Non-Bayesian Additive Regularization for Multimodal Topic Modeling of Large Collections

Konstantin Vorontsov
Yandex, Moscow Institute of Physics and Technology
voron@yandex-team.ru

Oleksandr Frei
Schlumberger Information Solutions
oleksandr.frei@gmail.com

Peter Romov
Yandex
romovpa@yandex-team.ru

Murat Apishev
Moscow State University
great-mel@yandex.ru

Marina Suvorova
Moscow State University
m.dudarenko@gmail.com

Anastasia Yanina
Moscow Institute of Physics and Technology
yanina.anastasia.mipt@gmail.com

ABSTRACT
Probabilistic topic modeling of text collections is a powerful tool for statistical text analysis based on the preferential use of graphical models and Bayesian learning. Additive regularization for topic modeling (ARTM) is a recent semi-probabilistic approach, which provides a simpler inference for many models previously studied only in the Bayesian settings. ARTM reduces barriers to entry into topic modeling research field and facilitates combination of topic models. In this paper we develop the multimodal extension of ARTM approach and implement it in BigARTM open source project for online parallelized topic modeling. We demonstrate the ability of non-Bayesian regularization to combine modalities, languages and multiple criteria to find sparse, diverse, and interpretable topics.

Categories and Subject Descriptors
H.3.3 [Information Search and Retrieval]: Clustering;
I.2.7 [Natural Language Processing]: Language models;
G.1.6 [Optimization]: Constrained optimization

General Terms
Theory, Algorithms, Experimentation

Keywords

1. INTRODUCTION
Topic modeling is a rapidly developing branch of statistical text analysis [2]. Topic model reveals a hidden thematic structure of a text collection and finds a compressed representation of each document in terms of its topics. Practical applications of topic models include information retrieval, classification, categorization, summarization and segmentation of texts. Topic models are increasingly used for non-textual and heterogeneous data including signals, images, video and networks. More ideas, models and applications are outlined in the survey [7].

From a statistical perspective, a probabilistic topic model (PTM) defines each topic by a multinomial distribution over words, and then describes each document with a multinomial distribution over topics.

Modern literature on topic modeling offers hundreds of models adapted to different situations [7]. Nevertheless, most of these models are too difficult for practitioners to quickly understand, adapt and embed into applications. This leads to a common practice of tasting only the very basic models such as Probabilistic Latent Semantic Analysis, PLSA [12] and Latent Dirichlet Allocation, LDA [4]. Most practical inconveniences are rooted in Bayesian learning, which is the dominating approach in topic modeling.

Bayesian learning is a very powerful and general theoretical framework, and topic modeling is just one of its applications. Bayesian inference is elegant when used with conjugate priors. However, from the linguistic point of view the Dirichlet conjugate prior is not necessary the best choice as it conflicts with natural assumptions of sparsity. Better motivated non-conjugate priors require a laborious mathematical work and lead to intricate learning algorithms. The development of combined and multi-objective topic models also remains a challenging task in the Bayesian approach. An evolutionary approach to multi-objective Bayesian topic modeling has been proposed in [14], but it seems to be computationally infeasible for large text collections. Until now, there was no freely available software to combine topic models.

From an optimization perspective, topic modeling can be considered as a special case of approximate stochastic matrix factorization. To learn a factorized representation of a text collection is an ill-posed problem, which has an infinite set of solutions. A typical regularization approach in this case is to impose problem-specific constraints in a form of additive terms in the optimization criterion.

Additive Regularization for Topic Modeling (ARTM) is a semi-probabilistic approach based on classical (non-Bayesian) regularization [31]. In ARTM a topic model is
learned by maximizing a weighted sum of the log-likelihood and additional regularization criteria. These criteria are not required to be log-priors or even to have a probabilistic sense. The optimization problem is solved by a general regularized expectation-maximization (EM) algorithm, which can be easily applied to any combination of regularization criteria. The non-Bayesian regularization provides a much simpler inference for many topic models previously studied only in the Bayesian setting [33, 32]. In particular, the LDA model can be alternatively understood as a smoothing regularizer that minimizes Kullback–Leibler (KL) divergence of each topic distribution with a fixed multinomial distribution. The maximization of the KL-divergence naturally leads to sparsing [33]. This possibility is difficult to see from the Bayesian perspective, thereby all Bayesian approaches to sparsing are much more complicated [26, 35, 16, 10, 5].

ARTM makes topic models easier to design, to explain, to infer, and to combine, naturally reducing barriers to entry into topic modeling research field.

In this paper we develop the multimodal extension of ARTM approach and incorporate its parallel online implementation into BigARTM open source projet.

Multimodal data has become increasingly important in many application areas. Large data collections coming from the web or sensor networks consist of heterogeneous linked data. Typically, texts are accompanied by images, audio or video clips, usage data, metadata containing authors, links, date-time stamps, etc. In these cases documents are considered as multimodal containers, words being the elements of one of the modalities. All modalities are useful for determining more relevant topics, and, vice-versa, topics are useful for crossmodal retrieval, making recommendations for users or making predictions when data of some modalities are missing. We introduce the multimodal additively regularized topic model with an arbitrary number of modalities and generalize the regularized EM-algorithm for this case.

Online algorithms have proven to be very efficient for large document collections, including those arriving in a stream. Online algorithms are now available for PLSA [1], LDA variational inference [11], LDA stochastic inference [18], and some other topic models. We show that the online algorithm is not necessarily associated with a particular type of model, nor a particular type of inference, but only with a certain reorganization of steps in the EM-like iterative process. Our online algorithm remains the same for PLSA and LDA models, as well as for any combination of regularizers and any number of modalities.

The rest of the paper is organized as follows. In section 2 we introduce notation and definitions of topic modeling and ARTM. In section 3 we introduce a multimodal topic modeling for documents with additional discrete metadata. In section 4 we generalize online EM-algorithm from [11] for multimodal ARTM and discuss some details of its parallel implementation in BigARTM library. In section 5 we report results of our experiments on large datasets.

2. ARTM: ADDITIVE REGULARIZATION FOR TOPIC MODELING

Let $D$ denote a finite set (collection) of texts and $W$ denote a finite set (vocabulary) of all terms from these texts. Each term can represent a single word or a key phrase. Each document $d \in D$ is a sequence of terms from the vocabulary $W$. Assume that each term occurrence in each document refers to some latent topic from a finite set of topics $T$. Text collection is considered to be a sample of triples $(w_i, d_i, t_i)$, $i = 1, \ldots, n$, drawn independently from a discrete distribution $p(w, d, t)$ over the finite probability space $W \times D \times T$. Terms $w_i$ and documents $d_i$ are observable variables, while topics $t_i$ are latent variables.

The topic model of Probabilistic Latent Semantic Analysis, PLSA [12] explains the terms probabilities $p(w|d)$ in each document $d \in D$ by a mixture of term probabilities for topics and topic probabilities for documents:

$$p(w|d) = \sum_{t \in T} p(t|d) p(w|t) = \sum_{t \in T} \phi_{wt} \theta_{td}, \; w \in W.$$  

This representation follows immediately from the law of total probability and the assumption of conditional independence $p(w|t) = p(w|d, t)$, which means that each topic generates terms regardless of the document.

The parameters $\theta_{td} = p(t|d)$ and $\phi_{wt} = p(w|t)$ form matrices $\Theta = (\theta_{td})_{T \times D}$ and $\Phi = (\phi_{wt})_{W \times T}$. These matrices are stochastic, that is, their vector-columns represent discrete distributions. The number of topics $|T|$ is usually much smaller than $|D|$ and $|W|$.

To learn parameters $\Phi$, $\Theta$ from the collection we maximize the log-likelihood:

$$L(\Phi, \Theta) = \sum_{d \in D} \sum_{w \in W} n_{dw} \ln p(w|d) \to \max_{\Phi, \Theta},$$  

where $n_{dw}$ is the number of occurrences of the term $w$ in the document $d$.

Following the ARTM approach, we introduce $r$ additional criteria $R_i(\Phi, \Theta)$, $i = 1, \ldots, r$, called regularizers. We would like to maximize them separately, but the maximization of their linear combination with nonnegative regularization coefficients $\rho_i$ is technically more convenient:

$$R(\Phi, \Theta) = \sum_{i=1}^{r} \rho_i R_i(\Phi, \Theta) \to \max_{\Phi, \Theta}.$$

Then we add a regularization term $R(\Phi, \Theta)$ to the log-likelihood and solve a constrained multicriteria optimization problem via scalarization:

$$L(\Phi, \Theta) + R(\Phi, \Theta) \to \max_{\Phi, \Theta}.$$  

It follows from Karush-Kuhn-Tucker conditions that the local maximum $(\Phi, \Theta)$ of the problem (1), (2) satisfies the following system of equations with auxiliary variables interpreted as conditional probabilities $p_{tdw} = p(t|d, w)$ [33]:

$$p_{tdw} = \text{norm}(\phi_{wt} \theta_{td});$$

$$n_{wt} = \sum_{d \in D} n_{dw} p_{tdw};$$

$$n_{td} = \sum_{w \in W} n_{dw} p_{tdw};$$

$$\phi_{wt} = \text{norm}_{w \in W}(n_{wt} + \phi_{wt} \frac{\partial R}{\partial \phi_{wt}});$$

$$\theta_{td} = \text{norm}_{t \in T}(n_{td} + \theta_{td} \frac{\partial R}{\partial \theta_{td}}).$$
where the “norm” operator transforms a real vector \( (x_t)_{t \in T} \) to a vector \( (\tilde{x}_t)_{t \in T} \) representing a discrete distribution:

\[
\tilde{x}_t = \text{norm} x_t = \max\{x_t, 0\} / \sum_{t \in T} \max\{x_t, 0\}
\]

The system (3)–(5) can be solved by various numerical methods. In particular, the simple-iteration method is a popular choice due to its simplicity. It has been proven in [32] that it is equivalent to the EM-algorithm for PLSA and LDA topic models.

Many Bayesian topic models can be considered as special cases of ARTM with different regularizers [33, 32]. For example, PLSA [12] corresponds to the absence of regularization. \( R = 0 \). LDA [4] corresponds to the smoothing regularizer, which minimizes the KL-divergence \( KL(\alpha\|\beta_t) \) and \( KL(\beta_t\|\phi_t) \) for fixed distributions \( \beta, \alpha \). Choosing uniform distributions for \( \beta \) and \( \alpha \) corresponds to symmetric Dirichlet priors in Bayesian approach.

Additive regularization lets users build topic models for various applications simply by choosing a suitable combination of predefined regularizers from an extendable library.

For example, in [32] as many as five regularizers are combined together to improve interpretability of the model. The key idea is to split the set of topics \( T \) into two subsets: \( T = S \cup B \). and to configure regularizers in such a way that domain-specific terms go into the set \( S \), while common used words land in the set \( B \). Sparsity of domain topics \( t \in S \) is promoted by two regularization parameters that maximize the KL-divergence \( KL(\alpha\|\beta_t) \) and \( KL(\beta_t\|\phi_t) \). Smoothness of background topics \( t \in B \) is promoted by minimizing KL-divergences \( KL(\alpha\|\beta_t) \) and \( KL(\beta_t\|\phi_t) \). Finally, a covariance regularizer is used to decrease the correlation between columns in the \( \Phi \) matrix, thus promoting the diversity of the topics. The final combination of regularizers is as follows:

\[
R(\Phi, \Theta) = -\beta_0 \sum_{t \in S} \sum_{w \in W} \beta_w \ln \phi_{wt} - \alpha_0 \sum_{d \in D} \sum_{t \in S} \alpha_t \ln \theta_{td} + \\
+ \beta_1 \sum_{t \in B} \sum_{w \in W} \beta_w \ln \phi_{wt} + \alpha_1 \sum_{d \in D} \sum_{t \in B} \alpha_t \ln \theta_{td} - \\
- \gamma \sum_{t \in T} \sum_{\tau \in T} \sum_{w \in W} \phi_{wt} \phi_{wt},
\]

where \( \beta_0, \alpha_0, \beta_1, \alpha_1, \gamma \) are regularization coefficients.

This combination was extended in [33] by a new regularizer that maximizes the KL-divergence \( KL(p(t)\|q(t)) \), leading to a topic selection. Starting from an excessively high number of topics the regularizer eliminates insignificant, duplicated, and linearly dependent topics [34]. Compared to Hierarchical Dirichlet Process [29], the new regularizer results in a better topic selection algorithm: it gives a more stable number of topics, takes less time to execute, and has an ability to combine it with other topic models via additive regularization.

An important subject for ARTM models is optimization of the regularization coefficients \( \rho_t \). According to Tikhonov’s theory of ill-posed inverse problems [30], the regularization coefficients must tend to zero with the number of iteration. In practice, the regularization path is selected by adaptive tuning of regularization coefficients [32, 33, 34]. This empirical technique is based on visual control of multiple intrinsic and extrinsic performance measures on each iteration.

3. MULTIMODAL ARTM

Now assume that a document can contain not only words, but also terms of other modalities. Each modality is defined by a finite set (vocabulary) of terms \( W_m, \ m = 1, \ldots, M \). The sets \( W_m \) are disjoint.

Examples of not-words modalities are: authors, class or category labels, date-time stamps, references to/from other documents/authors, named entities, objects found in the images associated with the documents, users that read or downloaded documents, advertising banners, etc.

As in the previous section, the collection is considered to be a sample of i.i.d. triples \( (w, d, t) \sim p(w, d, t) \) drawn from the finite probability space \( W \times D \times T \), but now \( W = W^1 \cup \cdots \cup W^M \) is a disjoint union of the vocabularies across all modalities.

Following the idea of Correspondence LDA [3] and Dependency LDA [25] we introduce a topic model \( p(w \| d) \) for each modality \( W_m, \ m = 1, \ldots, M \):

\[
p(w \| d) = \sum_{t \in T} p(w \| t) p(t \| d) = \sum_{t \in T} \phi_{wt} \theta_{td}, \ w \in W^m.
\]

Stochastic matrices \( \Phi^m = (\phi_{wt})_{W^m \times T} \) of term probabilities for the topics, if stacked vertically, form a \( W \times T \)-matrix \( \Phi \).

To learn parameters \( \Phi^m, \Theta \) from the multimodal collection we maximize the log-likelihood for each \( m \)-th modality:

\[
\mathcal{L}_m(\Phi^m, \Theta) = \sum_{d \in D} \sum_{w \in W^m} n_{dw} \ln p(w \| d) \rightarrow \max_{\Phi^m, \Theta},
\]

where \( n_{dw} \) is the number of occurrences of the term \( w \in W^m \) in the document \( d \). Note that topic distributions of documents \( \Theta \) are common for all modalities.

In ARTM we add a weighted sum of regularization criteria \( R(\Phi, \Theta) \) to the log-likelihood and solve a constrained multicriteria optimization problem:

\[
\sum_{m=1}^{M} \sum_{w \in W^m} \tau_m \mathcal{L}_m(\Phi^m, \Theta) + R(\Phi, \Theta) \rightarrow \max; \Phi^m, \Theta
\]

where regularization coefficients \( \tau_m \) are used to balance the importance of different modalities. The local maximum \((\Phi, \Theta)\) of the problem (6), (7) satisfies the following system of equations with auxiliary variables \( p_{td} \)

\[
p_{tdw} = \text{norm}(\phi_{wt} \theta_{td});
\]

\[
n_{dw} = \sum_{d \in D} t_m(w) n_{dw} p_{tdw};
\]

\[
n_{td} = \sum_{w \in W^m} t_m(w) n_{dw} p_{tdw};
\]

\[
\phi_{wt} = \text{norm}_{w \in W^m}(n_{wt} + \phi_{wt} \frac{\partial R}{\partial \phi_{wt}});
\]

\[
\theta_{td} = \text{norm}_{t \in T}(n_{td} + \theta_{td} \frac{\partial R}{\partial \theta_{td}}),
\]

where \( m(w) \) is the modality of the term \( w, w \in W^m(w) \).

The system of equations (8)–(10) follows from Karush–Kuhn–Tucker conditions (see Appendix A for the proof).

For single modality (\( M = 1 \)) it gives the regularized EM-algorithm described in the previous section.
Many previous topic models for labeled documents can be considered as special cases of multimodal ARTM. Most of them are based on LDA model and use Dirichlet priors, which correspond to smoothing regularization. From ARTM perspective, there is little reason to always use only the smoothing regularizer.

The following topic models exactly correspond to the multimodal ARTM, up to the modality sense. A topic model of document content and hypertext connectivity [6] introduces a modality to represent hyperlinks between documents. The Conditionally Independent LDA, CI-LDA [21] has the modality of named entities mentioned in a given document. The Tag-LDA [27] has the modality of tags as a special kind of words. The LDA-JS and LDA-post [9] has publications cited in a given document; an special kind of words. The dependency LDA [25] has the modality of publications cited in a given document. The Conditional Independent LDA, CI-LDA [21], both models are likely to share similar topics. Both models are based on LDA model and use Dirichlet priors, which correspond to smoothing regularization. From ARTM perspective, there is little reason to always use only the smoothing regularizer.

4. ONLINE PARALLEL EM-ALGORITHM

Like Online LDA [11] and Online PLSA [1] we split the collection $D$ into batches $D_b$, $b = 1, \ldots, B$, and organize EM iterations so that each document vector $\theta_d$ is updated until convergence at a constant matrix $\Phi$, see Algorithm 1 and 2. The matrix $\Phi$ is updated rarely, after all documents from the batch are processed. For a large collection the matrix $\Phi$ often stabilizes after small initial part of the collection is processed. Therefore a single pass through the collection might be sufficient to learn a topic model. The second pass may be needed for the initial part of the collection.

The online reorganization of the EM iterations is not necessarily associated with Bayesian inference used in [11]. Different topic models, from PLSA to multimodal and regularized models, can be learned by the above online EM algorithm.

Algorithm 1 does not specify how often to synchronize the matrix $\Phi$ at steps 5–8. It can be done after every batch or less frequently (for instance if $\frac{\partial R}{\partial \phi}$ takes long time to evaluate). This flexibility is important for concurrent implementation of the algorithm, where multiple batches are processed in parallel. In this case synchronization can be triggered when a fixed number of documents had been processed since the last synchronization.

Each $D_b$ batch is stored on disk in a separate file, and only a limited number of batches is loaded into the main memory at any given time. The entire $\Theta$ matrix is also never stored in the memory. As a result, the memory usage stays constant regardless of the size of the collection.

To split collection into batches and process them concurrently is a common approach, introduced in AD-LDA algorithm [20], and then further developed in PLDA [36] and PLDA+ [17] algorithms. These algorithms require all concurrent workers to become idle before an update of the $\Phi$ matrix. Such synchronization step adds a large overhead in the online algorithm where $\Phi$ matrix is updated multiple times on each iteration. An alternative architecture without the synchronization step is described in [28], however it mostly targets a distributed cluster environment. In our work we develop an efficient single-node architecture where all workers benefit from the shared memory space.

To run multiple ProcessBatch in parallel the inputs and outputs of this routine are stored in two separate in-memory queues, locked for push and pop operations with spin locks. This approach does not add any noticeable synchronization overhead because both queues only store smart pointers to the actual data objects, so push and pop operations does not involve copying or relocating objects in the memory.

Smart pointers are also essential for lifecycle of the $\Phi$ matrix. This matrix is read by all processors threads, and can be written at any time by the merger thread. To update $\Phi$ without pausing all processor threads we keep two copies—an active $\Phi$ and a background $\Phi$ matrices. The active matrix is read-only, and is used by the processor threads. The background matrix is being built in a background by the merger thread at steps 6 and 7 of Algorithm 1, and once it is ready merger thread marks it as active. Before processing a new batch the processor thread gets the current active matrix from the merger thread. This object is passed via shared smart pointer to ensure that processor thread can keep ownership of its $\Phi$ matrix until the batch is fully processed. As a result, all processor threads keep running concurrently with the update of $\Phi$ matrix.

All processor threads share the same $\Phi$ matrix, which means that memory usage stays at constant level regardless of how many cores are used for computation. Using memory for two copies of the $\Phi$ matrix in our opinion gives
a reasonable usage balance between memory and CPU resources. An alternative solution with only one Φ matrix is also possible, but it would require a heavy usage of atomic CPU instructions. Such operations are very efficient, but still come at a considerable synchronization cost, and using them for all reads and writes of the Φ matrix would cause a significant performance degradation for merger and processor threads. Besides, an arbitrary overlap between reads and writes of the Φ matrix eliminates any possibility of producing a deterministic result. The design with two copies of the Φ matrix gives much more control over this and in certain cases allows the algorithm to behave in a fully deterministic way.

The design with two Φ matrices only supports a single merger thread, and we believe it should handle all terministic way. Of the Φ matrix gives much more control over this and in certain cases allows the algorithm to behave in a fully deterministic way.

The resulting dictionary was capped at \(|W| = 100,000\) most frequent words. Perplexity is used as an intrinsic quality measure:

\[
\mathcal{B}(D,p) = \exp\left(-\frac{1}{n} \sum_{d\in D} \sum_{w\in d} n_{dw} \ln p(w | d)\right). \tag{11}
\]

Vowpal Wabbit (VW) is a library of online algorithms that cover a wide range of machine learning problems. For topic modeling VW has the VW.LDA algorithm, based on the Online Variational Bayes LDA [11]. VW.LDA is neither multi-core nor distributed, but an effective single-threaded implementation in C++ made it one of the fastest tools for topic modeling.

Gensim library specifically targets the area of topic modeling and matrix factorization. It has two LDA implementations — LdaModel and LdaMulticore, both based on the same algorithm as VW.LDA (Online Variational Bayes LDA [11]). Gensim is entirely written in Python. Its high performance is achieved through the usage of NumPy library, built over low-level BLAS libraries (such as Intel MKL, ATLAS, or OpenBLAS). In LdaModel all batches are processed sequentially, and the concurrency happens entirely within NumPy. In LdaMulticore the workflow is similar to BigARTM — several batches are processed concurrently, and there is a single aggregation thread that asynchronously merges the results.

Table 1 compares the performance of VW.LDA, Gensim LdaModel and LdaMulticore (v0.10.3 under Python 2.7), and BigARTM, using Amazon EC2 c3.8xlarge instance (Intel-based CPU with 16 physical cores and hyper-threading).

Each run performs one pass over the Wikipedia corpus and produces a model with \(|T| = 100\) topics. The collection

\[
\sum_{\phi \in D} \theta_{\phi} b_{\phi} \text{ sum runs on contiguous memory blocks. In both matrices all values smaller than } 10^{-16} \text{ are always replaced with zero to avoid performance issues with denormalized numbers.}
\]

5. EXPERIMENTS

Runtime performance.

In first experiment we evaluate the runtime performance and intrinsic quality of BigARTM against two popular software packages — Gensim [24] and Vowpal Wabbit. All three libraries (VW.LDA, Gensim and BigARTM) work out-of-core, e. g. they are designed to process data that is too large to fit into a computer’s main memory at one time. This allowed us to benchmark on a fairly large collection — 3.7 million articles from the English Wikipedia. The conversion to bag-of-words was done with gensim.make_wikicorpus script, which excludes all non-article pages (such as category, file, template, user pages, etc), and also pages that contain less than 50 words. The dictionary is formed by all words that occur in at least 20 documents, but no more than in 10% documents in the collection. The resulting dictionary was capped at \(|W| = 100,000\) most frequent words. Perplexity is used as an intrinsic quality measure:

\[
\mathcal{B}(D,p) = \exp\left(-\frac{1}{n} \sum_{d\in D} \sum_{w\in d} n_{dw} \ln p(w | d)\right). \tag{11}
\]

Vowpal Wabbit (VW) is a library of online algorithms that cover a wide range of machine learning problems. For topic modeling VW has the VW.LDA algorithm, based on the Online Variational Bayes LDA [11]. VW.LDA is neither multi-core nor distributed, but an effective single-threaded implementation in C++ made it one of the fastest tools for topic modeling.

Gensim library specifically targets the area of topic modeling and matrix factorization. It has two LDA implementations — LdaModel and LdaMulticore, both based on the same algorithm as VW.LDA (Online Variational Bayes LDA [11]). Gensim is entirely written in Python. Its high performance is achieved through the usage of NumPy library, built over low-level BLAS libraries (such as Intel MKL, ATLAS, or OpenBLAS). In LdaModel all batches are processed sequentially, and the concurrency happens entirely within NumPy. In LdaMulticore the workflow is similar to BigARTM — several batches are processed concurrently, and there is a single aggregation thread that asynchronously merges the results.

Table 1 compares the performance of VW.LDA, Gensim LdaModel and LdaMulticore (v0.10.3 under Python 2.7), and BigARTM, using Amazon EC2 c3.8xlarge instance (Intel-based CPU with 16 physical cores and hyper-threading).

Each run performs one pass over the Wikipedia corpus and produces a model with \(|T| = 100\) topics. The collection

\[
\sum_{\phi \in D} \theta_{\phi} b_{\phi} \text{ sum runs on contiguous memory blocks. In both matrices all values smaller than } 10^{-16} \text{ are always replaced with zero to avoid performance issues with denormalized numbers.}
\]

5. EXPERIMENTS

Runtime performance.

In first experiment we evaluate the runtime performance and intrinsic quality of BigARTM against two popular software packages — Gensim [24] and Vowpal Wabbit. All three libraries (VW.LDA, Gensim and BigARTM) work out-of-core, e. g. they are designed to process data that is too large to fit into a computer’s main memory at one time. This allowed us to benchmark on a fairly large collection — 3.7 million articles from the English Wikipedia. The conversion to bag-of-words was done with gensim.make_wikicorpus script, which excludes all non-article pages (such as category, file, template, user pages, etc), and also pages that contain less than 50 words. The dictionary is formed by all words that occur in at least 20 documents, but no more than in 10% documents in the collection. The resulting dictionary was capped at \(|W| = 100,000\) most frequent words. Perplexity is used as an intrinsic quality measure:

\[
\mathcal{B}(D,p) = \exp\left(-\frac{1}{n} \sum_{d\in D} \sum_{w\in d} n_{dw} \ln p(w | d)\right). \tag{11}
\]

Vowpal Wabbit (VW) is a library of online algorithms that cover a wide range of machine learning problems. For topic modeling VW has the VW.LDA algorithm, based on the Online Variational Bayes LDA [11]. VW.LDA is neither multi-core nor distributed, but an effective single-threaded implementation in C++ made it one of the fastest tools for topic modeling.

Gensim library specifically targets the area of topic modeling and matrix factorization. It has two LDA implementations — LdaModel and LdaMulticore, both based on the same algorithm as VW.LDA (Online Variational Bayes LDA [11]). Gensim is entirely written in Python. Its high performance is achieved through the usage of NumPy library, built over low-level BLAS libraries (such as Intel MKL, ATLAS, or OpenBLAS). In LdaModel all batches are processed sequentially, and the concurrency happens entirely within NumPy. In LdaMulticore the workflow is similar to BigARTM — several batches are processed concurrently, and there is a single aggregation thread that asynchronously merges the results.

Table 1 compares the performance of VW.LDA, Gensim LdaModel and LdaMulticore (v0.10.3 under Python 2.7), and BigARTM, using Amazon EC2 c3.8xlarge instance (Intel-based CPU with 16 physical cores and hyper-threading).

Each run performs one pass over the Wikipedia corpus and produces a model with \(|T| = 100\) topics. The collection

\[
\sum_{\phi \in D} \theta_{\phi} b_{\phi} \text{ sum runs on contiguous memory blocks. In both matrices all values smaller than } 10^{-16} \text{ are always replaced with zero to avoid performance issues with denormalized numbers.}
\]
Table 1: The comparison of BigARTM with VW.LDA and Gensim; train is the time for model training, inference is the time for calculation of \( \theta \), of 100 000 held-out documents, perplexity is calculated according to (11) on held-out documents.

<table>
<thead>
<tr>
<th>Library</th>
<th>_procs</th>
<th>train</th>
<th>inference</th>
<th>perplexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>BigARTM</td>
<td>1</td>
<td>35 min</td>
<td>72 sec</td>
<td>4000</td>
</tr>
<tr>
<td>LdaModel</td>
<td>1</td>
<td>309 min</td>
<td>395 sec</td>
<td>4161</td>
</tr>
<tr>
<td>VW.LDA</td>
<td>1</td>
<td>73 min</td>
<td>120 sec</td>
<td>4108</td>
</tr>
<tr>
<td>BigARTM</td>
<td>4</td>
<td>9 min</td>
<td>20 sec</td>
<td>4061</td>
</tr>
<tr>
<td>LdaMulticore</td>
<td>4</td>
<td>60 min</td>
<td>222 sec</td>
<td>4111</td>
</tr>
<tr>
<td>BigARTM</td>
<td>8</td>
<td>4.5 min</td>
<td>14 sec</td>
<td>4304</td>
</tr>
<tr>
<td>LdaMulticore</td>
<td>8</td>
<td>57 min</td>
<td>224 sec</td>
<td>4455</td>
</tr>
</tbody>
</table>

Table 2: Comparison of LDA and BigARTM models: \( P_{10k} \), \( P_{100k} \) — hold-out perplexity on 10K and 100K documents sets, \( S_\phi, S_\Theta \) — sparsity of \( \Phi \) and \( \Theta \) matrices (in %), \( \kappa_\phi, \kappa_\Theta \), \( K_p \), \( K_c \) — average topic kernel size, purity and contrast respectively.

<table>
<thead>
<tr>
<th>Model</th>
<th>( P_{10k} )</th>
<th>( P_{100k} )</th>
<th>( S_\phi )</th>
<th>( S_\Theta )</th>
<th>( \kappa_\phi )</th>
<th>( \kappa_\Theta )</th>
<th>( K_p )</th>
<th>( K_c )</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDA</td>
<td>3436</td>
<td>3801</td>
<td>0.0</td>
<td>0.0</td>
<td>873</td>
<td>0.533</td>
<td>0.607</td>
<td></td>
</tr>
<tr>
<td>ARTM</td>
<td>3577</td>
<td>3947</td>
<td>96.3</td>
<td>80.9</td>
<td>1079</td>
<td>0.785</td>
<td>0.731</td>
<td></td>
</tr>
</tbody>
</table>

The LDA model was split into batches with 10K documents each (\texttt{chunksize} in Gensim, \texttt{minibatch} in VW.LDA). The update rule in online algorithm used a discounting factor \( \rho = (b + \tau_0)^{-0.5} \), where \( b \) is the number of batches processed so far, and \( \tau_0 \) is a constant offset parameter introduced in [11], in our experiment \( \tau_0 = 64 \). Updates were performed after each batch in non-parallel runs, and after \( P \) batches when running in \( P \) threads. To make a fair comparison we have configured BigARTM to only use the smoothing regularizers, which is equivalent to the LDA model. LDA priors were fixed as \( \alpha = 0.1, \beta = 0.1 \) for all models.

Combination of regularizers.

All regularizers built-in BigARTM library can be used in any combination. In the following experiment we combine regularizers described in section 2: sparsing of \( \phi \), sparsing of \( \theta \), and pairwise decorrelation of \( \phi \) distributions. This combination improves several quality measures without significant loss of perplexity for the offline implementation of ARTM [33]. The goal of our experiment is to show that the same remains true for the online implementation in BigARTM.

We use the following built-in performance measures: the hold-out perplexity, the sparsity of \( \Phi \) and \( \Theta \) matrices, and several characteristics (size, purity, and contrast) of the topic’s lexical kernels, averaged across all topics.

Table 2 compares the results of additive combination of regularizers (ARTM) and the usual LDA model.

Figure 2 presents performance measures as functions of the number of processed documents. The first chart shows perplexity and sparsity of \( \Phi \), \( \Theta \) matrices, and the second chart shows average lexical kernel measures.

Text classification.

Support vector machine (SVM) based on token frequencies is known to be one of the best methods for text classification. However, according to [25] topic models demonstrate even better quality in case of unbalanced interdependent and intersecting classes.

Our experiment aims to prove that multimodal regularized topic models in BigARTM are as good as Dependency LDA from [25]. Dependency LDA is in fact a multimodal topic model with two modalities: words and class labels.

The EUR-lex collection contains about 20K documents split into train and test sets to provide the reproducibility of the results [25]. The original size of the dictionary is over 190K tokens. Preprocessing from [25] removes all tokens encountered less than 20 times, and reduces the dictionary to about 20K tokens. Class labels, encountered only once, are also removed to result in about 3250 classes. Each document might belong to several classes.

For both Dependency LDA and ARTM the label regularization [25] was used. The quality measures in our experiment are as follows: \( \text{AUC}_{PR} \) — the area under the precision-recall curve; \( \text{AUC} \) — the area under ROC-curve; OneErr — the ratio of documents with the most probable label not from the correct set; IsErr — the ratio of documents with not ideal classification.

The results are provided in Table 3. ARTM performs better than both Dependency LDA and SVM by the three measures out of four. It is interesting to note that while the number of topics increases up to 15000, ARTM provides better classification quality, while the optimal number of topics for Dependency LDA is 200.

Cross-language search.

The following experiment shows that multimodal topic model may be used as multilingual one, with languages of
Table 3: Multimodal ARTM, Dependency LDA and SVM classification models. The best results are in bold. $T_{opt}$ is the optimal number of topics.

<table>
<thead>
<tr>
<th>Model</th>
<th>$T_{opt}$</th>
<th>AU$C_P$</th>
<th>AU$C_I$</th>
<th>OneErr</th>
<th>IsErr</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARTM</td>
<td>15 000</td>
<td>0.529</td>
<td>0.980</td>
<td>27.1</td>
<td>94.2</td>
</tr>
<tr>
<td>DTLDA</td>
<td>200</td>
<td>0.492</td>
<td>0.982</td>
<td>32.0</td>
<td>97.2</td>
</tr>
<tr>
<td>SVM</td>
<td>–</td>
<td>0.435</td>
<td>0.975</td>
<td>31.6</td>
<td>98.1</td>
</tr>
</tbody>
</table>

Table 4: Cross-language search precision for different models. The best value in each column is bolded.

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of topics $T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLTM [19]</td>
<td>50 100 200 500</td>
</tr>
<tr>
<td>JPLSA [23]</td>
<td>0.812</td>
</tr>
<tr>
<td>PLTM-He [18]</td>
<td>0.989</td>
</tr>
<tr>
<td>PLTM-He kd-trees [18]</td>
<td>0.943 0.985 0.994 0.993</td>
</tr>
<tr>
<td>BigARTM</td>
<td>0.972 0.990 0.996 0.997</td>
</tr>
</tbody>
</table>

parallel texts treated as modalities. The experiment was held on the EuroParl collection [15] of European Parliament Proceedings. Proceedings in English and Spanish were chosen, as these languages are often used for mulitlingual topic model comparison. As in [19, 23, 18], a single document is a speech of one speaker at one session.

We use precision to measure quality of the cross-language search. The precision is defined as the fraction of query documents $q$ closest to their own translation $t$, according to Hellinger distance:

$$ H^2(d, q) = \frac{1}{2} \sum_{t \in T} (\sqrt{p(t|d)} - \sqrt{p(t|q)})^2. $$

Training set includes proceedings from 1996 to 1999, and from 2001 to 2002, test set includes proceedings of 2000 and the first 9 months of 2003. The same partitioning is used in [23] and [18]. Moreover, as in [19, 18], the test comprised documents of the length more or equal than 100 words. The total number of documents is 67379 in the training set, and 16068 in the test set. Built-in capability of BigARTM to filter the dictionary was used: all rare words, that appear in less than 20 documents, and stop-words, that appear in more than 50% of documents, were discarded.

Table 4 shows the comparison of models in [19, 23, 18] and our ARTM. For the first two models, the authors provide search precision for 50 topics only. ARTM performs slightly worse than JPLSA, but we note, that one iteration of BigARTM takes 30 seconds for 50 topics and 40 seconds for 100 topics, while one iteration of JPLSA takes 31 minutes. ARTM performs better if compared with models from [18].

**Recommending articles of collective blog.**

Here we describe how multimodal topic modeling can be used for recommending articles in a collective blog. Collective blog is an on-line platform where users can publish articles and respond to the articles of other authors. To make recommendations we add user’s positive feedback to the article as a modality. For the experiment we used dataset of about 130K articles with user feedback from http://habrahabr.ru — the most popular IT-oriented soc-

Table 5: The quality of recommendations for baseline matrix factorization model, unimodal model with only modality of user likes, and two multimodal models incorporating words and user-specified data (tags and categories).

<table>
<thead>
<tr>
<th>Model</th>
<th>Recall@5</th>
<th>Recall@10</th>
<th>Recall@20</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline [13]</td>
<td>0.591</td>
<td>0.652</td>
<td>0.678</td>
</tr>
<tr>
<td>likes</td>
<td>0.62</td>
<td>0.59</td>
<td>0.65</td>
</tr>
<tr>
<td>likes + words</td>
<td>0.79</td>
<td>0.64</td>
<td>0.68</td>
</tr>
<tr>
<td>all modalities</td>
<td>0.80</td>
<td>0.71</td>
<td>0.69</td>
</tr>
<tr>
<td>no regularization</td>
<td>0.79</td>
<td>0.71</td>
<td>0.68</td>
</tr>
</tbody>
</table>

6. CONCLUSIONS

We have presented an Additive Regularization of Topic Models (ARTM), a powerful non-Bayesian framework for topic modeling. ARTM facilitates the development of topic models and allows merging models together in arbitrary combinations. Combining multiple modalities with multiple regularization criteria covers dozens of models previously studied in the Bayesian settings.

BigARTM is an open source project for parallel online multimodal regularized topic modeling of large text collections. Its implementation is faster than existing popular topic modeling tools. BigARTM provides high flexibility for various applications due to multimodality and additive combinations of regularizers. BigARTM has a built-in library of regularizers and quality measures.

BigARTM architecture has a rich potential. In future version it will be extended to run on a distributed cluster environment, improve performance and reduce memory usage for sparse topic models, implement APIs for Java and C#.
Acknowledgements.

The work was supported by the Russian Foundation for Basic Research (14-07-00847, 14-07-00908, 14-07-31176), by Skolkovo Institute of Science and Technology (081-R), and Russian Science Foundation (15-18-00091).

7. REFERENCES


[31] K. V. Vorontsov. Additive regularization for topic


**Appendix A**

Consider the system of equations (8)–(10).

A topic $t$ is called *regular* for a modality $m$ if $n_{wt} + \phi_{wt} \frac{\partial R}{\partial \theta_{td}} > 0$ for at least one term $w \in W^m$. If the reverse inequality holds for all $w \in W^m$ then the topic $t$ is called *irregular*: in this case the $t$-th vector-column in the matrix $\Phi^m$ equals zero and cannot represent a discrete distribution. This means that the topic $t$ for the modality $m$ must be excluded from the model. This mechanism can be used determine the number of topics.

A document $d$ is called *regular* if $n_{td} + \theta_{td} \frac{\partial R}{\partial \theta_{td}} > 0$ for at least one topic $t \in T$. If the reverse inequality holds for all $t \in T$ then the document $d$ is called *irregular*: in this case the $d$-th vector-column in the matrix $\Theta$ equals zero and cannot represent a discrete distribution. This means that the document $d$ must be excluded from the model. For example, it may be too short or irrelevant for the collection.

**Theorem 1.** If the function $R(\Phi, \Theta)$ is continuously differentiable and $(\Phi, \Theta)$ is the local maximum of the problem (6), (7) then for any regular topic $t$ and any regular document $d$ the system of equations (8)–(10) holds.

**Proof.** For the local minimum $\Phi^m, \Theta$ of the problem (6), (7) the Karush–Kuhn–Tucker (KKT) conditions can be written as follows:

\[
\sum_d n_{wd} \frac{\theta_{td}}{p(w|d)} + \frac{\partial R}{\partial \mu_{td}} = \lambda_t - \lambda_{wt}; \quad (12)
\]

\[
\lambda_{wt} \geq 0; \quad \lambda_{wt} \phi_{wt} = 0; \quad (13)
\]

\[
\sum_m \sum_{w \in W^m} n_{wd} \frac{\phi_{wt}}{p(w|d)} + \frac{\partial R}{\partial \mu_{td}} = \mu_d - \mu_{td}; \quad (14)
\]

\[
\mu_{td} \geq 0; \quad \mu_{td} \theta_{td} = 0; \quad (15)
\]

where $\lambda_t$ and $\mu_d$ are KKT multipliers for normalization constraints, $\lambda_{wt}$ and $\mu_{td}$ are KKT multipliers for nonnegativity constraints.

Let us multiply both sides of equation (12) by $\phi_{wt}$, both sides of equation (13) by $\theta_{td}$, and reveal the auxiliary variable $p_{tdw}$ from (8) in the left-hand side of both equations. Then let us sum the right-hand side of the first equation over $d$, the right-hand side of the second equation over $t$:

\[
\phi_{wt} \lambda_t = \sum_d n_{wd} \frac{\phi_{wt} \theta_{td}}{p(w|d)} + \phi_{wt} \frac{\partial R}{\partial \phi_{wt}} = n_{wt} + \phi_{wt} \frac{\partial R}{\partial \phi_{wt}}, \quad (16)
\]

\[
\theta_{td} \mu_d = \sum_m \sum_{w \in W^m} n_{wd} \frac{\phi_{wt} \theta_{td}}{p(w|d)} + \theta_{td} \frac{\partial R}{\partial \theta_{td}} = n_{td} + \theta_{td} \frac{\partial R}{\partial \theta_{td}}. \quad (17)
\]

An assumption that $\lambda_t \leq 0$ contradicts the regularity condition for the $(t, m)$ pair. Then $\lambda_t > 0$. Either $\phi_{wt} = 0$ or both sides of the first equation are positive. Combining these two cases in one formula, we write:

\[
\phi_{wt} \lambda_t = \max\left\{n_{wt} + \phi_{wt} \frac{\partial R}{\partial \phi_{wt}}, 0\right\}. \quad (18)
\]

Analogously, an assumption that $\mu_d \leq 0$ contradicts the regularity condition for the document $d$. Then $\mu_d > 0$. Either $\theta_{td} = 0$ or both sides of the second equation are positive, consequently:

\[
\theta_{td} \mu_d = \max\left\{n_{td} + \theta_{td} \frac{\partial R}{\partial \theta_{td}}, 0\right\}. \quad (19)
\]

Let us sum both sides of the first equation over $w \in W^m$, then both sides of the second equation over $t \in T$:

\[
\lambda_t = \sum_{w \in W^m} \max\left\{n_{wt} + \phi_{wt} \frac{\partial R}{\partial \phi_{wt}}, 0\right\}; \quad (20)
\]

\[
\mu_d = \sum_{t \in T} \max\left\{n_{td} + \theta_{td} \frac{\partial R}{\partial \theta_{td}}, 0\right\}. \quad (21)
\]

Finally, we obtain (9) and (10) by expressing $\phi_{td}$ from (14) and (16), then by expressing $\theta_{td}$ from (15) and (17). □