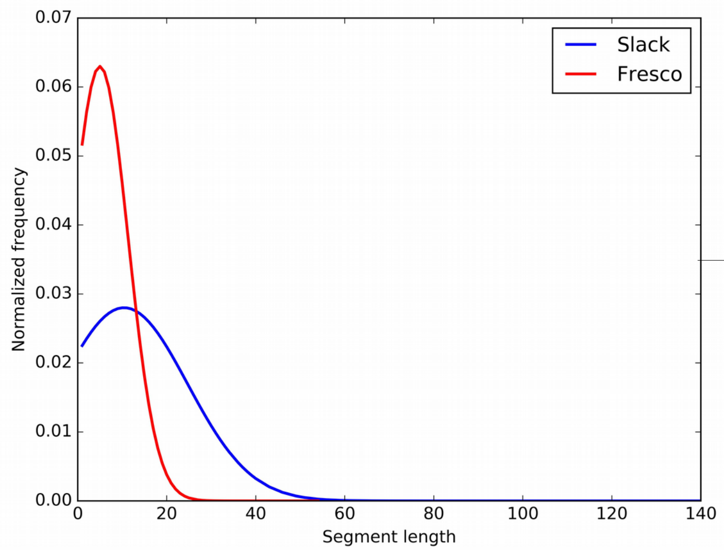
- In SC:
  - $I(R,S)/I(R,C)$ computes the fraction of information contained in S about R, normalized by the information contained in C about R
  - $G(S)/G_{max}$ computes total segment duration normalized by total duration of chat text sequence
  - $H(S)/H_{max}$ computes normalized sum of inter segment distances in terms of poster infromation
- Value of second and third term decreases with cardinality of S

Datasets

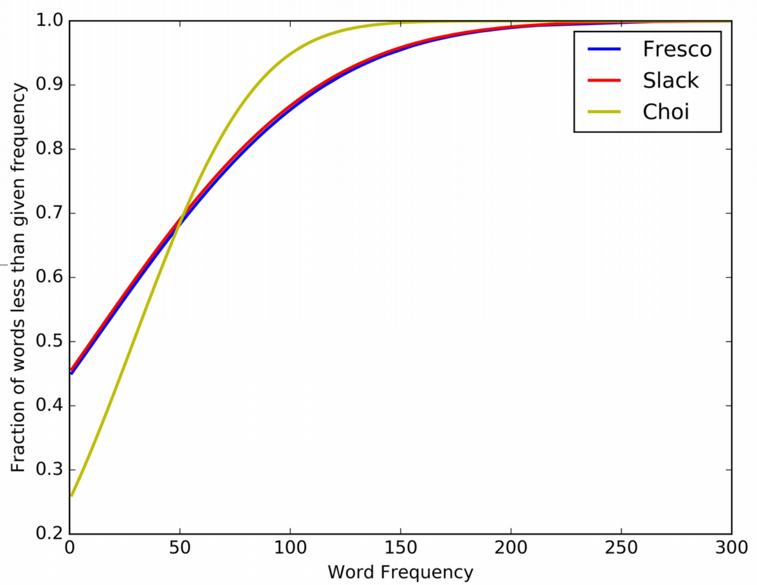
- We collected chat data from two forums for our experiments:
  - Slack (public data)
  - Fresco (a proprietary platform deployed inside our organization)
- We have manually annotated them for the text segmentation task for testing the performance of our algorithm.
- The collected raw data was in the 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 where N was selected randomly between 5 and 15. (Similar to Choi. 2000)
- 60% of these documents were used for tuning hyperparameters which include weights (w_1, w_2, w_3), θ, and β; and the remaining were used for testing.

Data Statistics

	Slack	Fresco
#Posts	9000	5000
#Segments	900	800
#Documents	73	73



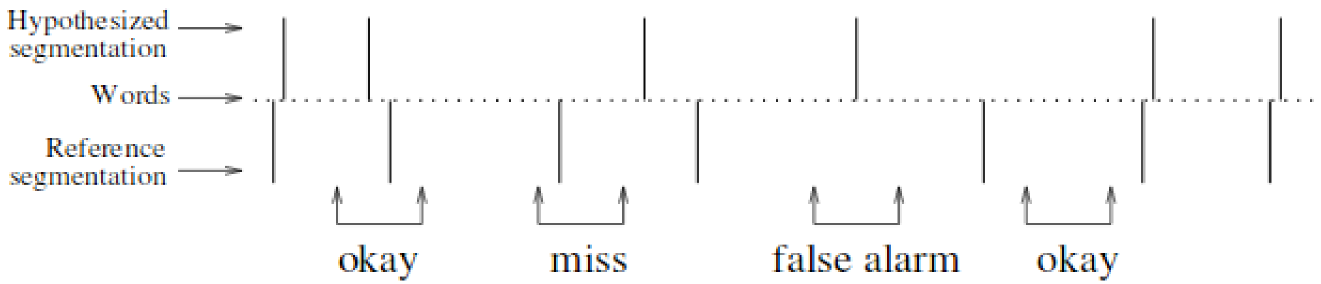
Normalized frequency distribution of the segment length of the two chat datasets.
Clearly, segments of Fresco are smaller.



Cumulative distribution of the vocabulary of the chat dataset compared with normal text dataset used by Choi.

Evaluation metric for segmentation

- We use the widely used metric (for evaluating segmentation) P_k(ref,hyp) which is defined as [5]:
  - ***The probability that a randomly chosen pair of sentences k sentences apart is inconsistently classified i.e for one segmentation, the pair lie in the same segment while for the other it lies in different segments.***



  - ***A sliding window of fixed size k (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 intrasegment errors (miss and false alarm) for all posts k apart is calculated by comparing inferred and annotated boundaries.***
  - ***Low value of P_k indicates good performance.***

[5] Statistical Models for Text Segmentation (Beeferman et al. 1999)

Results (in P_k)

Methods	Slack	Fresco
Random	60.6	54
No-Boundary	36.76	45
Average Time	32	35
C-99[5]	35.18	37.75
Dynamic Prog.[6]	28.7	35
Encoder-Decoder Dist.	29	38
LDA-Distance[7]	36	44
IB-Variants:		
Text (only IB)	33	42
TimeDiff	**26.75**	**34.25**
Poster	34.52	41.50
Text+TimeDiff	**26.47**	**34.68**
Text+Poster	28.57	38.21
Text+TimeDiff+Poster	**25.47**	**34.80**

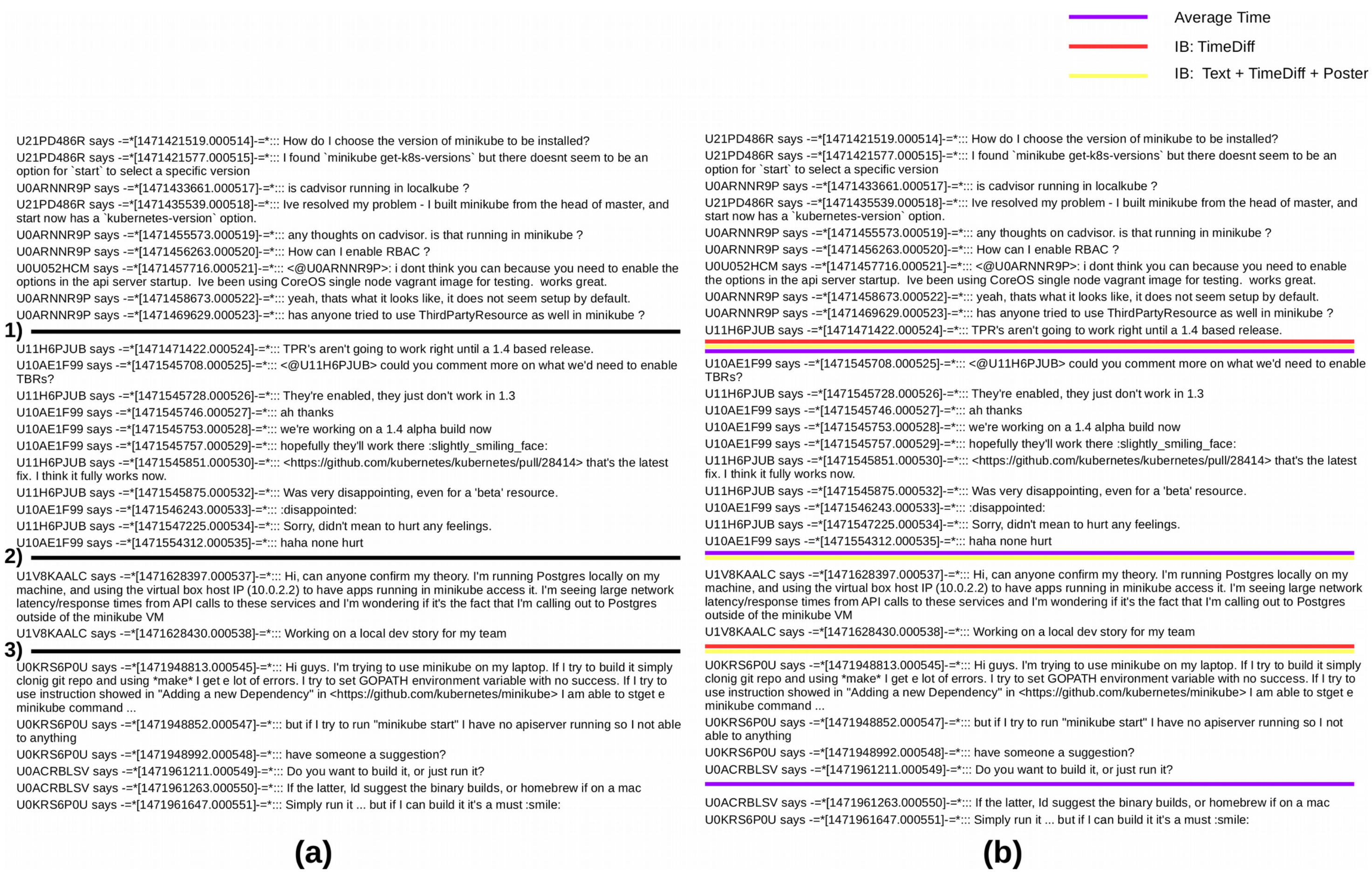
- Random → Inserting boundaries at random.
- No-Boundary → Inserting no boundaries.
- Average Time → Boundaries are inserted after a fixed time. This time is calculated from a separate portion of the annotated dataset.
- C-99 → The classic text segmentation algorithm.
- Dynamic Prog. → The dynamic programming algorithm proposed in [6] is used.
- Encoder-Decoder Dist. → A seq2seq RNN is trained to get a dense repesentation of posts. Agglomerative merging is done then based on Euc. distance
- LDA-Distance → Representations come from a topic model having 100 topics.

[5] Advances in domain indepen-dent linear text segmentation (Choi 2000)
[6] Linear Text Segmentation using a Dynamic Programming Algorithm (Kehagias et al. 2003)
[7] Latent dirichlet allocation (Blei et al. 2003)

Quantitative Analysis

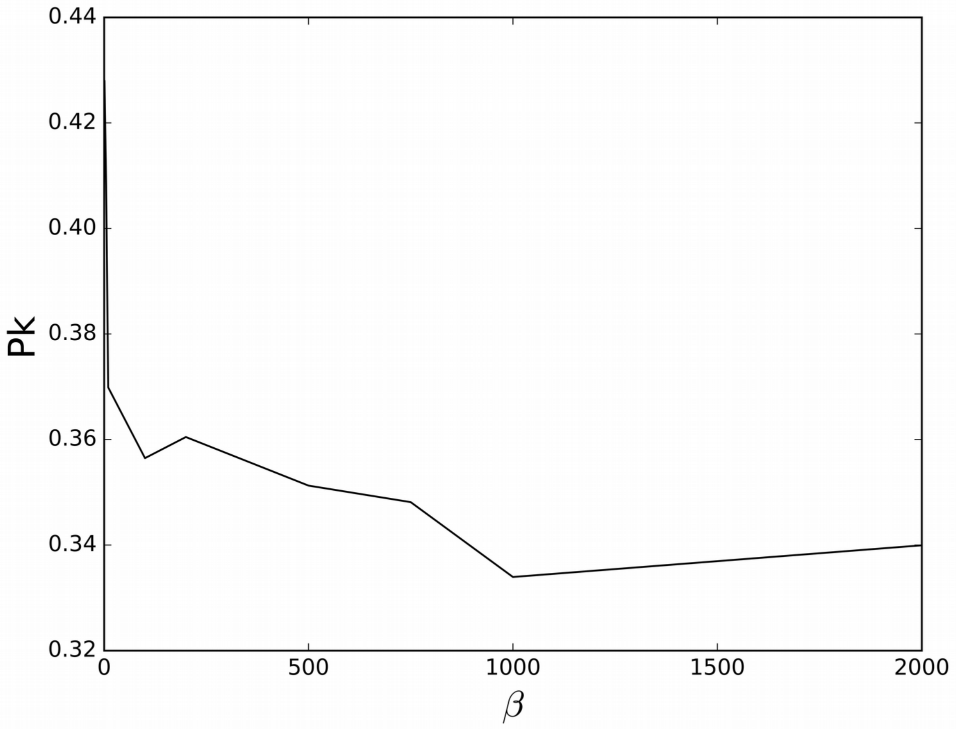
- For Slack dataset, 4 different variants of the proposed IB based method achieve higher performance than others.
- In case of Fresco dataset, different variants of the proposed method achieve superior performance but not as significantly in terms of absolute P_k value, as they do for the Slack dataset.
  - 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.
  - This can be seen from the normalized frequency distribution of both datasets.

Qualitative Analysis



Effect of Parameters

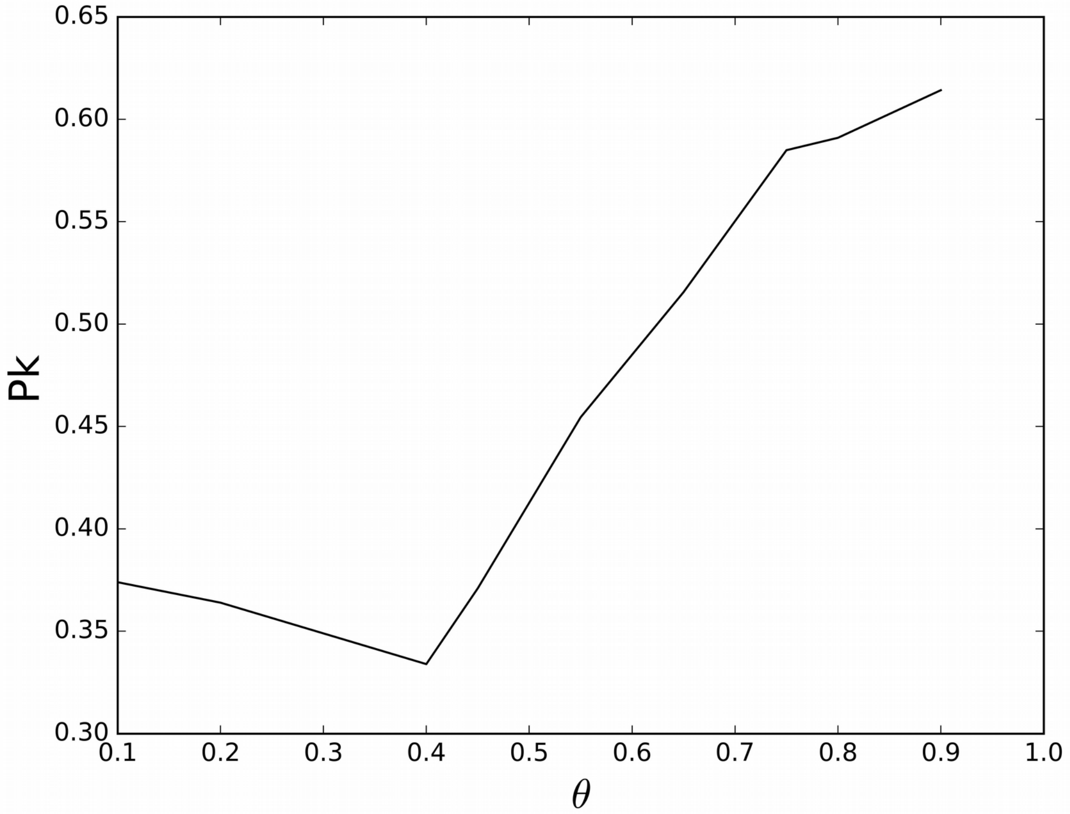
P_k *v/s β*



- β represents a trade-off between the compression and preserved information.
- As β does not lie on any extremity, it indicates its importance in the IB objective.

Effect of Parameters

P_k v/s θ



- Initially, the average value of P_k decreases as more coherent posts are merged.
- After reaching a minimum, it starts increasing potentially due to merging of more dissimilar posts.

Future Work

- It would be interesting to explore some deep learning methods applied to the task of chat text segmentation.
- In future, it will be interesting to investigate the possibility of incorporating semantic word embeddings in the proposed IB method.
- Major research needs to be conducted to tackle low frequency/out of vocabulary words in the context of online chats.

Thank You!