N-gram over Context

Noriaki Kawamae
NTT Comware
1-6 Nakase, Mihama-ku, Chiba, Japan
Japan 261-0023
kawamae@gmail.com

ABSTRACT

Our proposal, N-gram over Context (NOC), is a nonparametric topic model that aims to help our understanding of a given corpus, and be applied to many text mining applications. Like other topic models, NOC represents each document as a mixture of topics and generates each word from one topic. Unlike these models, NOC focuses on both a topic structure as an internal linguistic structure, and N-gram as an external linguistic structure. To improve the quality of topic specific N-grams, NOC reveals a tree of topics that captures the semantic relationship between topics from a given corpus as context, and forms N-gram by offering power-law distributions for word frequencies on this topic tree. To gain both these linguistic structures efficiently, NOC learns them from a given corpus in a unified manner. By accessing this entire tree at the word level in the generative process of each document, NOC enables each document to maintain a thematic coherence and form N-grams over context. We develop a parallelizable inference algorithm, D-NOC, to support large data sets. Experiments on review articles/papers/tweet show that NOC is useful as a generative model to discover both the topic structure and the corresponding N-grams, and well complements human experts and domain specific knowledge. D-NOC can process large data sets while preserving full generative model performance, by the help of an open-source distributed machine learning framework.

Keywords

Nonparametric models, Topic models, Latent variable models, Graphical models, N-gram topic model, MapReduce

1. INTRODUCTION

As the sheer volume of user generated content on the Web now exceeds the individual human processing capabilities, statistical topic models are an essential component of natural language processing for human-computer interaction, and statistical machine learning. Specifically, Latent Dirichlet Allocation (LDA) [3] and its extensions are seen as particularly useful models for improving web services, such as information retrieval, knowledge discovery, and social media analysis, and then enjoy success in these domains. They define a probabilistic scheme for generating each document as a mixture of low-dimensional thematic units (topics), where each topic is a multinomial distribution over words, and is interpretable by the highest probability words in it. Since this topic presentation and its applications fit our goal more than the other presentation, our approach is built on topic models. The majority of the state-of-the-art topic models rely on the bag-of-words assumption, that ignores word order, and break the document's structure.

As N-grams carry more meaning than the sum of individual words and their meaning depend on context, some topic models focus on the order of words [12, 15, 18] as an external linguistic structure. However, these models miss a internal linguistic structure such as relationships between topics that exist in both a given corpus and each document. As an internal linguistic structure, a semantic structure facilitates the reader's comprehension at granularities finer than topics. Although introducing both these linguistic structures simultaneously would make topic models complicated, they are essential for applications to capture the thematic coherence behind documents. This is the motivation why we propose a new topic model to maintain both these linguistic structures, since no previous topic models perfectly fit for these requirements.

Motivated by these requirements, our proposal new topic model, N-gram over Context (NOC), focuses on a topic tree as an internal structure, and forms N-gram on this tree as an external structure. NOC aims to help our understanding of a given corpus, and be applied to many text mining applications. To achieve this goal, this model is based on assumptions that 1)the quality of N-gram depends on the discovered topics and their structure, and 2)these internal and external linguistic structure are complementary and help each other in their learning process. Indeed, we can see that a hierarchical topic structure represents a semantic unit and forms context in each document. Based on these assumptions, NOC learns these structures together from a given corpus, and reveals both a topic hierarchy and topic specific N-grams together in a unified model. As these structures vary on a corpus, NOC employs a nonparametric approach. Given the very large or infinite number of predictive distributions over phrases, their tree structures exist within each topic and are shared for parameter estimation.

WWW 2016, April 11–15, 2016, Montréal, Québec, Canada.
ACM 978-1-4503-4143-1/16/04.
Include the http://dx.doi.org/10.1145/2872427.2882981.

Then, NOC constructs a hierarchical topic tree that captures the semantic relationship between topics in a given corpus, and offers power-law distributions for word frequencies on N-gram on this tree. Consequently, NOC explains the generative process of each document by using fine-grained topic hierarchies with N-gram words.

Through experiments, we confirm the following contributions of NOC.
Theoretical contribution: We extend N-gram topic models to obtain an understandable structure that shows which topics are present in the corpus using both internal/external linguistic structures, and how these topics are semantically connected without using domain-specific dictionaries. For this goal, we show how to construct, utilize, and learn this tree, by employing a nonparametric approach. Since learning these structures together helps each other, NOC can grasp them more clearly than learning them individually. Although this structure seems complicated, this topic tree serves as the required minimum constitution that constructs a thematic coherent topic tree for explaining the generative process of documents, and forms topical N-grams that reflect the syntactic structure.
Practical contribution: We develop an efficient approximation algorithm using distributed learning, a distributed version of NOC (D-NOC). This algorithm is designed to employ a Gibbs sampling and follow the MapReduce programming framework[1] for easy implementations. It conquers the complex dependencies included in hierarchical topic models, and could be the approximated Gibbs sampling of NOC. Since this learning framework is similar to that of Parameter servers [9] and Petuum [10] or Spark [11], D-NOC could be implemented on these promising platforms without major modification. NOC represents context as the document's linguistic structure, well complements both human experts and domain specific knowledge, and yields topic specific N-grams. These results support that NOC could be applicable to topic specific query suggestion and concept search.

2. PREVIOUS WORKS

2.1 N-gram topic models

Inspired by the assumption that the internal structure of documents exhibits meaningful content, and can help user's comprehension and retrieval experience, a considerable amount of research has attempted to explore word co-occurrence patterns, i.e. topics, embedded in documents, and their order, and then provide a predictive probability distribution for the next word following the previous words. For example, Identifying Sentiments over N-gram (ISN) model [14] extends Bigram topic model (BTM) [29] to obtain N-grams reflecting dependency on nearby context.

By employing smoothing methods that exhibit the power law characteristics, known as Zipf's law [33] in linguistics, topic models could gain the appropriate structure of a given text data. Phrase Discovering LDA (PDLDA) [18] incorporates Pitman-Yor process (PYP) [24] hierarchy into the process of forming phrases. Supervised N-gram topic model (SNT) [15] extends ISN by combining both PYP and supervision. Unfortunately, they miss the relationship between topics in forming N-grams, and fail to capture thematic coherence structure.

Since semantic structures are of significant interest, topic models should represent this structure in a thematically coherent manner. Although NTSeg [12] employs the notions of word-topics and segment-topics to maintain the document's structure, it is based on the flat topic structure and misses the semantic relationship between topics. While the composite model [8] captures the interaction between short and long range dependencies between words by using a hidden Markov Model (HMM), Syntactic Topic Model (STM) [6] uses parse trees as syntactic information, and generates each word of a sentence by a distribution that combines document-specific topic weights and parse-tree-specific syntactic transitions. These structures of both models would be applied or extended to N-gram topic models.

In describing the generative process of each document, NOC employs a hierarchical topic structure instead of flat structure. In this topic tree, semantic is contained in a structure consisting of parent topics and their child topics, where the former are more general than the latter.

2.2 Scalable topic models

A promising approach to scaling topic models over large data sets is to distribute and parallelize both the data and their algorithms over multiple processors [20, 22, 26, 32]. We focus on Markov chain Monte Carlo sampling [28] inference than variational inference [32], since the former has the preferable advantages such as easy implementations and sampling speed. Mimno et al. [20] propose parallel algorithms for the collapsed sampler for LDA, while Newman et al. [22] parallelize Hierarchical Dirichlet processes (HDP) [28] through the straightforward mapping approach. Smola et al. [26] proposed two parallelization paradigms for LDA: a decoupling between sampling and state updates for multi-core architectures and a blackboard architecture to deal with state synchronization for clusters of workstations. However, it is not clear how these approaches can be applied to non-parametric hierarchical topic models [4], since they contain complex dependencies which render the inference hard to parallelize and impose costly alignment overheads between nodes. Although applying topic models to web scale data sets is not as straightforward as one may think, modern machine learning technology would enable inference algorithms to parallelize over multiple processors, and makes models computationally feasible and discovers meaningful phrases in their correct context.

Our challenge is to continue this approach for more complicated hierarchical topic modeling using collapsed Gibbs sampling, build D-NOC on top of an open-source distributed machine learning framework, and show a pseudo code for easy implementation. We showed that shortcomings inherent in Gibbs sampling [32] could be alleviated by 1)proposed approximated algorithm, and 2)usage of Memcached.

2.3 Chinese restaurant process and its extensions

The Chinese restaurant process (CRP) [2] is a stochastic process that defines a distribution on the space of partitions of the positive integers and is used as a metaphor to illustrate the Dirichlet process (DP) [21]. This process allows the number of clusters to grow as new data points are observed, and yields the same clustering structures as created by a

[1]Apache$^{\mathrm{TM}}$Hadoop®: http://hadoop.apache.org/

Table 1: Top frequent 2-gram extracted from 1564 ACM conference papers with data science published in 2014: Each group (title) is made by hand (interpreted by 2-gram), the value of () denotes the number of observed N-gram.

group	words
data science	big data (211), web search (176), social media (158), recommender system (87)
data analysis	machine learning (81), network analysis (75), text mining (43), user modeling (38)
data management	multi core (38), distributed computing (36), memory bandwidth (25), large scale (24)
topic model	topic model (19), Gibbs sampling (17), nonparametric bays (15), Dirichlet process (12),
submodular function	submodular function (6), greedy algorithm (6), coverage objective (3), max-cut (2)

DP [7]. A DP can be considered as a distribution of random probability measure G, which we write as $G \sim DP(\gamma, G_0)$, where γ is a scaling parameter, and G_0 is a base measure. Sethuraman [25] showed that measure G drawn from a DP is discrete, and defined by the stick breaking process.

Alternatively, we can view this process in terms of $CRP(\gamma, G_0)$ and compute the probability of the i-th customer sitting at table k in the restaurant j as follows:

$$P(z_i = k|\mathbf{z}_{\setminus i}, \gamma) = \begin{cases} \frac{n_{jk}}{\sum_k n_{jk}+\gamma}, k \text{ is an existing table,} \\ \frac{\gamma}{\sum_k n_{jk}+\gamma}, k \text{ is a new table.} \end{cases} \tag{1}$$

where $\mathbf{z}_{\setminus i}$ is the seating arrangement of the current $i-1$ customers, and n_{jk} is the number of customers sitting at table k in restaurant j.

PYP is a generalization of a hierarchical DP, and yields power-law behavior [24, 27]. For each word $w \in W$, let $G(w)$ be the estimated probability of w, and let $G = [G(w)]_{w \in W}$ be the vector of word probabilities. By placing a PYP prior on G_0, we gain G:

$$G \sim PYP(\gamma, d, G_0), \quad w \sim G, \tag{2}$$

where $G_0 (= [G_0(w)]_{w \in W})$ is the base probability measure of G and can be understood as the putative mean of draws from $PYP(\gamma, d, G_0)$, $0 \leq d < 1$ is a discount parameter that controls the power-law property of the distribution, and $\gamma > -d$ is a strength parameter.

The marginal of $PYP(\gamma, d, G_0)$ is described using the CRP metaphor. That is, the first customer sits at the first table, z_1=1, and the i customer chooses a table in j according to the following distribution,

$$P(z_i = k|\mathbf{z}_{\setminus i}, \gamma, d) = \begin{cases} \frac{n_{jk}-d}{\sum_k n_{jk}+\gamma}, \text{if } k \text{ is an existing table,} \\ \frac{\gamma+dK}{\sum_k n_{jk}+\gamma}, \text{if } k \text{ is a new table,} \end{cases} \tag{3}$$

where K is the current number of occupied tables. When the number of tables increases as many customers enter the restaurant with $d > 0$, this discount parameter yields a power-law.

The nCRP [5] extends the CRP representation to construct a set of topics, G, that are arranged over a tree-like structure whose semantic is contained in a structure consisting of parent topics and their children (topics), the former are more general than the latter, as follows:

$$G \sim nCRP(\alpha, G_0), \tag{4}$$

where α is a scaling parameter, and $\alpha \sim Gamma(e_1, e_2)$.

Both HDP and nCRP lead NOC to share the same topics among documents, where the probability weight on not only each topic, but also their topic level, varies with each document.

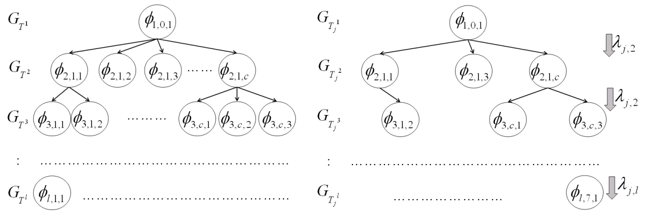


Figure 1: An example of structure for (left) the Global hierarchical topic tree, T, and (right) the j-th document specific hierarchical topic tree, T_j. Each tree is an infinite collection of DP and a transition rule between rules. Each node of the tree is associated with a topic, ϕ. Let l be the index of a topic level. The discrete set of atoms $\phi_{l,p,c}$ are drawn independently from G_0. This global topic tree is shared among documents via the corresponding T_j.

Table 2: Extensions of CRP: As DP can be considered as a distribution of random probability measure G, write $G \sim DP(\gamma, G_0)$, where γ is a scaling parameter, $\gamma \sim Gamma(e_1, e_2)$, and G_0 is a base measure.

	topic relation	accessible topics of each document generation
rCRP [17]	tree: $G \sim rCRP(\gamma)$	entire tree
nCRP [5]	tree: $G \sim nCRP(\gamma)$	a single path of tree
CRF [28]	flat: $G \sim DP(\gamma, G_0)$	entire topics
nCRF [1]	hierarchy:	entire topics
nHDP [23]	hierarchy:	entire topics

3. N-GRAM OVER CONTEXT

3.1 Motivation

Hierarchical semantic topic structure and N-gram: We explain our motivation using Table 1, where N-grams are extracted by NLTK [2] from ACM conference papers. This table shows that these groups seem to correspond topics of conventional topic models, and have a hierarchical semantic relationship. For example, both data analysis and data management groups is one of data science, and both topic model/sub modular would be one group of data analysis rather than the data management. As the number of tokens assigned to topic model/sub modular is fewer than the number of tokens associated with data analysis, topic model/sub modular specific words is fewer than those of data analysis. That is, these small topic specific words would be overwhelmed by other words and buried in each topic.

[2]Natural Language Toolkit: http://www.nltk.org/

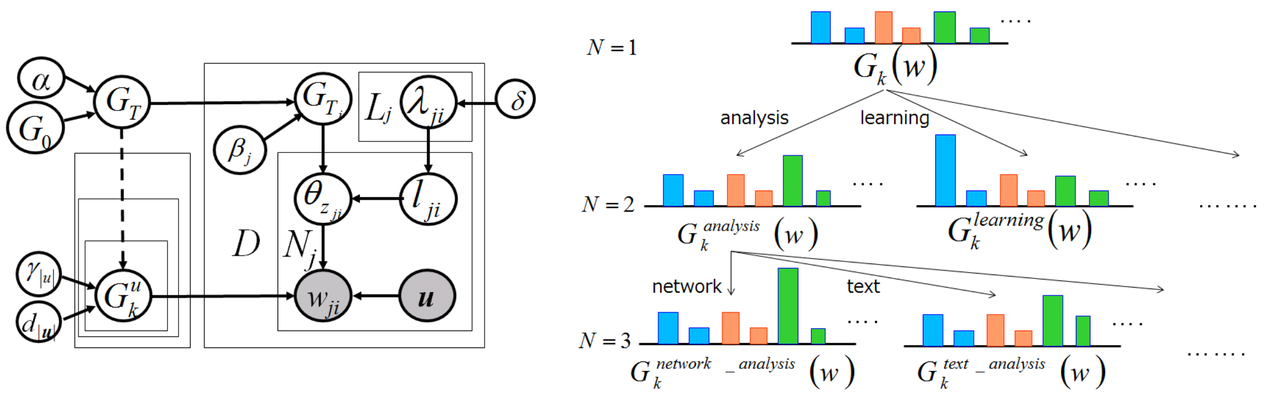


Figure 2: (left)Graphical Model of NOC: In this figure, shaded and unshaded variables indicate observed and latent variables, respectively. An arrow indicates a conditional dependency between variables and stacked panes indicate a repeated sampling with the iteration number shown. $\theta_{z_{ji}}$ is associated with one ϕ_k via the topic indicator, k. (right)Topic specific N-gram tree constructed from papers in Table 1: In this tree, $G_k^{\emptyset}(= [G_k^{\emptyset}(w)]_{w \in W})$ corresponds to ϕ_k of G_T, where $G_k^u(w)$ denotes the probability of w following u on k.

This problem leads N-gram topic models without the topic structure to miss these informative words/phrases and decrease the quality of N-grams. As the quality of topic specific N-grams seems to depend on the topic structure, we need a topic model to discover a fine grained topic tree from a given corpus, that prevents corresponding topic specific words from being buried.

Topic tree structure and thematic coherence: In the describing of generating each document, NOC differs from previous extensions of CRP in constructing and utilizing the topic tree. Using Table 2, we show how NOC employs a nonparametric approach to determine the appropriate topic structure, and discuss these differences. NOC allows each document to access this entire tree-structured hierarchy of topics. As nCRP permits each document to select only one path down the tree, similar topics will appear in many nodes of the tree and then grow the size of tree. Since a global topic tree is constructed from a given corpus, we can image that it might not to be a good fit for each document. NOC shares a hierarchical topic tree distribution as a base for a second level Dirichlet process (DP) [7] for each document, while HDP [28] shares a flat topic distribution as a base distribution among documents. That is, NOC constructs a document specific tree that shares the same topics with the global topic tree, and selects word specific paths that are thematically coherent via this document specific topic tree. Although the recursive Chinese Restaurant Process (rCRP) [17] enables a document to have a distribution over the entire topic tree, and samples a topic at the word level, it fails to ensure thematic coherence. While the nested Hierarchical Dirichlet process (nHDP) [23] extends nCRP to select a topic in a given document-specific topic tree, and then uses it on this constructed tree, NOC constructs a document specific tree, at the document level, to maintain the parent-child topic relationships more precisely than nHDP.

Summary: Like other topic models, NOC represents each document as a mixture of topics and generates each word from one topic. Unlike these models, NOC employs a hierarchical semantic topic structure that maintains the thematic coherence as context, and forms topic specific N-grams that provide the syntactic.

Table 3: Notation used in this paper

SYMBOL	DESCRIPTION
$D(W)$	♯documents (vocabulary size)
$N_j(L_j)$	♯words (topic level)in j-th document
$z_{ji}(l_{ji})$	the i-th topic (level) variable in j-th document
w_{ji}	the i-th word in j-th document
h_{ji}	the previous word sequence sharing with the same topic before w_{ji}
G_0	a $\|W\|$-dimensional uniform word distribution
$G_T(G_{T_j})$	the topic distribution of T (T_j)
$G_{T^l}(G_{T_j^l})$	the l-th level topic distribution of T (T_j)
$\phi_{l,p,c}(\phi_k)$	topic: the c-th child in l-th level of p-th parent topic in l-1 level (shorthand of $\phi_{l,p,c}$)
$\mathbf{u}$	a n-1 words sequence sharing the same topic
$G_k^{\mathbf{u}}$	the topic k specific word distribution following $\mathbf{u}$
$d_{\|\mathbf{u}\|}(\gamma_{\|\mathbf{u}\|})$	the $\|\mathbf{u}\|$ specific discount (concentration): parameter $d_{\|\mathbf{u}\|} \sim Beta(e_{\|\mathbf{u}\|}, f_{\|\mathbf{u}\|})$ $(\gamma_{\|\mathbf{u}\|} \sim Gamma(g_{\|\mathbf{u}\|}, h_{\|\mathbf{u}\|}))$
λ_{jl}	the l-th topic level of T_j specific beta random: variable $\lambda_{jl} \sim Beta(\delta_1, \delta_2)$
β_j	the j specific parameters: $\beta_{jl} \sim Gamma(a_1, a_2)$

3.2 Model Definition

3.2.1 T: Global hierarchical topic tree

Here, we define the global hierarchical topic tree, T, using Table 3 that shows the notation used in this paper, and explain it with Figure 1. This figure shows an example of a global hierarchical topic tree, T, and the topic distribution of T, G_T, where l-th level of T, and its topic distribution is denoted by T^l and G_{T^l}, respectively. The first level of the tree consists of only the background topic, and its children or descendant nodes on the l-th level are countably infinite, where the probability of transitioning to child node c equals to the probability of the c-th break of a stick-breaking construction [25].

3.2.2 T_j: *Document-specific hierarchical topic tree*

NOC allows each document to have a document-specific tree, T_j, the topic distribution of T_j, G_{T_j}, and share the set of topics and the parent-children topic relationships across over all documents via T. NOC uses each DP of T^l as a base for a second level DP drawn independently for constructing T_j^l and sampling a topic when generating each document. Then, this tree contains all thematic content of a document, and allows NOC to select a thematically consistent topic path at the word level rather than the document level. For selecting this word-specific path, NOC introduces a document-specific beta random variable, λ_{jl}, that acts as stochastic switch. The probability that document j uses the topic level l is

$$\tilde{\lambda}_{jl} \sim Beta(\delta^1, \delta^2), \lambda_{jl} = \tilde{\lambda}_{jl} \prod_{r=0}^{l-1}(1 - \tilde{\lambda}_{jr}), \qquad (5)$$

where δ^1, δ^2 are parameters, and l denotes topic level. Given a word in each token, this probability determines whether this word uses the current topic level l in a given topic tree, or continues to go down this tree. T_j has the same topic as T on the same topic level, but with different probability weights and topic level. This relationship constrains the increment of the number of topics and so prevents the global topic tree from generating many similar topics that will be observed in the nCRP. Therefore, the probability of the topic level will vary with the document, even if different documents share the same topics.

The difference between NOC and STM lies in how the per-word topic assignment is drawn, and in forming N-grams. STM represents a document-specific thematic component, and a syntactic component as the vector of topic weights, and the vector of transitions probability, respectively. STM merges these two vectors, and then generates words on this merged topic assignment. In NOC, the weight of syntactic components is conditioned on each thematic component, and syntactic components are common over a given corpus and only the thematic component weights are document dependent. NOC places syntactic components on document-specific thematic component T_j. This structures allows NOC to draw topics from T_j that share the semantic components with other documents via T, and from N-gram on the topic as a syntactic component.

3.2.3 *Topic-specific N-gram*

NOC yields a tree hierarchy of PYP priors by placing priors over the predictive probabilities recursively, as in previous works [27]. For notation simplification in the rest of paper, we represent $\phi_{l,p,c}$ by ϕ_k. As shown in Figure 2, NOC provides G_k^u that is N-gram word distribution following the previous word sequence, $\mathbf{u}$, on the same topic indicator, k, that is shared among $\mathbf{u}$. By placing PYP as a prior over G_k^u using Eq (2) recursively, we get to $G_k^{\emptyset}$ that corresponds to the topic, ϕ_k, of G_T. As there is no restriction on the number of consecutive same topics, $|\mathbf{u}|$ can be arbitrarily long but always shares the same topic.

As $G_k^{\mathbf{u}\backslash}$ can be also understood as the mean vector, and recursively placing a prior over $G_k^{\mathbf{u}\backslash}$ leads to k-specific unigram word distribution $G_k^{\emptyset}(= G_k)$ that corresponds to ϕ_k in G_T; this is the vector of probabilities over the current word without prior information. This process differentiates NOC from SNT in constructing a topic specific N-gram, except using supervisions. SNT places a background topic that corresponds to $\phi_{1,0,1}$ in NOC, as a base distribution, $G_k^{\emptyset}$, because SNT cascades all topics to the background topic and thereby neglects the relationships between topics except the background topic.

3.3 Generative process

With reference to the graphical model shown in Figure 2, the generative procedure of NOC is described as follows:

3.3.1 *Global hierarchical topic tree structure generation*

Each node in the tree defines a CRP over children and corresponds to an atom (topic) independent of G_0 using Eq (4):

- $G_T \sim nCRP(\alpha, G_0)$.

For each topic ϕ_k: $\phi_k \sim G_T$:

- For each word sequence $\mathbf{u} \in \{W\}, \ldots, \{W\}^{n-1}$, define $G_k^{\mathbf{u}}$ using Eq (2): $G_k^{\mathbf{u}} \sim PYP(\gamma_{|\mathbf{u}|}, d_{|\mathbf{u}|}, G_k^{\mathbf{u}\backslash})$, where $\mathbf{u}\backslash$ is the suffix of $\mathbf{u}$ consisting of all but the earliest word.

Since N is learnt from a given corpus and could be increased later, $N \leq 3$ might be sufficient enough setting at first.

3.3.2 *Document-specific topic tree structure generation*

For each document j ($j = 1$ to D), construct the j specific tree T_j and the l-th topic level specific parameters λ_{jl}:

- For each level of DP in T, define a second level DP with the corresponding G_{T^l}: $G_{T_j^l} \sim DP(\beta_j, G_{T^l})$.
- For each level in T_j, define a beta random variable: $\lambda_{jl} \sim Beta(\delta_1, \delta_2)$, where δ_1, δ_2 are parameters, and can be updated with both the number of tokens assigned l in j and the method-of-moments estimates of the parameters.

3.3.3 *Document generation*

For the i-th ($i = 1$ to N_j) token in the j-th document,

- Draw topic level l_{ji}: $l_{ji} \sim \lambda_{jl} \prod_{h=0}^{l-1}(1 - \lambda_{jh})$
- Draw topic indicator z_{ji}: $\phi_{z_{ji}} \sim G_{T_j^l}$ if $l_{ji} = l$.
- Draw word w_{ji}: $w_{ji} \sim Discrete(G_k^{\mathbf{u}})$ if $h_{ji} = \mathbf{u}$ and $\mathbf{u}$ shares the topic with $z_{ji} = k$, else $w_{ji} \sim Discrete(G_{z_{ji}}^{\emptyset})$.

3.4 Conditional distribution of NOC

3.4.1 *Conditional distribution of the global topic tree*

We can illustrate the generative process of NOC using the CRP metaphor, as shown in Figure 1. We use $m_{l,p,c}$ to count the number of customers sitting at table serving dish $\phi_{l,p,c}$, see Fig 2. Likewise, $M_{l,p}$ is the number of customers that are descendants of $\phi_{\acute{l},p,c}$ ($l < \acute{l}$) including $\phi_{l+1,c,*}$ itself. After observing draws, the customer downs T, selects the table serving ϕ according to the following distributions using Eq (1):

$$P_g(\phi|\mathbf{S}) = \left(\begin{array}{l} \frac{m_{1,0,1}}{M+\alpha}, \underline{\phi \text{ is a background topic } \phi_{1,0,1}}, \\ \frac{m_{l,p,c}}{m_{l,p,\cdot}+\alpha}, \underline{\phi \text{ is an existing topic } \phi_{l,p,c}}, \\ \frac{\alpha}{m_{l,p,\cdot}+\alpha}, \underline{\phi \text{ is a new topic } \phi_{l,p,c}}, \\ \frac{M_{l,p}}{m_{l,p,\cdot}+\alpha}, \underline{\phi \text{ is an existing topic } \phi_{\acute{l},c,*}}, \end{array} \right. \qquad (6)$$

where $\mathbf{S}$ is the previous assignments in T, $m_{l,p,\cdot} = \sum_{c=1}^{C_p} m_{l,p,c}$, $M = \sum_{l=1}^{L} \sum_{p=1}^{P_l} m_{l,p,\cdot}$.

3.4.2 *Conditional distribution of document specific topic tree*

As illustrated in Figure 2, the i-th customer chooses an existing table with a probability proportional to the number of customers sitting at the table and shares the same topic in each restaurant j, or selects a new table with probability β_j and orders a dish (topic) from a global distribution G_T. This process continues until a full path is defined by descending tree T, which yields the construction of T_j.

Combining Eq (6) and λ_{jl}, we denote $m_{l,p,c}$, $(m_{l,p,\cdot})$ in T_j by $m^j_{l,p,c}$ $(m^j_{l,p,\cdot})$ as document j-specific quantities, and gain the conditional distribution of ϕ in T_j assigned to z_{ji}, $\phi_{z_{ji}}$:

$$P_j(z_{ji}=k|\text{previous assignments in } T_j)$$
$$=\begin{cases} \underline{\lambda_{j1}, \phi_k \text{ is background topic } \phi_{1,0,1},} \\ \lambda_{jl}\sum_{h=1}^{l-1}(1-\lambda_{jh})\left(\frac{m^j_{l,p,c}}{m^j_{l,p,\cdot}+\beta_j}+\frac{\beta_j}{m^j_{l,p,\cdot}+\beta_j}\frac{m_{l,p,c}}{m_{l,p,\cdot}+\alpha}\right), \\ \underline{\phi_k \text{ is existing topic } \phi_{l,p,c} \text{ on the } l\text{-th level } (l>1),} \\ \lambda_{jl}\sum_{h=1}^{l-1}(1-\lambda_{jh})\left(\frac{m^j_{l,p,c}}{m^j_{l,p,\cdot}+\beta_j}+\frac{\beta_j}{m^j_{l,p,\cdot}+\beta_j}\frac{\alpha}{m_{l,p,\cdot}+\alpha}\right), \\ \underline{\phi_k \text{ is new topic } \phi_{l,p,c} \text{ on the } l\text{-th level } (l>1).} \end{cases} \quad (7)$$

Document level $G_{T^l_j}$ inherits the topics from G_{T^l} on each topic level, but assigns to them document-specific weight $\lambda_{jl}\sum_{h=1}^{l-1}(1-\lambda_{jh})$ and allows various number of customers sitting at each table on each topic level.

3.4.3 *N-gram words over topics*

Since G_k can be adapted by the conjugate prior and integrated out analytically, the next word w drawn from G^u_k after $h_{ji}=u$ with $z_{ji}=k$ is obtained using Eq (3) as, $P^{\mathbf{u}}_k(w)$:

$$P^{\mathbf{u}}_k(w|\mathbf{S})=\frac{C^k_{uw\cdot}-d_{|\mathbf{u}|}t^k_{\mathbf{u}w}}{C^k_{\mathbf{u}\cdot\cdot}+\gamma_{|\mathbf{u}|}}+\frac{\gamma_{|\mathbf{u}|}+d_{|\mathbf{u}|}t^k_{u\cdot}}{C^k_{\mathbf{u}\cdot\cdot}+\gamma_{|\mathbf{u}|}}P^{\mathbf{u}\backslash}_k(w|S), \quad (8)$$

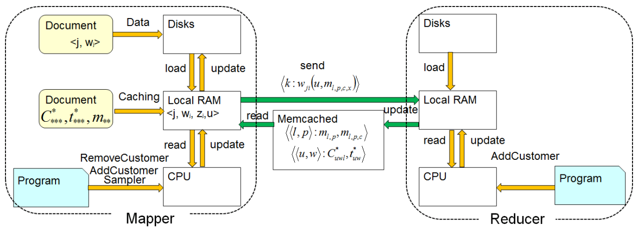
where $C^k_{\mathbf{u}wl}$ denotes the number of customers sitting at table l and eating dish (word) w after $\mathbf{u}$ in k, $C^k_{\mathbf{u}w\cdot}=\sum_l C^k_{\mathbf{u}wl}$, $C^k_{\mathbf{u}\cdot\cdot}=\sum_{\mathbf{w}} C^k_{\mathbf{u}w\cdot}$, $t^k_{\mathbf{u}w}$ is the number of tables serving w after u in k, and $t^k_{\mathbf{u}\cdot}=\sum_w t^k_{\mathbf{u}w}$, $\mathbf{S}$ represents the current seating arrangement. Note that multiple tables may offer the same dish, ϕ_k. In Eq (8), Hyperparameters $\gamma_{|\mathbf{u}|}$ (γ_0) and $d_{|\mathbf{u}|}$ (d_0) can be estimated by auxiliary variable sampling [21]. This distribution allows NOC to support 1-gram,2-gram,$\cdots$,N-gram simultaneously, and initiate N-gram by the change of topic instead of the a Boolean variable [12, 18].

4. INFERENCE IN NOC

4.1 Inference algorithm

Since multinomials and distributions can be marginalized out, we obtain the conditional distributions so that a Gibbs sampler can train NOC, as shown in 3.4. Details of the inference for NOC are shown in Algorithm (1), where we represent $C^k_{\emptyset wl}$, γ_0, d_0, $t^k_{\emptyset w}$, and $t^k_{\emptyset\cdot}$ with $C^k_{\mathbf{u}wl}$, $\gamma_{|\mathbf{u}|}$, $d_{|\mathbf{u}|}$, $t^k_{\mathbf{u}w}$, and $t^k_{\mathbf{u}\cdot}$ for simplification.

As in the previous work [27], the sampler iterates over all customers in each restaurant, and resamples the table where each customer sits using two routines: RemoveCustomer(line 16, Algorithm 2) and AddCustomer(line22, Algorithm 3). RemoveCustomer excludes a customer (topic) from the current seating and decrements the number of customers sitting there, and AddCustomer chooses a table for



Document <j, w_i>
Data
Disks
load
update
Document $C^*_{***}, t^*_{***}, m_{**}$
Caching
Local RAM <j, w_i, z_i, u>
read
update
RemoveCustomer
AddCustomer
Sampler
Program
CPU
Mapper
send
$\langle k : w_{ji}(u, m_{l,p,c,x})\rangle$
read
Memcached
$\langle\langle l,p\rangle : m_{l,p}, m_{l,p,c}\rangle$
$\langle\langle u,w\rangle : C^*_{uwl}, t^*_{uw}\rangle$
update
Disks
load
Local RAM
read
update
AddCustomer
CPU
Program
Reducer

Figure 3: System architecture of NOC with data and logical flow

the customer and increments the number of customers sitting there. This inference stores only the counts of how many tables there are in a restaurant and how many customers are sitting at each table in that restaurant, where the actual identity of the table at which a customer sits has no effect on the likelihood of the data. Since the topic assignment is described by using the metaphor of CRP, the token-based sampler used in this paper is almost identical to the collapsed sampler of the LDA.

At each iteration, the sampler decreases the topic assignment of the topic tree(line 14), and excludes the current customer eating w_{ji} to a table serving w_{ji} in the restaurant specified by the topic indicator k and previous word sequence u by using AddCustomer. Our sampling strategy for a given token i in document j is to propose the topic assignment z_{ji}, and the seating arrangement, $\mathbf{S}_{\backslash ji}$. Then, we need the conditional distribution of z_{ji} given all other topic assignments $\mathbf{z}_{\backslash ji}$, which is the set of seating arrangements with a customer corresponding to w_{ji} removed by using RemoveCustomer, as

$$P(z_{ji}=k|\mathbf{z}_{\backslash ji},\mathbf{S}_{\backslash ji})\propto P_j(z_{ji}=k|\mathbf{z}_{\backslash ji},\mathbf{S}_{\backslash ji})P^{\mathbf{u}}_k(w|\mathbf{S}_{\backslash ji}), \quad (9)$$

where $P_j(z_{ji}=k|\mathbf{z}_{\backslash ji},\mathbf{S}_{\backslash ji})$ is given by Eq (7), and $P^{\mathbf{u}}_k(w|\mathbf{S}_{\backslash ji})$ is given by Eq (8). After drawing a proposal for z_{ji}, the sampler adds a customer eating w_{ji} to a table serving w_{ji} in the restaurant specified by z_{ji} and u by using AddCustomer(line22), and updates the topic assignment of the topic tree(line27).

4.2 D-NOC: Distributed Algorithm of NOC

To scale NOC to web scale data sets, we employ a parallel algorithm that allows task distribution over P distinct processors like other approaches [22, 26], see Algorithm 4 and 5. This algorithm is composed of two steps: a Gibbs sampling step (Map) and a synchronization step (Reduce). This is based on the observation that the collapsed sampler for NOC can be decomposed into sparse terms (doc-proposal) and a dense term (topic/word-proposal). It allows our algorithm to distribute dense terms over nodes, and calculate sparse terms precisely in each node. Therefore, the key idea of D-NOC is that dense terms change their distributions slowly, and can be approximated efficiently by relaxing the requirement of sequential sampling when learning a model. This implies that dense terms can be distributed over nodes. Provided that the number of word tokens is larger than that of processors, our model can be built on the hypothesis that we can relax the requirement of sequential sampling when learning a model; this makes concurrent sampling closely approach sequential sampling [22, 26].

Algorithm 1 Inference for NOC

```
1: // initialization
2: zero all count variables, $m_{l,p,c}, m_{l,p,\cdot}, m^j_{l,p,c}, m^j_{l,p,\cdot}, C^*_*, t^*_*$
   and set $K$, and $\mathbf{U}_* = \emptyset$;
3: for $j = 1$ to $D$ do
4:   for $i = 1$ to $N_j$ do
5:     sample topic index $z_{ji} = k \sim Multinomial(1/K)$
6:     $m_{l,p,c}$ += 1; $m_{l,p,\cdot}$ += 1; $m^j_{l,p,c}$ += 1; $m^j_{l,p,\cdot}$ += 1
7:   end for
8: end for
9: // Gibbs sampling over burn-in and sampling period
10: for iteration=1 to $N_{iteration}$ do
11:   for $j = 1$ to $D$ do
12:     for $i = 1$ to $N_j$ do
13:       // for the current assignment of $z_{ji}$
14:       $m_{l,p,c}$ -= 1; $m_{l,p,\cdot}$ -= 1; $m^j_{l,p,c}$ -= 1; $m^j_{j,p,\cdot}$ -= 1
15:       //for PYP of phrases
16:       Call RemoveCustomer ($w_{ji}$,$\mathbf{u}$,$k$):
17:       // topic assignment
18:       draw $z_{ji}$ using equations (9).
19:       if $k$ is a new child of $T(T_j)$ then
20:         add a new child to $T(T_j)$
21:       end if
22:       Call AddCustomer ($w_{ji}$,$\mathbf{u}$,$k$); $\mathbf{u} = \mathbf{u} + w_{ji}$
23:       if $|\mathbf{u}| > 2$ then
24:         $\mathbf{U}_k = \mathbf{U}_k \cup u$
25:       end if
26:       // for the new assignment of $k$ in document $j$
27:       $m_{l,p,c}$ += 1; $m_{l,p,\cdot}$ += 1; $m^j_{l,p,c}$ += 1; $m^j_{l,p,\cdot}$ +=
          1
28:     end for
29:     update $\beta_j, \delta_{1,2}$
30:   end for
31:   update $\alpha$, $\gamma_{|\mathbf{u}|}$ and $d_{|\mathbf{u}|}$
32:   for $l = L - 1$ to 1 do
33:     for $p = 1$ to $P$ do
34:       // Merge topic phase
35:       Call MergeTopic ($l, p, m_{l,p,\cdot}$)
36:     end for
37:   end for
38: end for
```

In the sampling step, each processor samples its local topic assignments, $\mathbf{z}_x$, the topic assignments of documents stored in each processor x, by using both previous topic assignments $\mathbf{z}_x$, and the global counts and parameters. This step updates topic specific count using RemoveCustomer (Algorithm 2), and AddCustomer (Algorithm 3), again; note that these updates are performed in the synchronization step. While the topic merge phase that limits unnecessary growth in the number of topics [22] in Algorithm 1, D-NOC implements newly this phase consisting of Map and Reduce phase and then performs this pair after 10 iterations of Algorithm 4 and 5. This is why we call this algorithm an approximated Gibbs Sampling.

Figure 3 shows the architecture of our implemented prototype system, which could be applied to other topic models [16] and the other framework [31] in large scale machine learning. We implemented these algorithms on Hadoop[3], where each processor corresponds to a data node with a

Algorithm 2 Function: RemoveCustomer (w,$\mathbf{u}$,k)

```
if $|u|$=0 then
  decrement $C^k_{\emptyset w \cdot}$
else
  remove a customer from $l-th$ table from $\mathbf{u}$ with probabilities proportional to $C^k_{\mathbf{u}wl}$ and decrement $C^k_{\mathbf{u}wl}$
  if as a result the $l-th$ table becomes unoccupied then
    Call Remove customer ($w$,$\mathbf{u}_{\backslash}$,$k$)
  end if
end if
```

Algorithm 3 Function: AddCustomer (w,$\mathbf{u}$,k)

```
if $|\mathbf{u}|$=0 then
  increment $C^k_{\emptyset w \cdot}$
else
  sit a customer at the $l-th$ table in $\mathbf{u}$ with probability proportional to $max(0, C^k_{\mathbf{u}wl} - d_{|\mathbf{u}|})$ and increment $C^k_{\mathbf{u}wl}$
  sit a customer at a new table $l_{new}$ serving $w$ in $\mathbf{u}$ with probability proportional to $(\gamma_{|\mathbf{u}|} + d_{|\mathbf{u}|} t^k_{\mathbf{u}\cdot}) p^{\mathbf{u}\backslash}_k(w)$ using Eq (8) and increment $t^k_{\mathbf{u}w}$, and set $C^k_{\mathbf{u}wl_{new}} = 1$
  Call AddCustomer ($w$,$\mathbf{u}_{\backslash}$,$k$)
end if
```

task tracker, each sampling step is performed by a mapper, and each synchronization step is performed by a reducer. Additionally, we used Memcached [4], to store and broadcast $m_{l,p,c}, m_{l,p,\cdot}, C^k_{uwl}, t^k_{uw}$ across the Hadoop cluster. This leads all programs can access and read any parameters from any nodes in the cluster. Although we require Memcached to accelerate the I/O performance in the system build on Hadoop, we would not use it in the the system build on Spark.

In practice, the update process requires much time and memory, because this process consists of a series of computationally expensive task that merges local counts embedded in topic and the corresponding word hierarchies. In fact, we added these functions in the map phase, and verified that this introduction has minimal influence on the results. After processor x updates $\mathbf{z}_x$ by sweeping through its local data, it sends the modified count to the master node. Although $\mathbf{m}^j_{l,p,c,x}$ is the count of $\mathbf{m}^j_{l,p,c}$ stored in processor x and then is the local count, $\mathbf{m}_{l,p,c,x}$ is a local copy of $\mathbf{m}_{l,p,c}$ in x, that should be shared among all nodes and updated after every step.

In the synchronization step, the node aggregates the local counts from each processor to create a hierarchy, like building a suffix tree, and updates a single set of globally-consistent counts and related parameters. This process is repeated for either a fixed number of iterations or until the algorithm has converged.

5. EXPERIMENTS

5.1 Data sets

We focus here on the extraction of topics, its structure, and corresponding N-grams. The following data sets were used in comparative quantitative evaluations against previous topic models.

[3]Apache$^{\text{TM}}$Hadoop®: http://hadoop.apache.org/

[4]Memcached: http://memcached.org/

Algorithm 4 Map Phase for D-NOC

```
// Initialization
// Input
<Key,Value>:=<document ID $j$, document content $\mathbf{w}_j$>
// Approximated Gibbs Sampling
for iteration=1 to $N_{iteration}$ do
  for each processor $x$ in Parallel $P$ do
    Read from Memcached
    for $i = 1$ to $N_j$ for $j = 1$ to $D_x$ do
      $m_{l,p,c,x}$ -= 1; $m_{l,p,\cdot,x}$ -= 1; $m^j_{l,p,c}$ -= 1; $m^j_{l,p,\cdot}$ -= 1;
      draw $z_{ji}$ using equations (9).
      if $k$ is a new child of $T(T_j)$ then
        add a new child into $T(T_j)$
      end if
      $m_{l,p,c,x}$+=1; $m_{l,p,\cdot,x}$+=1; $m^j_{l,p,c}$+=1 ; $m^j_{l,p,\cdot}$+=1;
      if $|u| > 0$ then
        $\mathbf{u} = \mathbf{u} + w_{ji}$
      end if
      Send <Key,Value>:=<$k : w_{ji}(\mathbf{u}, m_{l,p,c,x})$> to the reducer
    end for
    update $\beta_j$ and $\delta_{1,2}$
  end for
end for
```

ACM papers: 8 years (2001-2008) of research papers in the proceedings of ACM CIKM, SIGIR, KDD, and WWW[5]. Preprocessing was applied to Data1. This yielded a total set of 3078 documents and 20286 unique words.

Amazon review data[6] (Amazon(s)): This is one of the biggest publicly available text data sets currently in text analysis research [13]. This data consists of 5,686,344 reviews and 3,784,413 unique words. We split strings using all wisp's characters and non-word characters as delimiters, removed numbers, and the words that appeared less than five times in the corpus, as a preprocessing step.

Amazon review data[7] (Amazon(l)): This is also the biggest public available corpus and is used in [19]. This data consists of 34,686,770 reviews. We conducted the same preprocessing step as done in Amazon(s).

Twitter data[8]: We selected tweets from 30/08/2012 to 29/01/2013, and gained 321,513,597 tweets.

We used the former two data sets that can be processed in a conventional single server, and the latter three data sets that require a scalable learning environment for parallel processing.

The goal of NOC is to help our understanding of a given corpus by representing the thematically and syntactically inherent generative process of a given document set. Therefore, we designed experiments to challenge NOC with the following questions:

- Do N-grams depend on the hierarchical topic structure? Can NOC represent a given corpus more efficiently than conventional topic models?: Subsection 5.2.1

[5]http://dl.acm.org/

[6]Amazon Product Review Data (Huge): http://www.cs.uic.edu/~liub/NBS/sentiment-analysis.html

[7]Amazon reviews: https://snap.stanford.edu/data/web-Amazon.html

[8]Twitter: http://twitter.com

Algorithm 5 Reduce Phase for D-NOC

```
// Initialization
$\mathbf{U} = \emptyset$, Clear all counts associated with $k$;
// Input
<Key,Value>
if Value = $w(\mathbf{u})$ then
  Sort $w, \mathbf{u}$ order by the length, alphabet
  $\mathbf{U} = \mathbf{U} \cup w(\mathbf{u})$
  $\mathbf{W} = \mathbf{U}$
  for each $w, \mathbf{u}$ in $\mathbf{W}$ do
    Let the latest word of $\mathbf{u}$ as $\mathbf{u}_l$, and the rest as $u_r$
    Call AddCustomer ($\mathbf{u}_l$,$\mathbf{u}_r$,$k$) (Algorithm 3):
  end for
else
  $m_{l,p,c} \leftarrow m_{l,p,c} + \sum_x (m_{l,p,c} - m_{l,p,c,x})$
end if
$m_{l,p,\cdot} = \sum_c m_{l,p,c}$
update $\alpha$, $\gamma_{|\mathbf{u}|}$ and $d_{|\mathbf{u}|}$
Broadcast $\alpha, \gamma_{|\mathbf{u}|}, d_{|\mathbf{u}|}, m_{l,p,c}, m_{l,p,\cdot}, C^*_{\mathbf{u}wl}, t^*_{\mathbf{u}w}$ via Memcached
```

- Can NOC extract the thematic coherence structure and complement human experts that lead to a practical application?: Subsection 5.2.2

- Can D-NOC be applied to a large data, and be an approximated algorithm of NOC?: Subsection 5.2.3

- Can NOC capture the semantic structure between topics and the syntactic?: Subsection 5.3

We ran the experiments on 1, 20, 30, and 40 PCs with Dual Core 2.66 GHz Xeon processors and the number of Gibbs sampling iterations was set to 2000, where each sampler took the first 100 iterations to burn in. Here, the estimation or usage of hyper parameters with counterparts followed the previous setting [12, 18].

5.2 Quantitative Evaluation

5.2.1 Test-set Perplexity

The purpose of this experiment is to show how well NOC maintains the document's structure, and how well NOC represents a given document corpus. To evaluate the ability of generative models, we numerically compared the models by computing test-set perplexity (PPX). Perplexity is a standard measure used in the language modeling community to assess the predictive power of a model, is algebraically equivalent to the inverse of the geometric mean per-word likelihood. A lower score implies that word w_{ji} is less surprising to the model and are better.

We computed perplexity as follows. First, we randomly took 10% words w^{test}_j from each text (review, paper, tweet text) to create a test part; the remainder w^{train}_j was used as the learning part. For every text, the test part was held out to compute perplexity. Second, the learning part was used for estimating the parameters by Gibbs sampling. Finally, a single set of topic counts was saved when a sample was taken; the log probability of test products that had never been seen before was computed in the same way as the perplexity computation of previous works [15]. The perplexity of w^{test}_{ji} was computed for all algorithms using 100 samples from 2000

Table 4: Benchmark of N-gram topic models

	N-gram	topic structure	Power-law
HDP-LDA [28]	×	flat	×
NTSeg [12]	○	flat	×
PDLDA [18]	○	flat	○
NOC(NOC_d)	○(○)	hierarchy	○(×)

different chains, after the burn in period, using the following standard practice of averaging over multiple chains;

$$PPX = \exp(-\frac{1}{N_{W^t}} \sum_{j \in D_{\text{test}}}^{|D_{\text{test}}|} \sum_{w_{ji}^{test} \in d_j}^{|N_j|} \log \frac{\sum_{r=1}^{R} p_r(w_{ji}^{test}|w_j^{train}))}{R}), \tag{10}$$

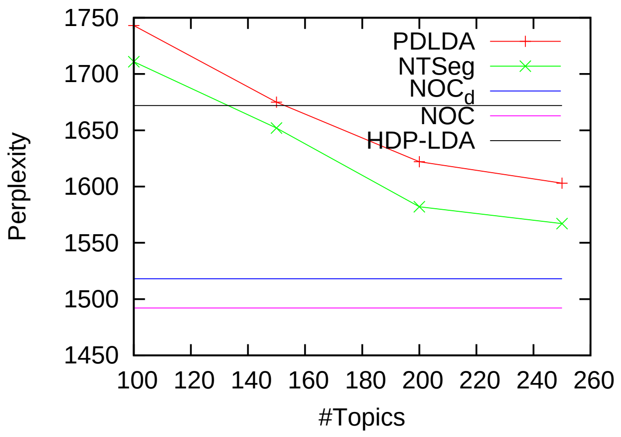
where N_{W^t} is the number of test words, R is the number of samples (from R different chains), and the predictive probability of models is given by equations (7), (8). To discuss the effect of topic order in each document, and PYP in N-gram, we prepared NOC_d, that differs NOC in using DP instead of PYP to form N-gram. We applied the models to the data sets, where all models had the constraint that no bigram is allowed for stop words and sentence/paragraph boundaries. Table 4 compares the characters of N-gram topic models used in this experiment.

As shown in Figure 4, the perplexity comparisons, PDLDA, and NTSeg were trained using various numbers of topics, while NOC and NOC_d define the number of topics. The number of topics of $HDP-LDA$ (NOC_d, NOC) is 267 (253, 241) for ACM, while that of $HDP-LDA$ (NOC_d, NOC) is 292 (271, 267) for Amazon(s). The outcome, that NOC offers lower perplexity than NOC_d, supports our idea that 1) PYP reduces perplexity, 2) N-grams depends on the topic structure, and 3) improve the explanation capability of a given corpus. Although Amazon data consists mainly of short texts that have only a few words and sentences, our model best demonstrated its superiority on this data set.

5.2.2 *Evaluation of representative N-gram*

We evaluate the ability to extract representative phrases that are comparable to human labeled phrases from the given data set. In order to generate a gold standard for these phrases, we utilized both rating scores (Amazon), ACM categories[9] of papers (ACM) and human annotation. For each score and the category, we aggregated candidate bigram phrases and then provided 5 users with the most frequent 100 bigrams/trigrams on the most popular items in DVD, Music, and Book (Amazon). These users were asked to select 10 phrases from them based on their clarity in terms of positive (v=4 or 5)/negative (v=1 or 2) in Amazon, I.2 ARTIFICIAL INTELLIGENCE /I.7 DOCUMENT AND TEXT PROCESSING in ACM. After that, we judged the top 20 phrases with the highest number of users to be the gold standard set of positive/negative (Artificial/Document) labels. This setting allowed us to treat each human generated gold standard phrase as "query", q, for each label, and documents returned in response to each query by each model as "retrieved documents", d, where documents that coincide with the labels of query are "relevant documents". NOC (NTSeg, and PDLDA) presented top documents with the highest probability of generating the given query. In the query likelihood model, the score of each document, relative to query q for models can be computed as the probability of q given d [30]. We compared these models using Precision and Recall of the returned documents whose probability exceeded 0.5.

[9]http://www.acm.org/about/class/ccs98-html



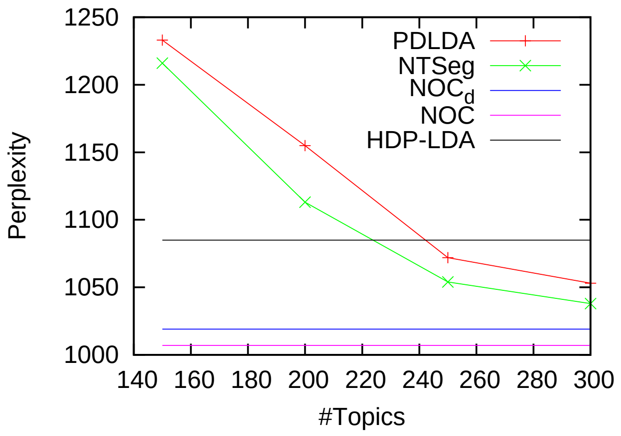


Figure 4: Test-set perplexity on (upper) ACM, and (lower) Amazon(s) on versus number of the topics

Table 5 shows that NOC provides the best results over the various length, N. That is, NOC can complement human domain experts and existing knowledge, and so offers rating/category-based document retrieval, even for N-grams.

5.2.3 *Scalability of NOC*

The purpose of this experiment is to investigate how our distributed NOC (D-NOC) algorithm performs relative to the basic NOC algorithm designed for only a single node. In all experiments, we set the number of mappers/reducers to 2/1, and the memory limit for every mapper and reducer instance to 2.0 GB on each processor. For running D-NOC, the details of experiment parameters, and perplexity computation are the same as stated in subsection 5.2.1.

We performed experiments to see whether the distributed algorithm of D-NOC could scale over various numbers of processors, P, and topics, while well approximating the performance of NOC (P=1) on these settings. To systematically evaluate these algorithms, we varied the number of

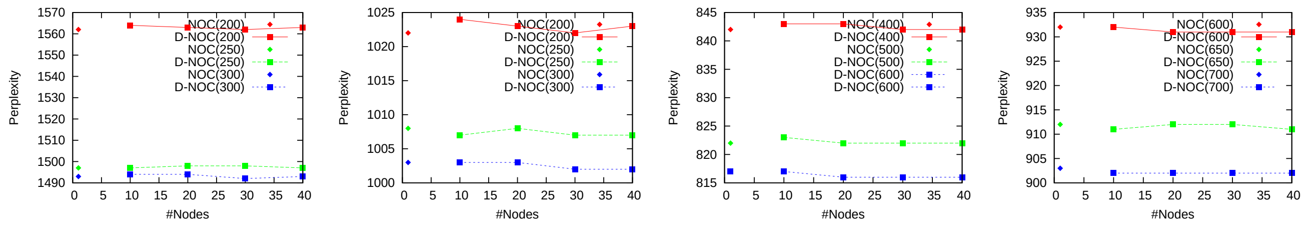


Figure 5: Convergence of PPX on (left) ACM, (center left) Amazon(s), (center right) Twitter, (right) Amazon(l) versus number of processors (nodes): The number of parenthesis is the number of topics.

Table 6: Word distribution learned from the Twitter data set by NTSeg, and NOC: We list top 10 words with high probability (bigram words) from NTSeg, and NOC.

NOC				NTSeg	
$G_k(w)$	$G^u_k(w)$	$G_{k'}(w)$	$G^u_{k'}(w)$	$P_{k''}(w)$	$P^u_{k''}(w)$
iphone, app, ipad, apple, 4s, apps, mac, ipod, mobile, download	iphone 4s, iphone user, iphone ipad, iphone android, iphone app, iphone apps, google play, appstore, careerjet maior, mobile app	developer, social, web, instagram, photographer, addict, lover, fan, proud,geek	social media, apple fan, apple fanboy, graphic designer, apple lover, husband father, apple enthusiast, web developer, distinguished educator, tech geek	iphone, user, de, android, apps, clique, vagas, team, music, instagram	iphone app, social media, ipod touch, iphone apps, apple lover, google play, iphone android, android app, apple fan, mobile app

Table 5: Evaluation of Representative Phrases Comparison of PDLDA, NTSeg and NOC: PDLDA, and NTSeg were trained using the number of topics Z that yielded the best performance in Figure 4. Results that differ significantly, t-test $p < 0.01$, from NTSeg are marked with ''.**

Data	Amazon(s)		ACM	
	bigram (N=2)			
Model	Precision	Recall	Precision	Recall
PDLDA	0.67	0.60	0.55	0.58
NTSeg	0.72	0.65	0.62	0.62
NOC	0.75	0.71^*	0.71^*	0.75^{**}
	trigram (N=3)			
PDLDA	0.46	0.42	0.38	0.33
NTSeg	0.57	0.55	0.47	0.44
NOC	0.65^*	0.66^{**}	0.61^{**}	0.67^{**}

nodes P and topics, and measured their performance using PPX; the results are shown in Figure 5.

This figure shows that PPX of D-NOC nearly matches that of NOC on the same number of topics regardless of processor number, and this tendency is observed for various topics. We can see that D-NOC offers the same performance as NOC on a single processor, and so is truly scalable in terms of both speed and data size. These results prove that D-NOC can scale to large data, since 1)the convergence rate for D-NOC matched that of NOC, 2)D-NOC attains similar PPX regardless of the number of P with the same number of topics, and 3)the amount of processable data scales with the number of data nodes on Hadoop. Although the overhead requires about 50-60% of the runtime in each sampling iteration, each iteration requires about 2 minutes regardless of the number of nodes, and the computation time decreases proportionally to the number of nodes. Consequently, D-NOC can process large data sets while preserving full generative model performance, and offers the approximated Gibbs Sampling of NOC.

5.3 Qualitative Evaluation

Table 6 provides an example of words of topics learned by NTSeg, and NOC. For the selected topic k learned by NOC, we picked the child topic of this topic, k', and picked the most similar topic k'' from topic k learned by NOC, where we measure similarity by KL divergence over words to select the topic having the most similar word distribution with k. We then chose one topic and the N-gram topic identified by these picked topics k, k', and k''. For each topic, we list the top 10 words in decreasing order of topic-specific probability, where $G_k(w)$ ($G^u_k(w)$) is 1-gram (2-gram) word distribution on k learned by NOC, and $p_k(w)$ ($p^u_k(w)$) is 1-gram (2-gram) word distribution on k learned by NTSeg.

In NOC, the parent topic consisted of concatenated words associated with "apple products", and this child topic consisted of concatenated words associated with "fan of these products". In NTSeg, these words and phrases are assigned to the same topic, as NTSeg and the other N-gram topic models neglect the hierarchical semantic relationships between topics. The other topic yields similar phenomena as well. This supports the results shown in subsection 5.2.2 such that NOC can construct compact and semantic topic structures that provide more syntactic information than the other models.

6. DISCUSSION

The disadvantages of parallelized topic models can be partially solved as follows: The difference between processing time in each server can be alleviated by reducing the block size in Hadoop. The time required for reading global counts and parameters can be shortened by using Memcached instead of HDFS in Hadoop. Additionally, Memcached keeps the network load at $\mathcal{O}(1)$ per each server, and restrains the memory required to store a given amount of topic-word/word counts $\mathcal{O}(P^{-1})$ over P servers. We showed that Hadoop allows us to implement scalable topic models, and that their approximated algorithms work well like previous algorithms for single node implementation. Consequently, the shortcomings inherent in Gibbs sampling [32] could be alleviated by 1)proposed approximated algorithm, and 2)usage of Memcached. Because the existing topical N-gram models seem to be special cases of NOC, this algorithm has the flexibility to accommodate these models, and can be applied to them.

As the average length of topic specific N-gram is 2.5 and the number of unique topics included in each document is 5.7 over Amazon(l), we can image that each document has relationships between topics and they show the semantic relationship. As shown in subsection 5.2.1, and 5.3, modeling both the topic hierarchy and N-grams helps each other to learn, and thereby yields the more compact topic structure with higher purity topic specific N-grams than the others.

7. CONCLUSION

This paper shows a N-gram topic model that employs a semantic topic hierarchical structure as the thematic coherence, and forms topic specific N-grams that can provide the syntactic. Our contribution lies in capturing these linguistic structures and showing its algorithm can be applied to a web scale data. Future work is to extend NOC to be applicable with streaming data.

8. REFERENCES

[1] A. Ahmed, L. Hong, and A. J. Smola. Hierarchical geographical modeling of user locations from social media post. In *WWW*, pages 25–36, 2013.

[2] D. Aldous. *Exchangeability and related topics*, volume 1117 of *Lecture Notes in Math.* Springer, Berlin, 1985.

[3] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[4] D. Blei, T.Griffiths, and M. Jordan. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *J. ACM*, 57(2):7:1–7:30, 2010.

[5] D. Blei, T.Griffiths, M. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. *NIPS*, 16, 2004.

[6] J. Boyd-Graber and D. M. Blei. Syntactic topic models. In *NIPS*, pages 185–192, 2008.

[7] T. S. Ferguson. A bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, Mar 1973.

[8] T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. Integrating topics and syntax. In *NIPS*, pages 537–544, 2004.

[9] http://parameterserver.org/.

[10] http://petuum.github.io/.

[11] http://spark.apache.org/.

[12] S. Jameel and W. Lam. An unsupervised topic segmentation model incorporating word order. In *SIGIR*, pages 203–312, 2013.

[13] N. Jindal and B. Liu. Opinion spam and analysis. In *WSDM*, pages 219–230, 2008.

[14] N. Kawamae. Identifying sentiments over n-gram. In *WWW*, pages 541–542, 2012.

[15] N. Kawamae. Supervised N-gram topic model. In *WSDM*, pages 473–482, 2014.

[16] N. Kawamae. Real time recommendations from connoisseurs. In *KDD*, pages 537–546, 2015.

[17] J. H. Kim, D. Kim, S. Kim, and A. Oh. Modeling topic hierarchies with the recursive chinese restaurant process. In *CIKM*, pages 783–792, 2012.

[18] R. V. Lindsey, W. P. Headden, III, and M. J. Stipicevic. A phrase-discovering topic model using hierarchical pitman-yor processes. In *EMNLP-CoNLL*, pages 214–222, 2012.

[19] J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*, pages 165–172, 2013.

[20] D. Mimno and A. McCallum. Organizing the oca: learning faceted subjects from a library of digital books. In *Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*, pages 376–385, 2007.

[21] R. M. Neal. Markov chain sampling methods for dirichlet process mixture models. *JCGS*, 9(2):249–265, 2000.

[22] D. Newman, A. Asuncion, P. Smyth, and M. Welling. Distributed algorithms for topic models. *JMLR*, 10:1801–1828, dec 2009.

[23] J. Paisley, C. Wang, D. M. Blei, and M. I. Jordan. Nested hierarchical dirichlet processes. *TPAMI*, 2014.

[24] J. Pitman and M. Yor. The two-parameter poisson-dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25(2):855–900, 1997.

[25] J. Sethuraman. A constructive definition of dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

[26] A. Smola and S. Narayanamurthy. An architecture for parallel topic models. *Proc. VLDB Endow*, 3(1-2):703–710, 2010.

[27] Y. W. Teh. A hierarchical bayesian language model based on pitman-yor processes. In *Coling/ACL*, pages 985–992, 2006.

[28] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical dirichlet processes. *JASA*, 101(476):1566–1581, 2006.

[29] H. Wallach. Topic modeling: Beyond bag-of-words. In *ICML*, pages 977–984, 2006.

[30] X. Wei and W. B. Croft. Lda-based document models for ad-hoc retrieval. In *SIGIR*, pages 178 – 185, 2006.

[31] J. Yuan, F. Gao, Q. Ho, W. Dai, J. Wei, X. Zheng, E. P. Xing, T.-Y. Liu, and W.-Y. Ma. Lightlda: Big topic models on modest computer clusters. In *WWW*, pages 1351–1361, 2015.

[32] K. Zhai, J. Boyd-Graber, N. Asadi, and M. L. Alkhouja. Mr. lda: a flexible large scale topic modeling package using variational inference in mapreduce. In *WWW*, pages 879–888, 2012.

[33] G. Zipf. *Selective Studies and the Principle of Relative Frequency in Language.* Harvard University Press, Cambridge, MA, 1932.