

マイクロブログにおけるトピック出現量推移の高速な抽出

福山 怜史^{1,a)} 若林 啓^{2,b)}

受付日 2019年3月8日, 採録日 2019年6月12日

概要: 近年, 多くのメディアでは, 関係するツイートの出現量が時間経過によって急上昇する話題を対象に情報発信が行われており, Twitter の話題の分析において話題の出現量の推移が注目されている. Twitter ではハッシュタグが一部のツイートにしか与えられていないため, すべてのツイートに含まれる話題の推移を網羅的に観測することは容易ではない. この問題に対して, Bitern topic model (BTM) によってトピックを推定し, 推定したトピックの出現量を利用する方法が有効である. しかし, Twitter ではリアルタイムに膨大なツイートが更新されるため, トピックの推定やトピック出現量の計算において時間的な効率性が求められる. 本研究では, ツイートデータを対象に, 高速にトピックを学習し, 各トピックの単位時間あたりの出現量の計算を効率的に行う手法を提案する. 提案手法では, BTM に対してミニバッチ学習を適用し, トピック学習の高速化を図る. またトピック出現量の計算では, 一部のデータを用いた近似的な計算を行うことによって, 実質的な高速化を図る. 実験では, 提案手法が既存手法より汎化性能が優れつつ学習における処理時間が短縮できることを確認した. またトピック出現量を近似する方法について複数の方法を示し, 近似による誤差の大きさと処理時間の短縮の観点から比較と検討を行った.

キーワード: マイクロブログ, トピック出現量, Bitern topic model, トピックモデル, ミニバッチ学習

Fast Extraction of Time Series Variation for Topic Popularity in Microblogs

SATOSHI FUKUYAMA^{1,a)} KEI WAKABAYASHI^{2,b)}

Received: March 8, 2019, Accepted: June 12, 2019

Abstract: Recently, Twitter has attracted as a media that reflects popular topics in real time. Especially, many media provide the information of the topics that the number of tweets belonging to itself suddenly increases. However, because most tweets are not classified by tagging, it is hard to observe the time series variation of the topic from all tweets. In order to solve this problem, a method using the topic model, which is a method for estimating topics by documents, is proposed. However, since tweets are posted enormous tweets in real time, we need efficient methods for estimating topics and calculating the topic popularity. We propose the efficient method to estimate topics and calculate the time series variation for the topic popularity for tweets. In order to speed up the estimation of topics, we improve Bitern topic model, which is an effective method for short texts, to minibatch training. In addition, we propose a method to efficiently calculate the approximate topic popularity from partial data. Our experiments suggest that the proposed method has higher generalization ability and faster training time than baseline. Also, we discuss efficient and less lossy methods that calculating the topic popularity from several methods.

Keywords: microblog, topic popularity, bitern topic model, topic model, minibatch training

¹ 筑波大学大学院図書館情報メディア研究科
Graduate School of Library, Information and Media Studies,
University of Tsukuba, Tsukuba, Ibaraki 305-8550, Japan

² 筑波大学図書館情報メディア系
Faculty of Library, Information and Media Science, Univer-
sity of Tsukuba, Tsukuba, Ibaraki 305-8550, Japan

a) s1721691@s.tsukuba.ac.jp

b) kwakaba@slis.tsukuba.ac.jp

1. はじめに

近年, 多くのメディアでは, ツイートの出現量が時間経過によって急上昇するような話題を対象に情報発信が行われている. このような背景から, Twitter に代表されるマイクロブログにおける話題の分析において, 話題の出現

量の時間的な推移の利用が注目されており、投稿されたツイートデータから話題の出現量の時間的な推移を推定する手法が求められている。この手法の1つに、話題に関係するハッシュタグが与えられたツイートを収集し、単位時間あたりのツイート数を計算し、話題の出現量とする方法がある。しかしこの手法は、Twitterではハッシュタグが一部のツイートにしか与えられていないことから、話題に関係するツイートを収集するという観点において網羅的な手法といえない。以上の観点から、ツイートに含まれるトピックを分析し抽出する手法が有効であると考えられる。この手法として、Biterm topic model (BTM) [1]が提案されている。BTMとは、bitermと呼ばれる同じ文書に出現する単語対の集合を分析することで共起しやすい単語の集合としてトピックを抽出する手法である。したがって、BTMによってトピックの抽出を行い、抽出したトピックごとに単位時間あたりの推定ツイート数を計算することによって、Twitterで観測された話題における、すべてのツイートを網羅した話題の出現量を得ることができる。

一方で、ハッシュタグを利用する手法に対して、この手法ではトピックの学習および単位時間あたりのトピックの出現量を計算する処理を必要とする。この処理では数日分のツイートから抽出した膨大なbitermを使用するため、時間的なコストが高い。以上の問題を軽減するために、本研究では、BTMの学習とトピック出現量の計算を高速化することで、Twitterにおける話題の時系列変化を効率的に抽出する手法を提案する。提案手法は、トピックの学習とトピック出現量の計算においてそれぞれ以下のアプローチによって高速化を図る。

- **トピック学習の高速化。** BTMではトピック学習の効率的な推論アルゴリズムとして、確率的周辺変分推論 (Stochastic collapsed variational Bayesian inference; SCVB0) [2]が提案されている。SCVB0では反復処理の各イテレーションにおいてbitermを1つずつ読み込みトピックを学習する。本研究では、SCVB0にミニバッチ学習を導入することで、ミニバッチ内に含まれる同一のbitermの重複を利用してトピック学習の高速化が可能になることを示す。さらに、ミニバッチサイズを大きくするほどbitermの重複数が大きくなるが、パラメータの更新周期が長くなるために学習の効率が低下するという課題があることを明らかにした上で、ミニバッチを複数のセグメントに分割し、セグメントごとにパラメータを更新するミニバッチセグメント学習を提案する。この拡張による高速化の原理について、4.1節、4.2節で述べる。
- **トピック出現量計算の高速化。** BTMによって学習したトピックからトピック出現量を計算するためには、各時間で投稿されたツイートに含まれるbitermのトピックを推論する計算処理が必要となる。この処理で

は、各時間出現したすべての異なりbitermに対してトピックの推論が行われるため、膨大な処理時間がかかる。提案手法では、各時間出現したbitermのうち、一部のbitermのみ用いてトピック出現量を近似的に計算する。これにより、トピック出現量には誤差が含まれてしまうが、単位時間出現するbitermをすべて用いる場合と比較してより高速に計算することができる。

実験では、実際に投稿されたツイートをを用いてトピックの学習およびトピック出現量の計算を行い、それぞれの手法で高速化の検証を行う。トピック学習の高速化では、既存手法と比較して、汎化性能が劣らず処理時間が短縮されているか確認する。トピック出現量の高速化では、複数の手法の中から元のトピック出現量に対して、最も誤差が少なくかつ処理時間が短縮されている手法を検討する。

本研究で提案する高速化手法は、大規模な短文書集合を用いたトピック学習やトピック出現量推定を行う状況であれば、トピック出現量の時間的な推移を抽出する目的以外にも適用できる。しかし、トピック出現量推移の抽出は、提案手法による高速化が求められる主要な応用の1つであるため、本稿ではこの目的を達成する文脈で議論を行う。

2. 関連研究

文書に含まれるトピックを分析する手法として、Latent Dirichlet Allocation (LDA) [3]が提案されている。LDAは、文書に含まれる単語がトピックによって生成される統計的なモデル (トピックモデル) をベイズ推定によって求める。しかし、ツイートのような文書に含まれる単語が少数なデータセットでは、LDAで抽出したトピックのまとまりが悪くなるのが指摘されている [4]。本研究ではこの問題を“短文書問題”と呼ぶ。

短文書問題を軽減したトピックを抽出する手法として、Biterm Topic Model (BTM) [1]がある。BTMは、トピックモデルの一種であり、文書をbitermと呼ばれる非順序の単語対の集合に変換し、各bitermのトピックを推論することによってトピックを学習する手法である。Chengら [1]は、LDAにおける短文書問題の原因は、文書ごとに定義されるトピック分布の潜在変数の推定が、文書に含まれる単語数が少ないときに不安定になることにあると見なし、文書集合全体でトピック分布が共有されるようにBTMのモデルを構築した。Chengらは、実世界の短文書データセットで実験を行い、LDAと比較して、BTMによって推定されるトピックの語の一貫性が高いことを確認している。

また、文書データに含まれる話題の抽出においてトピックの時間変化を考慮した手法として、Dynamic Topic Model (DTM) [5]やMultiscale Dynamic Topic Model (MDTM) [6]がある。これらの手法は、Latent Dirichlet Allocation (LDA) [3]にトピックの時間発展性を導入した

モデルである。このモデル化によって、DTM や MDTM では、時間経過によってトピックの単語分布が変化する。Koike ら [7] は、ニュース記事およびツイートから DTM によってトピックを抽出し、単位時間あたりの各トピックの出現量を計算する手法を提案している。しかし、DTM によって推定したトピックは時間ごとにトピックの単語分布が異なるため、各時間のトピック出現量が実際には一貫していない。また DTM では効率的な推論アルゴリズムが提案されていないことから、トピック数を大きく設定した場合、膨大な学習時間を必要とする。加えて、短文書問題によって、ツイートデータから直接トピックを推定することは容易ではない。

本研究では、トピック出現量の時間的な一貫性を重視し、トピック出現量を計算する期間において、トピックの時間発展性を考慮しない手法を採用する。また、ツイートデータから直接トピックを推定するため短文書問題の軽減を考慮する。以上の観点から、本研究では BTM によってトピックの推定を行う。

3. 前準備

3.1 Biterm topic model (BTM)

BTM では、文書の集合を biterm の集まりに変換して扱う。本研究では、ツイートが文書に対応する。\$D\$ をツイート数、ツイートの集まりを \$\Omega = \sigma^{(1)}, \dots, \sigma^{(D)}\$, ツイート \$\sigma^{(d)}\$ の単語列を \$w_1^{(d)}, \dots, w_{|\sigma^{(d)}|}^{(d)}\$ と表記する。それぞれの単語は語彙集合 \$\mathcal{V}\$ の要素であり、語彙数を \$W = |\mathcal{V}|\$ とする。また、それぞれのツイートにはタイムスタンプがついている。本稿では、特定の単位時間*1内のツイートには同一のタイムスタンプが付与されていると見なし、基準時刻を 1 とした \$\sigma\$ のタイムスタンプを \$\delta(\sigma) \in [T]\$ と表す。ただし、\$[T]\$ は 1 から \$T\$ までの整数の閉区間である。

biterm は、同一のツイートに含まれる 2 つの異なる単語からなる非順序対である。\$\Omega\$ に含まれる異なり biterm の集合 \$\mathcal{V}_B \subseteq \mathcal{V}^2\$ は以下のように定義される。

$$\mathcal{V}_B = \{\{w_i^{(d)}, w_j^{(d)}\} | d \in [D], i, j \in [|\sigma^{(d)}|], w_i^{(d)} \neq w_j^{(d)}\}.$$

また、重複を含めた biterm の集まりを \$\mathbf{B}\$ と表記する。1 つのツイート \$\sigma^{(d)}\$ において同一の biterm \$b = \{w_1, w_2\}\$ が複数回出現する際には、\$w_1\$ と \$w_2\$ の出現位置の添字を入れ替えた場合を除き、重複して出現したものと見なす。すなわち、ツイート \$\sigma^{(d)}\$ における biterm \$b \in \mathcal{V}_B\$ の頻度は以下のように表される。

$$n_{\sigma^{(d)}}(b) = |\{\{i, j\} | \{w_i^{(d)}, w_j^{(d)}\} = b\}|.$$

\$\mathbf{B}\$ における biterm \$b \in \mathcal{V}_B\$ の頻度は、\$\sum_{d=1}^D n_{\sigma^{(d)}}(b)\$ と表せる。\$\mathbf{B}\$ の要素数を \$N_{\mathbf{B}}\$ と表記し、その要素に便宜的に添

*1 たとえば、本稿の実験では、単位時間を 1 時間としている。

字をつけて \$\mathbf{B} = \{b_i\}_{i=1}^{N_{\mathbf{B}}}\$, \$b_i = \{w_{i,1}, w_{i,2}\}\$ と表記する。

BTM では、それぞれの biterm \$b_i\$ について、トピックを表す離散潜在変数 \$z_i\$ を考える。トピック数を \$K\$ として、\$z_i \in [K]\$ である。また、\$K\$ 次元のトピック分布を示すベクトルを \$\theta = \{\theta_k\}_{k=1}^K\$ とし、サイズ \$K \times W\$ の単語分布の行列を \$\Phi = \{\phi_k\}_{k=1}^K\$ とする。\$\Phi\$ の行列の各行ベクトル \$\phi_k\$ はトピックごとの単語分布を表し、その次元は \$W\$ である。\$\theta\$ および \$\phi_k\$ の \$L_1\$ ノルムの大きさは 1 とする。BTM は、トピック分布 \$\theta\$ およびトピック \$k\$ の単語分布 \$\phi_k\$ においてそれぞれ \$\alpha \cdot \beta\$ をハイパーパラメータとしたディリクレ分布を事前分布に持ち、以下の過程に従って biterm が生成されることを仮定する。

(1) Draw \$\theta \sim \text{Dirichlet}(\alpha)\$.

(2) For each topic \$k \in [K]\$,

(a) Draw \$\phi_k \sim \text{Dirichlet}(\beta)\$.

(3) For each biterm \$b_i \in \mathbf{B}\$,

(a) Draw \$z_i \sim \text{Multinomial}(\theta)\$.

(b) Draw \$w_{i,1}, w_{i,2} \sim \text{Multinomial}(\phi_{z_i})\$.

BTM を考案した Cheng らはこのモデルを周辺化ギブスサンプリング (Collapsed Gibbs Sampling; CGS) によって学習する手法を提案している [1]。しかし、CGS は \$N_{\mathbf{B}}\$ 個の潜在変数 \$z_i\$ の値をメモリに展開しておく必要があり、大規模なデータの学習に適用することが難しいという欠点があることに加え、次節で述べる SCVB0 に比べて学習の進行が遅いことが知られている。

3.2 BTM における SCVB0

BTM を効率的に学習するアルゴリズムはいくつか提案されている [8] が、現在知られている中で最も計算量の小さい手法の 1 つに SCVB0 [2] がある。この手法は、LDA における周辺化変分推論 (Collapsed variational Bayesian inference; CVB) [9] に対してトピックの期待値の式のテイラー展開を 0 次近似したアルゴリズム (CVB0) [10] を、BTM の学習アルゴリズムに適用し、確率的最適化のアルゴリズムに拡張したものである。Awaya らは、BTM における SCVB0 の学習アルゴリズムが BTM における CGS [1] よりトピックの学習が高速であることを確認している [2]。BTM における SCVB0 では、\$\mathbf{B}\$ から 1 個ずつ biterm \$b_i\$ を選択し、そのトピック \$z_i\$ が \$k\$ である確率を示す変分パラメータ \$\hat{z}_{i,k}\$ を式 (1) で推定する。

$$\hat{z}_{i,k} \propto (N_k + \alpha) \frac{(N_{w_{i,1}|k} + \beta)(N_{w_{i,2}|k} + \beta)}{(2N_k + W\beta)(2N_k + W\beta + 1)}. \quad (1)$$

コーパスにおけるトピック \$k\$ の出現頻度の期待値 \$N_k\$ とトピック \$k\$ で単語 \$w\$ が出現する頻度の期待値 \$N_{w|k}\$ の推定値として \$\hat{N}_k\$ および \$\hat{N}_{w|k}\$ をそれぞれ式 (2) および式 (3) によって計算する。

$$\hat{N}_k = |\mathbf{B}| \hat{z}_{i,k}, \quad (2)$$

$$\hat{N}_{w|k} = \begin{cases} |\mathbf{B}| \hat{z}_{i,k} & \text{if } w \in b_i, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

Robbins-Monro 法による確率的最適化にのっとり、この期待値の近似値から N_k と $N_{w|k}$ を式 (4) および式 (5) によって計算する。

$$N_k \leftarrow (1 - \nu^{(s)})N_k + \nu^{(s)}\hat{N}_k, \quad (4)$$

$$N_{w|k} \leftarrow (1 - \nu^{(s)})N_{w|k} + \nu^{(s)}\hat{N}_{w|k}. \quad (5)$$

$\nu^{(s)}$ は式 (6) によって計算される。

$$\nu^{(s)} = \frac{1}{(\tau + s)^\kappa}. \quad (6)$$

ただし、 $\tau > 0$, $0.5 < \kappa \leq 1$ は定数である。

また Awaya らは、 $N_{w|k}$ の計算を効率的に行う手法を提案している。この手法では、 s 回目の行列の更新における $N_{w|k}$ をスカラーと行列の積の形である $\rho_s \tilde{N}_{w|k}$ とする。 ρ_s は $\prod_{i=1}^s (1 - \nu^{(i)})$ によって計算される。以上の計算によって、biterm b に含まれる単語 w_1, w_2 の更新は $\tilde{N}_{w|k}$ の要素 (k, w_1) および (k, w_2) に対する $\frac{\nu^{(s)}|\mathbf{B}|}{\rho_s}$ の加算のみとなり、 $N_{w|k}$ の更新にかかる計算量は、 K および W に対しては $\mathcal{O}(1)$ になる。

以上の更新を十分な回数繰り返して得られた統計量 N_k と $N_{w|k}$ を用いて、トピック割合 θ とトピックの単語分布 Φ は、それぞれ式 (7) および式 (8) によって計算される。 N は $\sum_{k'=1}^K N_{k'}$, $N_{\cdot|k}$ は $\sum_{w'=1}^W N_{w'|k}$ を表す。

$$\theta_k = \frac{N_k + \alpha}{N + \sum_{k'=1}^K \alpha_{k'}}, \quad (7)$$

$$\phi_{k,w} = \frac{N_{w|k} + \beta}{N_{\cdot|k} + W\beta}. \quad (8)$$

3.3 トピック分布のハイパーパラメータの最適化

Wallach ら [11] は、LDA におけるトピック分布 θ のハイパーパラメータ α は各要素が異なる値を持つ非対称 Dirichlet 分布、トピック k の単語分布 ϕ_k のハイパーパラメータ β は各要素が同一の値を持つ対称 Dirichlet 分布が有用であることを確認している。

この性質を BTM でも考慮し、本研究ではトピック分布 θ のハイパーパラメータ α の最適化を行う。本研究では、BTM の学習と同時にハイパーパラメータ α の最適化を行い、後述する提案手法では、不動点反復法による式 (9) の反復計算によって更新を行う。

$$\alpha_k \leftarrow \alpha_k \frac{\Psi(N_k + \alpha_k) - \Psi(\alpha_k)}{\Psi(N + \sum_{k'=1}^K \alpha_{k'}) - \Psi(\sum_{k'=1}^K \alpha_{k'})}. \quad (9)$$

3.4 トピック出現量の計算

本研究では、それぞれの単位時間内に含まれるトピック

出現量を推定することで、それぞれのトピックの時間的な推移を抽出する。ツイート σ のトピック割合は、式 (10) によって計算される [1]。

$$P(z|\sigma) = \sum_{b \in \mathcal{V}_B} P(z|b)P(b|\sigma). \quad (10)$$

biterm $b_i = \{w_{i,1}, w_{i,2}\}$ が与えられたとき、 b_i がトピック k である確率 $P(z = k|b_i)$ は式 (11) によって計算される。

$$P(z = k|b_i) = \frac{\theta_k \phi_{k,w_{i,1}} \phi_{k,w_{i,2}}}{\sum_{k'} \theta_{k'} \phi_{k',w_{i,1}} \phi_{k',w_{i,2}}}. \quad (11)$$

θ_k および $\phi_{k,w}$ は、それぞれ式 (7) および式 (8) によって学習結果として求めた値を用いる。また、ツイート中の biterm b の割合 $P(b|\sigma)$ は式 (12) によって定義される。

$$P(b|\sigma) = \frac{n_\sigma(b)}{\sum_{b' \in \mathcal{V}_B} n_\sigma(b')}. \quad (12)$$

ある単位時間 $t \in [T]$ 内のトピック k の出現量は、当該の単位時間に含まれるツイート集合 $\Omega_t = \{\sigma | \delta(\sigma) = t\}$ におけるトピック割合の総和 $N_{k|t} = \sum_{\sigma \in \Omega_t} P(z|\sigma)$ と定義する。また、単位時間 t において頻度が 0 でない biterm 集合を $\mathcal{V}_{B,t} = \{b | \exists \sigma \in \Omega_t [n_\sigma(b) > 0]\}$ と定義する。式 (10) より、 $N_{k|t}$ は式 (13) に書き換えられる。

$$N_{k|t} = \sum_{\sigma \in \Omega_t} \sum_{b \in \mathcal{V}_{B,t}} P(b|\sigma)P(z = k|b). \quad (13)$$

$P(z|b)$ の値は σ に依存しないため、 $\mathcal{V}_{B,t}$ の要素 b についての和に書き換えることができる。したがって、 $N_{k|t}$ は以下のように計算することができる。

$$N_{k|t} = \sum_{b \in \mathcal{V}_{B,t}} P(z = k|b) \sum_{\sigma \in \Omega_t} P(b|\sigma). \quad (14)$$

4. 提案手法

提案手法では、トピックの学習の高速化のために、SCVB0 をミニバッチ学習に拡張する。また、抽出したトピックの出現量を推定するとき、単位時間に出現した biterm から一部に対して推論を適用することで高速化を図る。

4.1 BTM における SCVB0 によるミニバッチ学習

訓練するデータの中から複数のデータをサンプリングし、それを全体のデータの近似として学習する手法をミニバッチ学習と呼ぶ。SCVB0 によるミニバッチ学習では、BTM を SCVB0 のアルゴリズムで学習する際に \mathbf{B} からあらかじめ決定したミニバッチサイズの数だけ biterm をサンプリングし、ミニバッチごとに N_k と $N_{w|k}$ の更新を行う。LDA の学習においてミニバッチ学習を適用する手法は提案されており [12], [13]、本手法のミニバッチ学習はその素直な適用であるが、LDA ではミニバッチの要素として文書をサンプリングするのに対して、BTM では biterm

Algorithm 1 BTM における SCVB0 のミニバッチ学習

```

Randomly initialize  $N_k$  and  $N_{w|k}$ 
for each iteration do
  Sample minibatch of biterns  $\mathbf{B}^{(s)}$ 
  Compute  $N_b^{(s)}$  in  $\mathbf{B}^{(s)}$  for all  $b \in \mathcal{V}_{\mathbf{B}^{(s)}}$ 
  for  $b \in \mathcal{V}_{\mathbf{B}^{(s)}}$  do
    for each topic  $k \in [K]$  do
      Compute  $\hat{z}_{b,k}$  using Eq. (1)
    end for
  end for
  Update  $N_k$  using Eq. (15) and  $N_{w|k}$  using Eq. (16)
end for
Compute global parameters  $\theta$  using Eq. (7) and  $\Phi$  using Eq. (8)

```

をサンプリングする。コーパス中には同一の bitern が複数存在するため、BTM のミニバッチ学習では、LDA のミニバッチ学習とは異なり、ミニバッチ内に含まれる bitern の重複を利用して計算量を削減することができる。本節では、この性質に着目して計算量を削減したミニバッチ学習について述べる。さらに次節において、この性質に基づいて学習の効率化を目指したミニバッチ-セグメント学習手法について述べる。

SCVB0 による BTM のミニバッチ学習のアルゴリズムを Algorithm 1 に示す。 $\mathbf{B}^{(s)}$ は s 回目のイテレーションにおけるミニバッチに含まれる（重複を含む）bitern の集まり、 $\mathcal{V}_{\mathbf{B}^{(s)}}$ は $\mathbf{B}^{(s)}$ に含まれる異なり bitern の集合、 $N_b^{(s)}$ は $\mathbf{B}^{(s)}$ に含まれる bitern b の頻度を表す。ミニバッチ学習による、SCVB0 の更新式を式 (15) および式 (16) とする。

$$N_k \leftarrow (1 - \nu^{(s)})N_k + \nu^{(s)} \frac{|\mathbf{B}|}{|\mathbf{B}^{(s)}|} \sum_{b \in \mathcal{V}_{\mathbf{B}^{(s)}}} \hat{z}_{b,k} N_b^{(s)}, \quad (15)$$

$$N_{w|k} \leftarrow (1 - \nu^{(s)})N_{w|k} + \nu^{(s)} \frac{|\mathbf{B}|}{|\mathbf{B}^{(s)}|} \sum_{b \in \mathcal{V}_{\mathbf{B}^{(s)}}} \hat{z}_{b,k} N_b^{(s)} \delta(w \in b). \quad (16)$$

ただし、 $\hat{z}_{b,k}$ は、添字の bitern b を当該の bitern が元々出現する位置 i に読み替えて式 (1) で計算する。このアルゴリズムでは、ミニバッチとして複数の bitern をサンプリングした際に、ミニバッチ内に出現する各 bitern の頻度を数えあげ、bitern $b \in \mathcal{V}_{\mathbf{B}^{(s)}}$ の変分パラメータ $\hat{z}_{b,k}$ とその bitern の頻度 $N_b^{(s)}$ の積の総和を N_k と $N_{w|k}$ の更新量として計算する。これにより、トピック学習のボトルネックとなるミニバッチ内の各 bitern の変分パラメータの計算回数は、ミニバッチ内に含まれる異なり bitern 数まで削減され、1 イテレーションあたりの計算量は $O(|\mathcal{V}_{\mathbf{B}^{(s)}}|K)$ となる。

4.2 ミニバッチをセグメントに分割する学習（ミニバッチ-セグメント学習）

本研究では、ミニバッチ内で重複する bitern が存在する性質を利用し、さらに効率的な学習手法を提案する。4.1 節

Algorithm 2 BTM における SCVB0 のミニバッチ-セグメント学習

```

Randomly initialize  $N_k$  and  $N_{w|k}$ 
for each iteration do
  Sample minibatch of biterns  $\mathbf{B}^{(s)}$ 
  Compute  $N_b^{(s)}$  in  $\mathbf{B}^{(s)}$  for all  $b \in \mathcal{V}_{\mathbf{B}^{(s)}}$ 
  for each segment  $j \in [\lceil \frac{|\mathcal{V}_{\mathbf{B}^{(s)}}|}{S} \rceil]$  do
    for  $b \in \mathcal{V}_{\mathbf{B}^{(s)}}^{(j)}$  do
      for each topic  $k \in [K]$  do
        Compute  $\hat{z}_{b,k}$  using Eq. (1)
      end for
    end for
  end for
  Update  $N_k$  using Eq. (15) and  $N_{w|k}$  using Eq. (16)
end for
Compute global parameters  $\theta$  using Eq. (7) and  $\Phi$  using Eq. (8)

```

で説明しているミニバッチ学習では、ミニバッチサイズが大きければ大きいほど重複する異なり bitern の種類の数が多くなるため、変分パラメータの計算回数の削減量が大きくなることが期待される。一方で、ミニバッチサイズを大きくするほどミニバッチに含まれる異なり bitern 数が増加するため、変分パラメータの更新の時間的な周期が長くなり、結果的に実時間に対する学習の進行速度が低下する可能性がある。重複数を大きく保ちながら更新の時間的な周期を短くするには、重複数の大きい bitern のみを選んで小さいミニバッチを作るという方法も考えられるが、この方法では高頻度の bitern のみを著しく偏って学習することになる。

ミニバッチ-セグメント学習では、ミニバッチサイズを大きく保ちながら更新の時間的な周期を短くするために、ミニバッチ内の bitern の頻度を数えあげた後、あらかじめ決めた種類数ずつ異なり bitern 集合 $\mathcal{V}_{\mathbf{B}^{(s)}}$ を分割し、この分割（セグメント）ごとに N_k と $N_{w|k}$ の更新を行う。本研究では、各分割に含まれる異なり bitern の数をセグメントサイズと呼び、提案手法 2 を“ミニバッチ-セグメント学習”と呼ぶ。図 1 に示すように、ミニバッチ学習では、サンプリングした bitern すべてを用いて N_k と $N_{w|k}$ を更新することに対し、ミニバッチ-セグメント学習ではサンプリングした異なり bitern を複数のセグメントに分割し、このセグメントごとに N_k と $N_{w|k}$ の更新を行う。これにより、ミニバッチ学習の場合と比較して、サンプリングした bitern の数が同一でも、より短い周期で N_k と $N_{w|k}$ の更新を行うことができるため、トピック学習の進行がより高速になると考えられる。

SCVB0 によるミニバッチ-セグメント学習のアルゴリズムを Algorithm 2 に示す。本稿では、天井関数を $\lceil x \rceil$ と表記する。セグメントサイズ S を、1 セグメントに含まれる異なり bitern の数とする。また、 j 番目のセグメントに割り当てられた異なり bitern 集合を $\mathcal{V}_{\mathbf{B}^{(s)}}^{(j)}$ と表す。 j 番目の

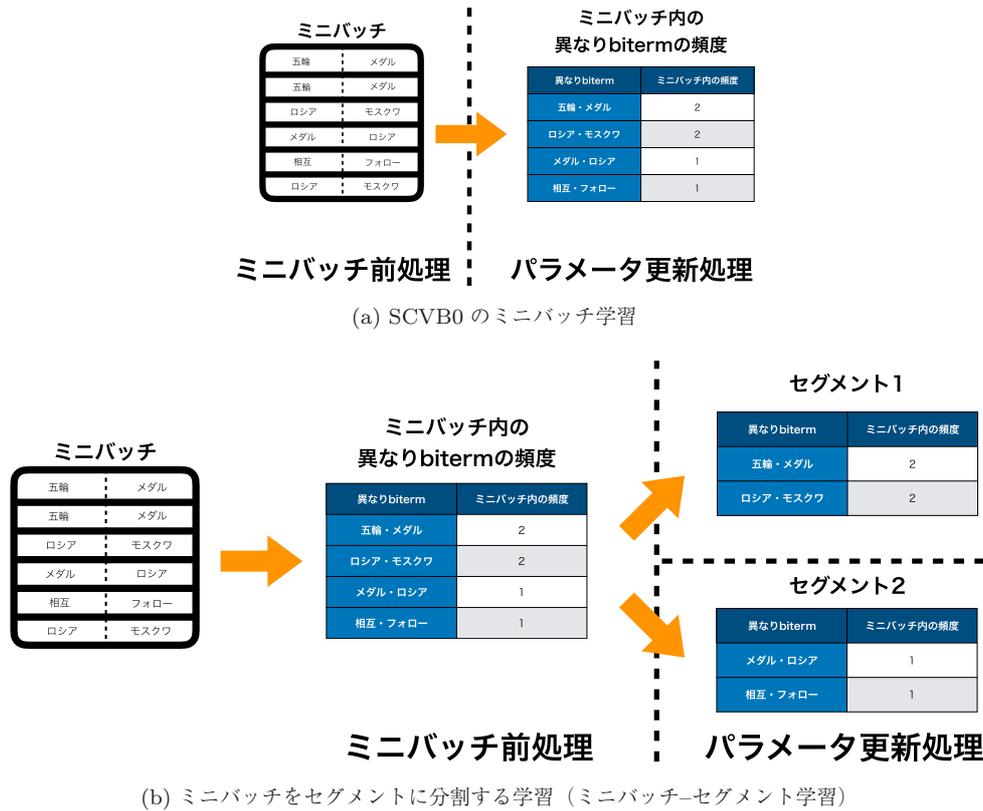


図 1 SCVB0 におけるミニバッチ学習とミニバッチ-セグメント学習. 提案手法 1 ではミニバッチ全体でパラメータ更新処理を行うことに対して, 提案手法 2 ではミニバッチをさらにセグメントに分割してセグメントごとにパラメータ更新を行う

Fig. 1 The proposed method for improving SCVB0. In minibatch training, the parameters are updated for each minibatch. In minibatch-segment training, each minibatch is further divided into segments for updating parameters.

セグメントにおける N_k と $N_{w|k}$ の更新では, 式 (15) と式 (16) における $\mathcal{V}_{\mathbf{B}^{(s)}}$ を $\mathcal{V}_{\mathbf{B}^{(s)}^{(j)}}$ に置き換えて計算が行われる. ミニバッチ-セグメント学習アルゴリズムの 1 イテレーションあたりの計算量は, ミニバッチ学習と同じ $O(|\mathcal{V}_{\mathbf{B}^{(s)}}|K)$ である.

4.3 単位時間あたりのトピック出現量の計算と近似

本研究では, 各単位時間のトピック出現量を求めるために, その時間 t に投稿されたツイート Ω_t に含まれる biterm に対して BTM で抽出したトピックの割合を計算し, ツイートに含まれるトピック k の割合の単位時間あたりの合計 $N_{k|t}$ を式 (14) によって計算する. 式 (14) の計算には, $\mathcal{V}_{B,t}$ に含まれるすべての biterm のトピック割合を式 (11) によって計算する必要がある, これがトピック出現量の計算のボトルネックになっていると考えられる.

各単位時間のトピック出現量の計算時間を短縮するために, 本研究では, $\mathcal{V}_{B,t}$ から一部を取り出し, 取り出された biterm のトピックに基づいて各トピックの出現量の計算を近似的に行う. トピック出現量に対する寄与の大きい biterm を特定する手がかりとして, 式 (14) で用いられる以下の項を “biterm 重み” $\omega_{t,b}$ として用いる.

$$\omega_{t,b} = \sum_{\sigma \in \Omega_t} P(b|\sigma). \quad (17)$$

この定義に基づいて, 取り出す biterm の決定方法として以下の手法を比較・検討する.

- biterm 重みの離散分布からサンプリングする.
- 異なり biterm からランダムに選択する.
- biterm 重みの高い順に選択する.

biterm 重みの離散分布からサンプリングする手法では, $\omega_{t,b}$ を biterm 重みの総和 $\sum_{b' \in \mathcal{V}_{B,t}} \omega_{t,b'}$ で割った値を biterm b についての離散分布のパラメータと見なし, この分布からあらかじめ決めた回数だけランダムにサンプリングを行い, このサンプリングされた biterm の集まり Ω'_t およびこれに含まれる異なり biterm 集合 $\mathcal{V}'_{B,t}$ を用いて式 (14) を計算する. 異なり biterm からランダムに選択する手法では, 異なり biterm をランダムに一定数選択し, トピック出現量の計算では元の biterm 重みを使う. biterm 重みの高い順に選択する手法では, 異なり biterm の数が一定になるまで, biterm 重みの高い順に異なり biterm を取り出す. 異なり biterm からランダムに選択する手法と同様に, トピック出現量の計算では元の biterm 重みを使う. いずれの手法も, 元の式 (11) で計算されるトピック出

現量に対する近似値を計算する手法である。

5. 評価実験

提案手法の有効性を検証するため、実際のツイートデータから提案手法によってトピック出現量の推定を行う。

5.1 実験データ

実験では、2012年8月1日から2012年8月7日の間に投稿されたツイートデータを用いる。ツイートデータの総数は286,223,678、1日あたりの平均ツイート数は40,889,096である。本研究では、ノイズとなる語を除去するために、ツイートに含まれる語彙に対して以下の制約を設ける。

- (1) 固有名詞・一般名詞・サ変接続の名詞以外の品詞を持つ単語を除く。
- (2) 2字以上の漢字・ひらがな・カタカナ・数字の組み合わせで構成されている単語以外を除く。
- (3) リツイートを示す「RT」・リプライを示す「@ユーザー名」・ハッシュタグを示す「#」に続く文字列およびURLを除く。
- (4) 笑いを意味する「w」・「W」・「笑」の単語を除く。

除去した結果、1つ以上の単語を含むツイートの総数は243,600,235で、1日あたりの平均ツイート数は34,800,034である。上記の制約を満たしたツイートデータの単語数について集計した結果を図2に示す。図2より、含まれる単語の数が10以下であるツイートが多くの割合を占めることが分かる。

BTMの学習のためにこれらのツイートからbitermの抽出を行う。抽出されたbitermの総数は3,077,224,154であり、異なりbiterm数は355,980,502である。本研究では、データ削減のために頻度が10以下のbitermを処理対象から除く。この結果、実験に用いるbitermの総数は2,453,945,076、異なりbiterm数では32,016,826である。

実験は、OSがUbuntu 16.04、CPUがIntel Xeon E5-2630 (2.40 GHz) 8core 2機の計算機上で行った。前処理

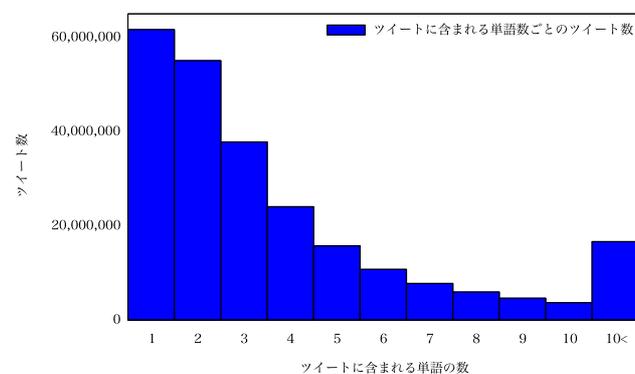


図2 ツイートに含まれる単語数ごとのツイート数

Fig. 2 The number of tweets per the number of words that contained in tweets.

における形態素解析器は ipadic を用いた MeCab [14]、各アルゴリズムの実装は Python で行った。

5.2 トピック学習の高速化の評価

提案手法によってトピックの学習が高速化されていることを検証するため、既存手法との比較を行った。

5.2.1 比較手法

実験データとして抽出した biterm に対して、以下の3種類の手法によってトピックを学習する。

SCVB0 提案手法のベースとなる BTM における SCVB0 の推論アルゴリズムを、ベースラインとして用いる。

この手法では、biterm を1つずつランダムに抽出し、式(4)および式(5)によって N_k と $N_{w|k}$ の更新を行う。

提案手法 (ミニバッチ学習) ミニバッチサイズの数だけ biterm を抽出し、式(15)および式(16)によって N_k と $N_{w|k}$ の更新を行う。ミニバッチサイズが1のとき、比較手法となる SCVB0 と同一の手法となる。

提案手法 (ミニバッチ-セグメント学習) ミニバッチサイズの数だけ biterm を抽出し、ミニバッチ内の異なり biterm をセグメントサイズずつ分割し、セグメントごとに N_k と $N_{w|k}$ の更新を行う。セグメントサイズが1のとき、ミニバッチ学習と同一の手法となる。

5.2.2 パラメータ設定

BTMのトピック数 K を512で固定する。またハイパーパラメータ α と β はそれぞれ $\alpha = 50/K$ と $\beta = 1/W$ とする。ただし、 α は BTM の学習と同時に式(9)によってトピックごとに更新される。減衰重み ν に関するパラメータである τ と κ は、Awaya ら [2] の実験と同一のパラメータである $\tau = 1,000$ と $\kappa = 0.8$ とする。

5.2.3 実験方法

本実験では提案手法が、SCVB0 と比較して、モデルの汎化性能を劣化させることなく、処理時間が短縮できることを確認する。実験では、実験用に1億個の biterm をサンプリングし、9:1の割合で学習用データと評価用データに分割し、学習用データで学習したモデル*2に対する評価用データの平均対数尤度を比較する。本研究では、評価用データに対する平均対数尤度を式(18)で計算する。

$$\frac{1}{|\mathbf{B}_{\text{test}}|} \sum_{b_i \in \mathbf{B}_{\text{test}}} \log \sum_{k=1}^K \theta_k \phi_{k,w_{i,1}} \phi_{k,w_{i,2}} \quad (18)$$

トピック分布のハイパーパラメータの更新は、biterm を10,000,000個サンプリングすることで行う。この最適化における不動点反復法の反復回数は20回とする。各手法で

*2 各手法では、 N_k と $N_{w|k}$ の更新で使われる式(1)による biterm のトピック確率の計算を、ループ処理ではなく、行列やベクトルに変形し、更新で使われるすべての biterm に対して一度に計算する。本研究では効率的な数学演算ライブラリとして、Intel Math Kernel Library (<https://software.intel.com/en-us/mkl>) を用いる。

学習時に抽出される総 biterm 数は 1 億とする。SCVB0 では、1 サンプルの学習を 1 イテレーションとして扱い、このイテレーションを 1 億回行う。提案手法のミニバッチ学習では、ミニバッチサイズ分の学習を 1 イテレーションとして扱い、ミニバッチサイズとイテレーション回数の積が 1 億になる組み合わせで行う。提案手法のミニバッチ-セグメント学習では、ミニバッチサイズを 1,000,000、イテレーション回数を 100 に固定し、セグメントサイズを変更する。

各手法では、alias 法 [15], [16] によって biterm をサンプリングする。alias 法は、与えられた離散分布について alias テーブルと呼ばれるデータ構造を構築することで、当該の離散分布からのサンプリングを $O(1)$ で実行可能とする手法である。alias 法における離散分布は、各 biterm の出現頻度をその総和から割った値によって求める。

本研究では、ミニバッチ学習におけるミニバッチサイズとイテレーション回数の組合せの候補を決定するために、予備実験を行った。予備実験では、学習用データおよびすべての biterm による実験データにおいてミニバッチを 100 回抽出し、ミニバッチに含まれる異なり biterm 数の平均値を計算した。表 1 に示した予備実験の結果より、ミニ

表 1 ミニバッチに含まれる異なり biterm 数の平均値。各値はそれぞれのデータに対してミニバッチを 100 回抽出したときの異なり biterm 数の平均値である

Table 1 The average of the number of unique biterm in mini batch. We calculated the average of each value from 100 experiments.

ミニバッチサイズ	異なり biterm 数の平均値
5	5.000
10	10.000
100	99.989
1,000	997.740
10,000	9860.823
100,000	93673.470
1,000,000	779655.323

表 2 各手法における学習用データを学習させたときの評価用データの平均対数尤度と学習に掛かる処理時間

Table 2 The average log-likelihood of the evaluation data and the average of the training time.

手法	ミニバッチサイズ	イテレーション	セグメントサイズ	平均対数尤度	処理時間 [sec]
SCVB0	1	100,000,000	-	-19.231	10,496 ± 103
ミニバッチ学習	10,000	10,000	-	-19.088	7,253 ± 154
	100,000	1,000	-	-19.706	7,077 ± 174
	1,000,000	100	-	-21.517	6,315 ± 149
ミニバッチ-セグメント学習	1,000,000	100	10	-18.998	5,797 ± 144
			100	-18.987	4,822 ± 9
			1,000	-18.995	5,282 ± 123
			10,000	-19.100	5,931 ± 143
			100,000	-19.875	6,074 ± 40

バッチサイズを大きくすればするほどミニバッチサイズに対する異なり biterm 数の割合が小さくなることが分かる。一方で、ミニバッチサイズの大きさが 1,000 以下の場合、biterm の推論回数の削減される割合が 1%未満となることから、ミニバッチ学習による高速化が期待できない。したがって本研究では、ミニバッチ学習の手法においてミニバッチサイズを 10,000 以上と設定し、実験を行う。

5.2.4 実験結果

各手法における学習用データを学習させたときの評価用データの平均対数尤度と学習に掛かる処理時間を表 2 に示す。ベースラインである SCVB0 と比較して、提案手法であるミニバッチ学習およびミニバッチ-セグメント学習の処理時間はいずれの条件においても短縮されていることが分かる。特にミニバッチ学習では、ミニバッチサイズを大きくすればするほど処理時間が短縮されていることから、ミニバッチに含まれる異なり biterm の頻度の数えあげにより効率的な処理が行われていることが分かる。

一方で平均対数尤度は、パラメータによって優劣に差がみられる。SCVB0 よりも平均対数尤度が高い手法は、表中で太字で示している。特にミニバッチ-セグメント学習は、セグメントサイズが $10 \cdot 100 \cdot 1,000$ の場合、SCVB0 およびミニバッチ学習のすべての条件と比較して、平均対数尤度が高い。これは、ミニバッチサイズに対して小さなセグメントサイズを設定することで変分パラメータの更新頻度が増加し、学習が早く進んだためであると考えられる。

ミニバッチ-セグメント学習は、ミニバッチ学習と計算量が同じであるため、同じミニバッチサイズのミニバッチ学習と処理時間が同程度になることが期待されるが、実際には処理時間が短くなっている。この理由については、CPU のキャッシュ効果の影響などが考えられるが、現時点では明確にできていない。処理時間についてのより詳細な分析は今後の課題とする。

SCVB0・ミニバッチ学習 (ミニバッチサイズ 10,000 かつイテレーション 10,000)・ミニバッチ-セグメント学習 (セ

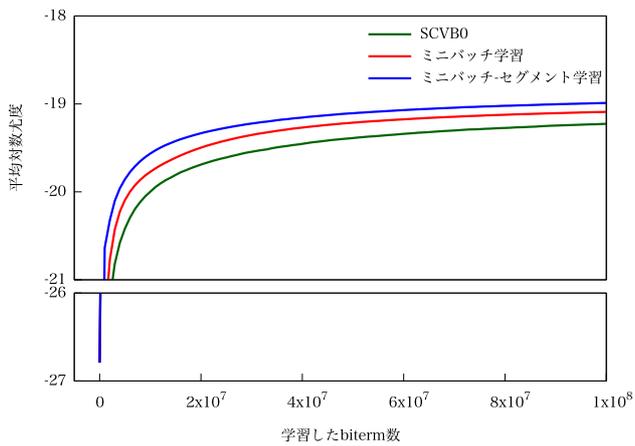


図 3 各手法の学習した biterm 数と平均対数尤度の変化
 Fig. 3 The average log-likelihood for processing biterms.

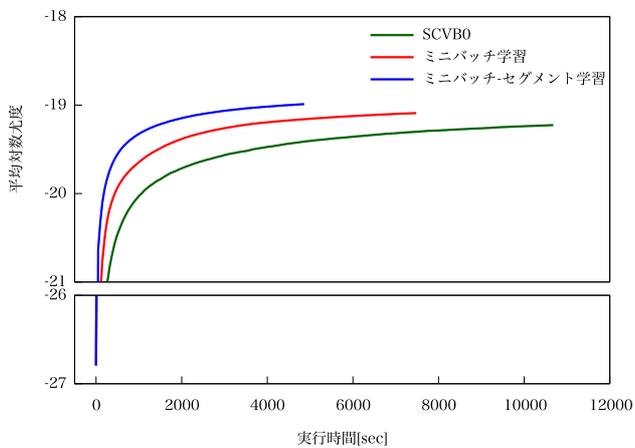


図 4 各手法の実行時間と平均対数尤度の変化
 Fig. 4 The average log-likelihood for running time.

グメントサイズ 100) における学習で学習した biterm 数に対する平均対数尤度の変化を図 3 に示す. SCVB0 と比較して, ミニバッチ学習およびミニバッチ-セグメント学習は学習した biterm 数の増加に対する平均対数尤度の収束が早いことが分かる. またミニバッチ学習とミニバッチ-セグメント学習を比較した場合, ミニバッチ-セグメント学習の方が平均対数尤度が早い段階で収束する. したがって, SCVB0 と比較して提案手法の方がより効率的に学習が進んでいることが分かる. 次に, 横軸を実行時間とした各手法の平均対数尤度を図 4 に示す. 実行時間の増加に対する平均対数尤度の増加の収束を比較すると, SCVB0 に対して, ミニバッチ学習およびミニバッチ-セグメント学習の収束が早く, 特にミニバッチ-セグメント学習が最も早く収束することが分かる. この結果から, 提案手法の効率的な学習が有効に働いていることが分かる. 以上より, 適切にミニバッチサイズとセグメントサイズを設定することで, 提案手法は SCVB0 よりも汎化性能が向上しつつ, 処理時間が高速化することができると考えられる.

5.3 biterm の一部を抽出しトピック出現量を近似する方法の評価

5.3.1 実験方法

提案手法の有効性を示すため, すべての biterm を使って推定されるトピック出現量に対する近似精度と推定に掛かる実行時間を比較する. 本実験では, 式 (14) によって計算された各单位時間におけるトピック出現量に対して, 4.3 節であげた手法で近似したトピック出現量との平均平方二乗誤差 (Root mean square error; RMSE) とトピック出現量の計算に掛かる処理時間を比較する*3.

近似したトピック出現量 $\hat{N}_{k|t}$ は元のトピック出現量 $N_{k|t}$ に対して単位時間あたりのトピックの出現量の総和が異なるため, 式 (19) によってスケールを揃えた近似値 $\tilde{N}_{k|t}$ を求める.

$$\tilde{N}_{k|t} = \frac{\sum_{k'=1}^K N_{k'|t} \hat{N}_{k|t}}{\sum_{k'=1}^K \hat{N}_{k'|t}} \quad (19)$$

すべての biterm によって計算されたトピック出現量 $N_{k|t}$ に対する $\tilde{N}_{k|t}$ の RMSE の平均値を式 (20) で計算する.

$$\frac{1}{T} \sum_{t=1}^T \sqrt{\frac{1}{K} \sum_{k=1}^K (N_{k|t} - \tilde{N}_{k|t})^2} \quad (20)$$

式 (20) による RMSE の平均値は単位時間あたりの各トピックの出現量の誤差の平均値を意味する.

biterm 重みの離散分布からサンプリングする手法では, 各单位時間で異なり biterm 数が 100,000・500,000・1,000,000 をそれぞれ超えるまで biterm のサンプリングを続ける. この手法では, 5.2 節と同様に alias 法 [15], [16] によって biterm をサンプリングする. 異なり biterm からランダムに選択する手法と biterm 重みの高い順に選択する手法では, 選択する biterm の数を 1,000,000 とする. 以上の手法によるトピック出現量の計算をそれぞれ 5 回行い, その RMSE と処理時間の平均値を比較する. ただし biterm 重みの高い順に選択する手法では, 一意な RMSE が計算されるため, 処理時間のみ平均値を計算する.

5.3.2 実験結果

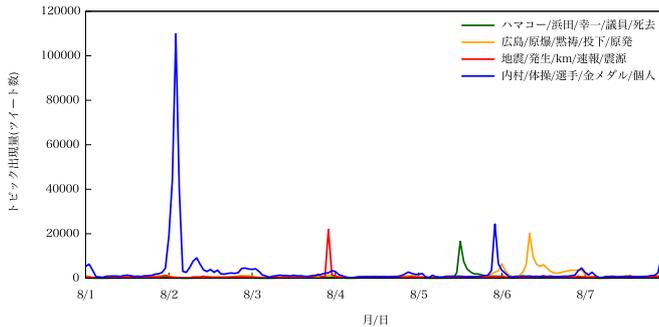
元のトピック出現量に対する RMSE と計算に掛かった処理時間を表 3 に示す. 表中の異なり biterm 数は, 各单位時間あたりの平均値を示している. 元のトピック出現量の計算では, 異なり biterm が単位時間あたりに平均 4,026,528 個存在し, その処理時間が 7 時間 53 分ほどかかるが, 提案手法により大幅に処理時間を削減できることが分かる.

*3 トピック出現量を計算するトピックは, 5.2 節の実験において最も平均対数尤度の高いトピックが抽出されたミニバッチ-セグメント学習で推定する. ミニバッチ-セグメント学習の設定は, ミニバッチサイズ 1,000,000・イテレーション 100・セグメントサイズ 1,000 の設定で行った. この設定は, すべての biterm を使って学習したトピックにおいて, 学習した biterm から計算した平均対数尤度が最も高い設定である.

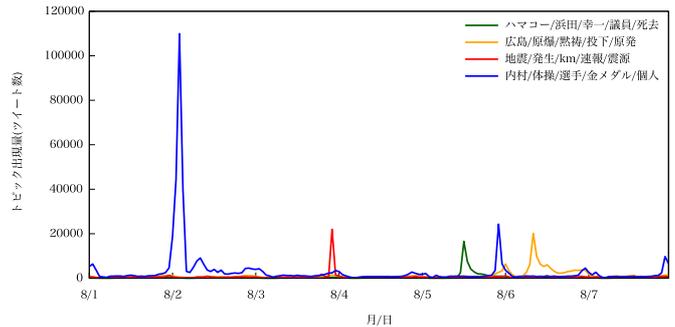
表 3 元のトピック出現量に対する各手法の RMSE と処理時間. RMSE および処理時間は 5 回の実験の平均値とする

Table 3 RMSE of each method for baseline. We calculated the average of each value from 5 experiments.

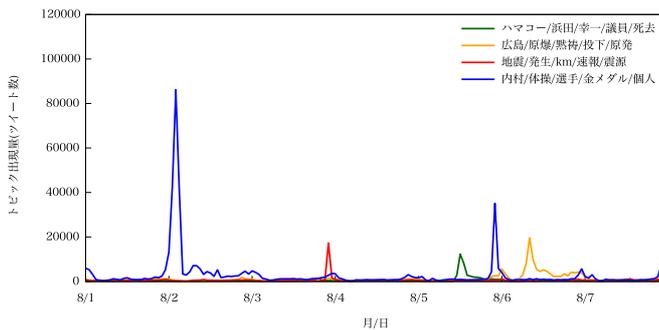
手法	異なり biterm 数	平均 RMSE	処理時間 [sec]
すべての biterm 重みを使う (元のトピック出現量)	4,026,528	-	28,351 ± 108
biterm 重みの離散分布からサンプリングする	100,000	60.565	2,263 ± 162
	500,000	22.509	5,503 ± 21
	1,000,000	13.340	10,263 ± 34
異なり biterm からランダムに選択する	1,000,000	213.840	7,788 ± 319
biterm 重みの高い順に選択する	1,000,000	217.605	8,526 ± 51



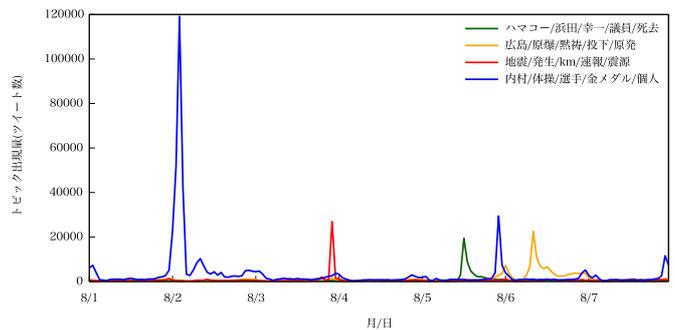
(a) すべての biterm 重みを使う手法 (元のトピック出現量)



(b) biterm 重みの離散分布からサンプリングする (異なり biterm 数: 100,000)



(c) 異なり biterm からランダムに選択する



(d) biterm 重みの高い順に選択する

図 5 各手法によって計算されるトピック出現量. 凡例は各トピックで出現確率の高い語を順に左から並べたものである

Fig. 5 The time series variation for each topic popularity by each method. In the legend, the words with high occurrence probability are shown in each topic from the left.

異なり biterm 数 1,000,000 の条件においては, 異なり biterm からランダムに選択する手法や biterm 重みの高い順に選択する手法の方が, biterm 重みの離散分布からサンプリングする手法に比べて処理時間が短い, 元のトピック出現量に対する RMSE は大きくなってしまふ. また, biterm 重みの離散分布からサンプリングする手法は, 用いる異なり biterm 数を少なくすることで, 異なり biterm からランダムに選択する手法や biterm 重みの高い順に選択する手法よりも処理時間が短くなり, かつ, RMSE も低くなることから, より優れた手法といえる.

biterm 重みの高い順に選択する手法の誤差は最も誤差が大きくなっているが, これは単位時間において出現頻度の

高い biterm のみを考慮しているため, ポピュラーなトピックの出現量を過剰に大きく推定していることが原因と考えられる. これに対して, 異なり biterm をランダムに選択する手法や biterm 重みの離散分布からサンプリングする手法では, 出現頻度の低い biterm もトピック出現量の計算に考慮されるため, 重みの低い biterm に関するトピックの出現量の精度が高まり, 結果として全体のトピック出現量の誤差の平均値が低くなったと考えられる.

実際に各手法によって計算されたトピック出現量を図 5 に示す. いずれの手法もスケールに誤差が生じているが, 時間経過によるトピック出現量の増減の概形は同じであることから, トレンドを把握する目的には大きな問題にはな

らないと考えられる。また, biterm 重みの離散分布からサンプリングする手法に基づいて計算された図 5(b) の結果では, スケールに対する誤差も小さいことが分かった。

6. 結論

本研究では, SCVB0 に対して, ミニバッチ学習をベースとした拡張と近似的なトピック出現量の計算によって, より高速な学習とトピック出現量の計算を行う手法を提案した。実験では, 2012 年 8 月 1 日から 8 月 7 日に投稿されたツイートデータに対して, 提案手法を適用し, トピックとトピック出現量を抽出した。この結果, 提案手法によるトピック抽出では既存手法に劣らない汎化性能を保ちつつ, 高速な学習が行われていることが確認された。また最も誤差が少ない近似手法が biterm 重みの離散分布からサンプリングする手法であることを確認した。

今後の展望として, BTM のトピックの継続的な利用が考えられる。ツイートのトピックを継続的に追跡する場合, 提案手法では何度もモデルを学習し直す必要があるため, 時間的なコストがかかる。今後は, 一度使用したモデルを再利用する手法を考案し, 継続的な利用が可能であるか検証を行うことが必要だと考えられる。また, 本研究で提案した高速化手法を, トピック出現量の時間的推移の抽出以外の目的において適用した場合の有効性を明らかにすることも, 今後の課題である。

謝辞 本研究の一部は, JSPS 科研費 (課題番号 16H02904) および筑波大学図書館情報メディア系プロジェクト研究の助成によって行われた。

参考文献

- [1] Cheng, X., Yan, X., Lan, Y. and Guo, J.: BTM: Topic modeling over short texts, *IEEE Trans. Knowledge and Data Engineering*, Vol.26, No.12, pp.2928–2941 (2014).
- [2] Awaya, N., Kitazono, J., Omori, T. and Ozawa, S.: Stochastic collapsed variational Bayesian inference for biterm topic model, *2016 International Joint Conference on Neural Networks (IJCNN)*, pp.3364–3370 (2016).
- [3] Blei, D.M., Ng, A.Y. and Jordan, M.I.: Latent Dirichlet Allocation, *Journal of Machine Learning Research*, Vol.3, pp.993–1022 (2003).
- [4] Tang, J., Meng, Z., Nguyen, X., Mei, Q. and Zhang, M.: Understanding the Limiting Factors of Topic Modeling via Posterior Contraction Analysis, *Proc. 31st International Conference on Machine Learning, Proc. Machine Learning Research*, Vol.32, No.1, Beijing, China, PMLR, pp.190–198 (2014).
- [5] Blei, D. and Lafferty, J.: Dynamic Topic Models, *ICML '06 Proc. 23rd International Conference on Machine Learning*, pp.113–120 (2006).
- [6] Iwata, T., Yamada, T., Sakurai, Y. and Ueda, N.: Online multiscale dynamic topic models, *KDD '10 Proc. 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.663–672 (2010).
- [7] Koike, D., Takahashi, Y., Utsuro, T., Yoshioka, M. and

Kando, N.: Time Series Topic Modeling and Bursty Topic Detection of Correlated News and Twitter, *International Joint Conference on Natural Language Processing*, pp.14–18 (2013).

- [8] Cui, Z., Sato, I. and Sugiyama, M.: Stochastic Divergence Minimization for Biterm Topic Models, *IEICE Trans. Information and Systems*, Vol.E101.D, No.3, pp.668–677 (2018).
- [9] Teh, Y.W., Newman, D. and Welling, M.: A Collapsed Variational Bayesian Inference Algorithm for Latent Dirichlet Allocation, *Proc. 19th International Conference on Neural Information Processing Systems, NIPS '06*, pp.1353–1360 (2006).
- [10] Asuncion, A., Welling, M., Smyth, P. and Teh, Y.W.: On Smoothing and Inference for Topic Models, *Proc. 25th Conference on Uncertainty in Artificial Intelligence, UAI '09*, pp.27–34 (2009).
- [11] Wallach, H.M., Mimno, D.M. and McCallum, A.: Rethinking LDA: Why Priors Matter, *Advances in Neural Information Processing Systems 22*, Bengio, Y., Schuurmans, D., Lafferty, J.D., Williams, C.K.I. and Culotta, A. (Eds.), Curran Associates, Inc., pp.1973–1981 (2009) (online), available from (<http://papers.nips.cc/paper/3854-rethinking-lda-why-priors-matter.pdf>).
- [12] Foulds, J., Boyles, L., DuBois, C., Smyth, P. and Welling, M.: Stochastic Collapsed Variational Bayesian Inference for Latent Dirichlet Allocation, *Proc. 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.446–454 (2013).
- [13] Sato, I. and Nakagawa, H.: Stochastic Divergence Minimization for Online Collapsed Variational Bayes Zero Inference of Latent Dirichlet Allocation, *Proc. 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.1035–1044 (2015).
- [14] Kudo, T., Yamamoto, K. and Matsumoto, Y.: Applying Conditional Random Fields to Japanese Morphological Analysis, *Proc. 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP-2004)*, pp.230–237 (2004).
- [15] Walker, A.J.: An Efficient Method for Generating Discrete Random Variables with General Distributions, *ACM Trans. Math. Softw.*, Vol.3, No.3, pp.253–256 (online), DOI: 10.1145/355744.355749 (1977).
- [16] Wang, J., Wan Tsang, W. and Marsaglia, G.: Fast Generation of Discrete Random Variables, *Journal of Statistical Software*, Vol.11 (2004).



福山 怜史

2017 年福井大学工学部情報・メディア工学科卒業。現在, 筑波大学大学院図書館情報メディア研究科博士前期課程在学中。ソーシャルメディアに焦点を当てたデータマイニングの研究に従事。



若林 啓 (正会員)

2012年法政大学大学院工学研究科博士課程修了。博士(工学)。同年、筑波大学図書館情報メディア系助教。機械学習の研究に従事。電子情報通信学会、日本データベース学会、ACM各正会員。

(担当編集委員 平松 薫)