Faster PLSA

Valeriy Avanesov Ilya Kozlov

The Institute for System Programming (ISP) of the Russian Academy of Sciences
kozlov-ilya@ispras.ru, avanesov@ispras.ru

Abstract

Probabilistic Latent Semantic Analysis (PLSA) is an effective technique for information retrieval, but it has a serious drawback: it consumes a huge amount of computational resources, so it is hard to train this model on a large collection of documents. The aim of this paper is to improve time efficiency of the training algorithm. Two different approaches are explored: one is based on efficient finding of an appropriate initial approximation, the idea of another is that for the most of collection topics may be extracted from relatively small fraction of the data.

1 Introduction

Topic modeling is an application of machine learning to text analysis. Topic modeling is useful for different text analysis tasks, for example: document categorization [1], spam detection [2], phishing detection [3] and many other applications.
One of the widespread algorithms is Probabilistic Latent Semantic Analysis (PLSA), suggested by Thomas Hofmann in [4].

1.1 Generative model

PLSA is based on generative model "bag of words": every document is assumed to be a multinomial distribution over topics. Every topic is a multinomial distribution over words. Generation model may be defined as follow:

- For every position in document d i.i.d choose topic t from distribution of topics by document
- Choose word w from topic t

The aim of topic modeling is to recover topics and distribution of document by topics.

1.2 Topic modeling as optimization problem

According to generative model one can estimate probability to observe collection D as:

$$p(D) = \prod_{d\in D}\prod_{w\in d}\sum_{t} p(t|d)p(w|t) \tag{1}$$

Proceedings of the Tenth Spring Researcher's Colloquium on Databases and Information Systems, Veliky Novgorod, Russia, 2014

Denote $\varphi_{wt} = p(w|t)$ and $\theta_{td} = p(t|d)$. One may obtain φ_{wt} and θ_{td} as solution of optimization problem

$$L = \sum_{d\in D}\sum_{w\in d}\log\sum_{t}\varphi_{wt}\theta_{td} \to \max \tag{2}$$

with boundary

$$\forall t \ \sum_{w}\varphi_{wt} = 1, \ \forall d \ \sum_{w}\theta_{td} = 1 \tag{3}$$

and

$$\forall t, w \ \varphi_{wt} \geq 0, \ \forall d, t \ \theta_{wt} \geq 0 \tag{4}$$

1.3 Topic modeling as matrix decomposition

1.3.1 Kullback-Leibler divergence

Kullback-Leibler divergence is a non-negative measure of diffrence between two different probability distribution:

$$KL(p_i||q_i) = \sum_{i=1}^{n} p_i \ln\left(\frac{p_i}{q_i}\right) \tag{5}$$

Consider an empirical distribution $\hat{p}_i$ and some parametric distribution $q_i = q_i(\alpha)$ which is used to explain $\hat{p}_i$. Easy to see that in this case minimization of KL–divergence is equivalent to estimation of α by maximum-likelihood:

$$KL(p_i||q_i(\alpha)) = \sum_{i=1}^{n} p_i \ln\left(\frac{p_i}{q_i(\alpha)}\right) \to \min_{\alpha} \tag{6}$$

$$\Leftrightarrow \sum_{i=1}^{n} p_i \ln(q_i(\alpha)) \to \max_{\alpha}$$

Thus one can easily see that (2) equivalent to weighted Kullback-Leibler divergence minimization:

$$\sum_{d\in D} n_d KL_w\left(\frac{n_{dw}}{n_d}||\sum_{t\in T}\varphi_{wt}\theta_{td}\right) \to \min_{\Phi,\Theta} \tag{7}$$

where n_{wd}– number of words w in document d, n_d – number of words in document d.

1.3.2 Matrix decomposition

Denote empirical distribution of words by document as $\hat{p}(w,d) = \frac{n_{wd}}{n_d}$. According to this notation one can consider the problem (2) as matrix decomposition:

$$F \approx_{KL} \Phi\Theta \tag{8}$$

where matrix $F = (\hat{p}(w,d))_{W \times D}$ is empirical distribution of words by document, matrix $\Phi = (\varphi_{wt})_{W \times D}$ is distribution of words by topics and matrix $\Theta = (\theta_{td})_{T \times D}$ is distribution of documents by topics. Thus, our optimization problem may be rewritten in Kullback-Leibler notation as

$$KL(F, \Phi\Theta) \to \min \tag{9}$$

Thus PLSA may be observed as stochastic matrix decomposition.

1.4 Expectation-Maximization algorithm

Unfortunately (2) has no analytical solution. Thus we use Expectation - Maximization (EM) algorithm. This algorithm consists of two steps:

1. Estimation of number n_{dwt} of words w, produced by topic t in document d. (E - step)
2. Optimization of distribution of documents by topics and optimization of distribution of topics by words relying on the n_{dwt} values obtained during E - step . (M - step)

One can estimate n_{dwt} as follows:

$$n_{dwt} = \frac{n_{wd} p(w|t) p(t|d)}{\sum_t p(w|t) p(t|d)} \tag{10}$$

where n_{wd} – number of words w in document d. Thus, probability $p(w|t)$ may be estimated as

$$p(w|t) = \frac{n_{wt}}{n_t} = \frac{\sum_d n_{dwt}}{\sum_w \sum_d n_{dwt}} \tag{11}$$

Where n_{wt} – number of words w, produced by topic t:

$$n_{wt} = \sum_{d \in D} n_{dwt} \tag{12}$$

n_t – number of words, produced by topic t

$$n_t = \sum_{w \in V} n_{wt} \tag{13}$$

Similarly for $p(t|d)$:

$$p(t|d) = \frac{n_{td}}{n_d} \tag{14}$$

Where n_d – number of words in document d, n_{td} – estimated number of words in document d, produced by topic t:

$$n_{td} = \sum_{w \in V} n_{dwt} \tag{15}$$

As one can see, the asymptotic time of this algorithm is $O(D \times V \times T \times I)$ where D – the number of documents, V – average number of distinct words in document, T - – number of topics and I – number of iterations until convergence. Inference of the PLSA on a large dataset requires a lot of time thus the methods of decreasing of computation time are important.
Number of topics and number of documents are defined by application. Size of vocabulary (number of distinct words) can be decreased by text normalization (removing of stop-words, lowercasing, etc). Number of iterations until convergence depends of initial approximation of PLSA parameters Φ and Θ, so a good initial approximation can reduce the number of iterations until convergence. The current study presents an efficient approach to find a beneficial initial approximation. The other method of computation time reduction based on idea that matrix Φ may be obtained on small representative part of documents collection.

2 Related Work

The original algorithm was described in 1999 in [4]. Since 1999 numerous papers were devoted to PLSA, but only a few of them are devoted to time efficiency improvement. In [5] authors improve time efficiency by parallelization of the algorithm using OpenMP. Authors report 6 times speed-up on 8 CPU machine. Work [6] improves the result of a previous work by using MPI. But both of these studies try to solve the problem of time efficiency purely by programming methods.

In [7] Farahat uses LSA for finding an initialization for PLSA. LSA is based on SVD [1] matrix decomposition in L_2 norm and lacks probabilistic interpretation. PLSA performs stochastic matrix decomposition based on Kullback-Leibler divergence and has a simple probability interpretation, but it inherits the problem of every non-convex optimization algorithm: it may converge to a local minimum instead of the global one. Combination of LSA and PLSA leverages the best features of these models: usage of LSA training result as an initial approximation helps to avoid convergence to a poor local minimum. But the problem of time efficiency is not explored. In [7] it is shown that L_2 norm usage is appropriate to find an initialization for PLSA inference algorithm, we will use this result in our work.
An idea of obtaining a distribution over topics for a document not included in a collection that PLSA was initially trained on was expressed in [8]. The author suggests to perform this through EM-scheme holding matrix Φ fixed. However, he proposes this method only for query processing but not for PLSA training speed-up.

3 Proposed approach

In this work we present two different approach for computational time reduction. One is based on finding initial approximation and reduction of number of iteration to convergence. The other is based on obtaining Φ_{part} on representative sample, fixing Φ_{part} and then obtaining Θ on whole collection.

3.1 Finding initial approximation

In this work we do not use LSA nor clustering methods. Instead we take a subset of our collection (for example 10%), apply PLSA to this sample and calculate an initial approximation using obtained matrix Φ. Computa-

[1] SVD – Singular value decomposition, a factorization of a matrix into the product of a unitary matrix, a diagonal matrix, and another unitary matrix

tion time of PLSA is proportional to the number of documents in collection and training PLSA on 10% part of collection is at least ten times faster than on the whole collection (per iteration).

3.1.1 Taking a sample

In order to take a representative sample we need to take a random sample. The exact size of sample is not important so we use a rather simple scheme: we include documents independently with probability 10%. So we take a representative part of collection and its size is approximately 10% of size of the whole collection.

3.1.2 Initial approximation of Φ (words-topics)

For training PLSA on the sample we use a random initialization. Computation time is linear by the number of the documents in the collection, so the process of training turns out to be relatively fast. The obtained matrix Φ_{part} can be used as initial approximation of matrix Φ for the whole collection. An issue of this approach is that some words from the vocabulary do not occur in the sample and every topic in matrix Φ_{part} has zero weight for these words. If we would use matrix Φ_{part} as is, these probabilities would stay zero on every step (10), (2). It would have disastrous result for likelihood (or perplexity) of our model:

$$likelihood = \prod_{w \in d} \sum_{t \in \Theta} p(w|t)\theta_t \equiv 0 \qquad (16)$$

because some word w^* would have zero probability for every topic. Thus some kind of smoothing is necessary. In this work we use a trivial one:

1. add some constant for every position in every topic

$$\forall t, w \;\; p(w|t) \mathrel{+}= const$$

In this work we use

$$const = \frac{1}{vocabularySize}$$

2. normalize:

$$\forall t \sum_w p(w|t) = 1$$

3.1.3 Initial approximation of Θ (document-topics)

During the previous step we found an initial approximation for matrix Φ (words by topic). Now we have to find an initial approximation for matrix Θ (documents-topics) given Φ. Its every column $\Theta_{d.}$ can be found as a solution of the following optimization problem:

$$\Theta_{d.} = arg\max_{\theta} P(d|\Theta) = arg\max_{\theta} \prod_{w \in d} \sum_{t \in \Theta} p(w|t)\theta_{wt} \qquad (17)$$

with boundaries

$$\sum_{t \in \Theta} \theta_t = 1 \qquad (18)$$

But our aim is to decrease the training time and computation method of solving this problem is not fast enough. We propose to find $\Theta_{d.}$ in the norm L_2. The solution in L_2 would not be a solution in our space with Kullback-Leibler divergence, but we are not looking for the exact solution, but for an appropriate initialization.

Assume that all words are replaced by their serial numbers in the vocabulary. Let us consider topics and documents as vector in $\mathbb{R}^V$, where V stands for vocabulary size. $i-th$ coordinate in vector-document represents the number of times a word with the number i occurs in the document:

$$\vec{d}^{(i)} = \#(word \;\; i \;\; occur \;\; in \;\; document \;\; d)$$

For topic-vector $i-th$ coordinate shows probability to generate word i from this topic t:

$$\vec{t}^{(i)} = p(i|t)$$

Consider vector space, formed by topic-vectors. Topics form basis in this space. One can find an initial approximation for distribution of documents d by topics as orthogonal projection to this space:

$$\vec{d} = \vec{d}_{\perp} + \vec{d}_{\parallel} \qquad (19)$$

Where

$$\forall \vec{t} \in \Theta \;\; (\vec{d}_{\perp}, \vec{t}) = 0 \qquad (20)$$

For computation efficiency we perform orthogonalization and normalization of basis $\{\vec{t}\} \rightarrow \{\vec{t'}\}$, where $\forall i \neq j (t'_i, t'_j) = 0$ and $\forall i (t'_i, t'_i) = 1$. It allows us to find the projection faster and simpler:

$$\vec{d}_{\parallel} = \sum_i \alpha_i \vec{t'}_i \qquad (21)$$

where $\vec{t'}_i$ – i-th vector of the orthonormal basis

$$(\vec{d}, \vec{t'}_i) = (\vec{d}_{\parallel}, \vec{t'}_i) = (\vec{t'}_i, \sum_i \alpha_i \vec{t'}_i) = \alpha_i \qquad (22)$$

Where scalar product is defined as follows

$$(\vec{x}, \vec{y}) = \sum_{i=1}^{V} x^i * y^i \qquad (23)$$

Document-vector expansion in the orthonormal basis is obtained, so one can return to topics basis and perform normalization as follows:

$$\sum_{t \in \Theta} d_t = 1 \qquad (24)$$

Where d_t – weight of topic t in document d

Due to the nature of L_2 norm some topic weight may be too small or even negative, so smoothing is necessary. We do it analogously to the previous subsection:

1. Replace negative weight by zero
2. add some constant for every weight. In this paper we use

$$const = \frac{1}{numberOfTopics}$$

3. perform normalization

$$\forall d \sum_{t \in \Theta} d_t = 1$$

3.2 Fixed Φ approach

This approach is based on the fact that matrix Φ may be found by training PLSA on representative subset $D' \subset D$. The rows of the matrix Θ may be obtained through the following constrained optimization problem independently for each document $d \in D$:

$$L(\theta_{\cdot d}) = \sum_{w \in W} n_{dw} \ln \sum_{t \in T} \hat{\phi}_{wt} \theta_{td} \to \max_{\theta_{\cdot d}} \tag{25}$$

$$\sum_{t \in T} \theta_{td} = 1 \tag{26}$$

$$\theta_{td} \geq 0 \tag{27}$$

This approach consists of the following steps:

1. Take a random subset of documents $D' \in D$ of sufficient size s
2. Obtain $\hat{\Phi}$ through training PLSA on D' using EM algorithm described in [4]
3. Obtain $\theta_{\cdot d}$ through solving optimization problem (25), (26), (27) for each document $d \in D$

The third step needs some explanation.
We solve this problem with EM-algorithm.
E-step estimate the probabilities $p(t|d, w)$ as

$$p(t|d, w) = \frac{\phi_{wt}\theta_{td}}{\sum\limits_{\tau \in T} \phi_{w\tau}\theta_{\tau d}} \tag{28}$$

M-step In order to solve the problem (25), (26), (27) temporary omit the non-negativeness constraint (27) – we will see that the solution is non-negative.

Lagrange function for problem (25), (26) takes the form:

$$L(\theta_{\cdot d}) = \sum_{w \in W} n_{dw} \ln \sum_{t \in T} \hat{\phi}_{wt} \theta_{td} - \lambda(\sum_{t \in T} \theta_{td} - 1) \tag{29}$$

Take a derivative:

$$\frac{\partial L}{\partial \theta_{td}}(\theta_{\cdot d}) = \sum_{w \in W} n_{dw} \frac{\hat{\phi}_{wt}}{\sum\limits_{t \in T} \hat{\phi}_{wt}\theta_{td}} - \lambda = 0 \tag{30}$$

Move λ to the right part and multiply both sides by θ_{td} and according to (28)

$$\sum_{w \in W} n_{dw} p(w|t, d) = \lambda \theta_{td} \tag{31}$$

Now sum both sides by every $t \in T$

$$\sum_{w \in W} \sum_{t \in T} n_{dw} p(w|t, d) = \lambda \tag{32}$$

From equation (31) obtain θ_{td} and substitute λ from (32):

$$\theta_{td} = \sum_{w \in W} n_{dw} \frac{p(w|t, d)}{\sum\limits_{\omega \in W} \sum\limits_{\tau \in T} n_{d\omega} p(\omega|\tau, d)}$$

The denominator is independent of t, thus

$$\theta_{td} \propto \sum_{w \in W} n_{dw} p(w|t, d) \tag{33}$$

Primal feasibility is easily verifiable by θ_{td} summation by t.

Dual feasibility follows from (32) and probability non-negativeness.

So this point satisfies the Karush-Kuhn-Tucker conditions.

Also, one can easily see that $\theta_{td} \geq 0$, thus we have found a solution of the problem (25), (26), (27).

4 Experiments

We conduct two kinds of experiments: we evaluate perplexity [9] for our approaches and for classical PLSA and compare classification performance on topics distributions, obtained by PLSA and by our approaches.

4.1 Datasets

Both experiments were conducted on three datasets: tweets, news articles, abstracts of scientific papers.

- Twitter dataset
  Twitter dataset contains tweets, posted by 15000 Twitter users, written in English. We merge all tweets, posted by single user into a single document. Every document contains approximately 1000 tweets. Documents with less than 50 words are omitted.

- The 20 Newsgroups data set [2]
  The 20 Newsgroups dataset is often used for topic modeling testing on text categorization. It contains short news articles on one of twenty newsgroups. It is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups.

- arxive [3]
  The third data set consists of abstracts of scientific articles. It consists of approximately 900000 abstracts from 6 areas: Physics, Mathematics, Computer Science, Quantitative Biology, Quantitative Finance, Statistics. Distribution of articles by areas is not uniform: Physics contains 600 thousands, Mathematics contains 270 thousands and Quantitative Biology only 5 thousands. Abstracts with less than 20 words are omitted. For experiments with fixed Φ we omit small area and take only Physics, Mathematics and Computer Science.

Some text normalization is performed: stoplisting, lowercasing, rare words removing (ones that occur less than 5 times in the whole collection).

4.2 Initial approximation

Four types of initial approximation are compared:

[2] http://qwone.com/~jason/20Newsgroups/
[3] http://arxiv.org/

- Random initial approximation for matrix Φ (words by topic) and matrix Θ (document by topic). Denoted "randomly".
- Calculate an initial approximation for matrix Φ on sample. Use random initial approximation for matrix Θ. Denoted "phi".
- Calculate an initial approximation for matrix Θ on sample. Use random initial approximation for matrix Φ. Denoted "theta".
- Calculate an initial approximation for matrix Θ and matrix Φ on sample. Denoted "full".

4.2.1 Perplexity depending on initial approximation

We evaluate the dependence of perplexity on different types of initial approximation. During these experiments the number of iterations is fixed and equals 100. The number of topics is 25 for every experiment.

In figure 1, figure 2 and figure 3 one can see perplexity depending on number of iterations for different datasets and different types of initial approximation. The keys are given above in the beginning of Section 4.2

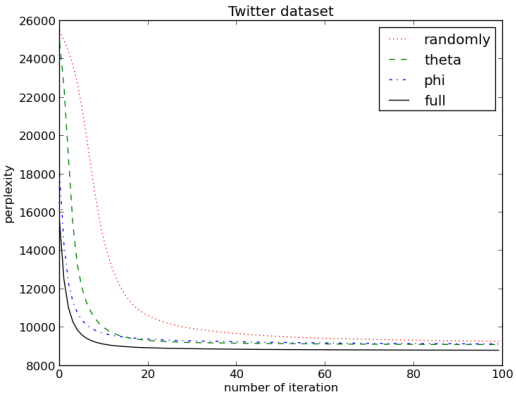

Figure 1: Perplexity for Twitter data set

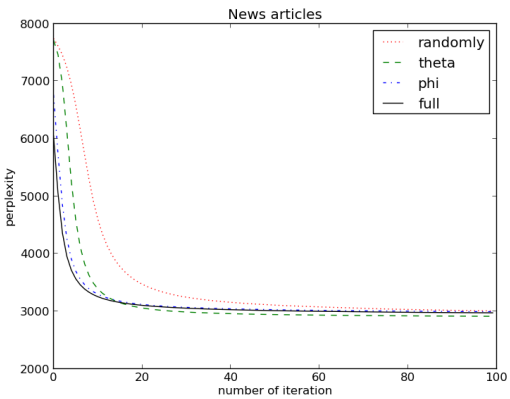

Figure 2: Perplexity for news articles

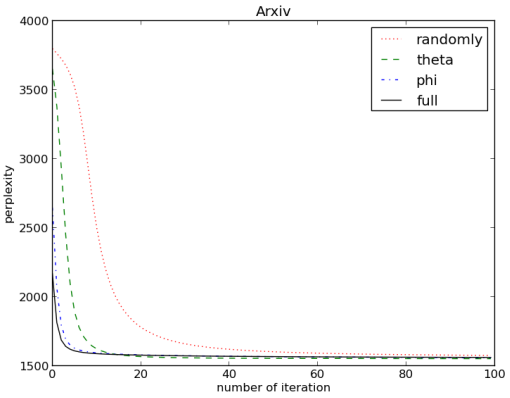

Figure 3: Perplexity for scientific articles

As one can see all the types of initial approximation decrease perplexity of model. Model with initial, finding by our approach converges faster then model with random initial approximation. The same behavior is observed for every data set. Perplexity values eventually obtained in 100 iterations for different datasets and different types of initialization are presented in table 1.

Table 1: Perplexity in 100 iterations

initialization	news articles	tweets	arxiv
random	3006.86	9287.68	1575.58
Θ only	2913.73	9111.91	1551.84
Φ only	2993.90	9143.94	1559.13
Θ and Φ	2973.90	8816.67	1561.07

4.2.2 Computation time depending on initial approximation

In these experiments we evaluate training time depending on initial approximation. We perform iterations until the stop criterion is satisfied: change of perplexity is less than 1 five times in a row. Choosing a threshold is not the aim of this work. Similar results were observed for a wide range of thresholds. Difference in dispersion is less than standard deviation. Results for different datasets and different types of initialization are presented in tables 2, 3 and 4. (It include all training time: training on sample, orthogonalization, finding initial approximation for Θ, training on whole collection)

Table 2: Computation time depending on initial approximation for Twitter dataset

initialization	perplexity	time, sec	#iteration
random	9258.1	7164.9	114
Θ only	9143.5	4131.9	57
Φ only	9213.2	4814.9	67
Θ and Φ	8856.5	3722.6	48

It can be seen that our approach decreases calculation time 1.5-2 times in every data set. Perplexity in models with initial approximation, achieved by our approach is less or equal to the perplexity of model with random initial distribution.

Table 3: Computation time depending on initial approximation for news articles

initialization	perplexity	time, sec	#iteration
random	2996.1	161.5	110
Θ	2946.9	102.3	64
Φ	3020.6	97.5	61
Θ and Φ	2999.4	106.6	67

Table 4: Computation time depending on initial approximation for Arxiv

initialization	perplexity	time, sec	#iteration
random	1592.8	4017.8	59
Θ	1560.9	2413.7	30
Φ	1587.7	2012.9	25
Θ and Φ	1583.7	1893.3	21

4.3 Fixed Φ approach

4.3.1 Perplexity

Perplexity inspection is a common way to compare different topic models [9] [10] [11]. We compare our model with varying size of training subset with PLSA on different datasets. One can see the results on Figures 4, 5 and 6 .

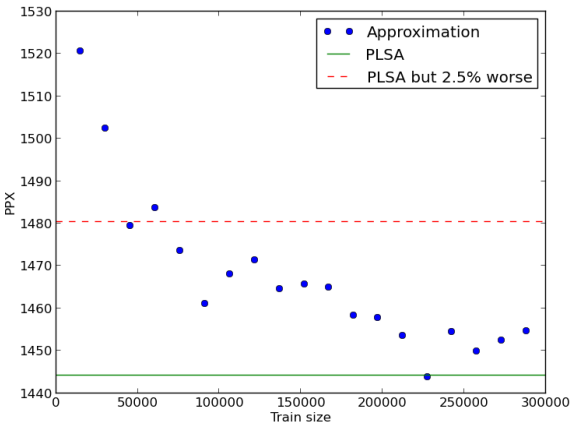

Figure 4: Perplexity depending on training subset size for arxiv dataset

As one can see perplexity values for PLSA and for our approximation are nearly equal, especially for such a large collections as arxiv or Twitter dataset. Worth mentioning that all the perplexity curves have a "horizontal tail" – Φ matrix is not inferred better on a larger sample. It means that the size of a sample needed to infer Φ matrix does not depend on the size of a dataset. This fact is especially significant for training on huge datasets.

4.3.2 Text categorization

The other way to compare different topic models is application task, for example document categorization.
In these experiments we classify news articles by categories and Twitter users by gender treating topic distributions as features. We obtain topic distributions by training PLSA on the whole collection and by our approximation with topics, trained on varying fraction of

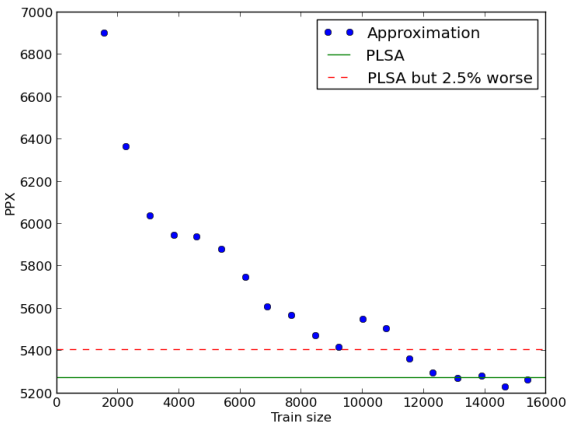

Figure 5: Perplexity depending on training subset size for Twitter dataset

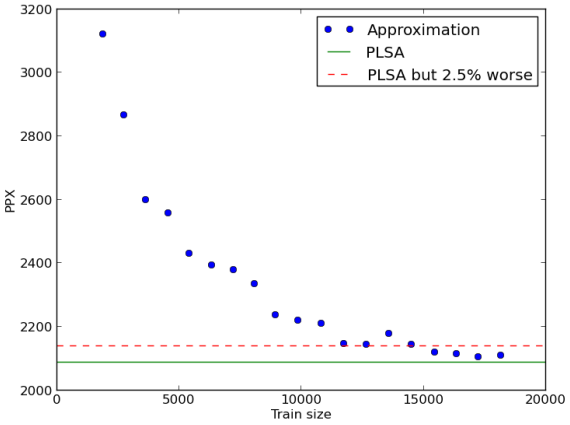

Figure 6: Perplexity depending on training subset size for News articles dataset

the whole collection. Then we estimate the quality of classification by cross-validation with 10 folds. In both experiments we use 20 topics. For classification we use random forest classifier from package scikit-learn [4]. The results can be found on Figures 7 and 8. Majority to minority class ratio for Twitter dataset is 1.17

As one can see our approximation exhibits comparable result if the training dataset is not too small. It works noticeably better for a large collection. Another important observation is that in all the experiments the curve reaches a plateau, it confirms that matrix Φ may be found by training PLSA on a representative subset.

4.3.3 Time efficiency

One of the aims of our work is time efficiency improvement. We compare training time for PLSA and for our approximation with $|D'| = \frac{1}{4}|D|$. In Table 5 calculation time for PLSA and for our approximation (time to train PLSA on the subset is included) are presented.

Another important characteristic is average time to process one document with our approximation and PLSA. Obtained values and speed-up in comparison to

[4] http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

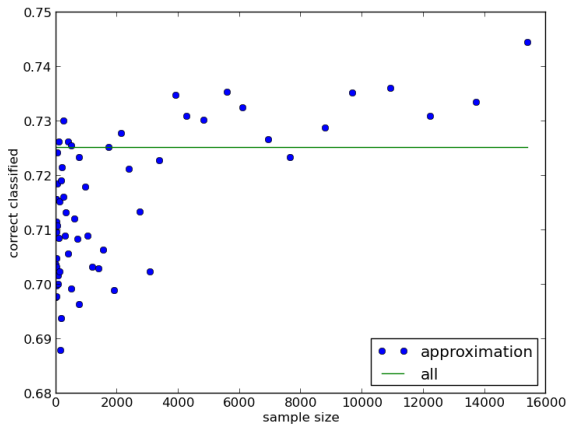

Figure 7: Classification Twitter user by gender

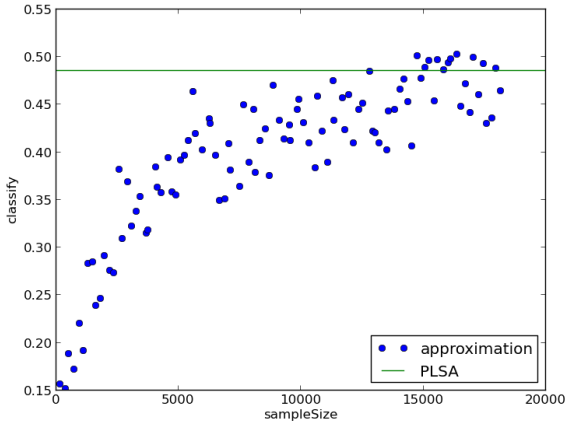

Figure 8: Classification news articles by category

Dataset	PLSA, sec	approx., sec	speed-up, times
Arxiv	4333	1972	2.2
Twitter	6384	2702	2.4
News articles	164	64	2.6

Table 5: Time to process collection

time needed to train original PLSA on a single document in average are given in Table 6 (time to train PLSA on the subset is not included).

Dataset	PLSA, ms	approx., ms	speed-up, times
Arxiv	14.3	3.0	4.8
Twitter	414.2	70.7	5.9
News articles	9.0	0.7	12.9

Table 6: Average time to process one document

5 Conclusion

We develop two methods for computation time reduction one is based on finding appropriate initial approximation and other is based on fixing Φ matrix and tested these methods on three different datasets. Method, based on finding initial approximation demonstrate the same behavior on every used dataset, the calculation time and number of iterations to converge is decreased, yet the quality of topic model does not decrease. We confirm that transition from Kullback-Leibler divergence to L_2 norm is appropriate to find an initial approximation for PLSA.

Method, based on finding initial approximation demonstrate more significant speed-up, but precision is drop. However drop of precision is not significant, especially on large datasets.

References

[1] Timothy N. Rubin, America Chambers, Padhraic Smyth, and Mark Steyvers. Statistical topic models for multi-label document classification. *Mach. Learn.*, 88(1-2):157–208, July 2012.

[2] Cailing Dong and Bin Zhou. Effectively detecting content spam on the web using topical diversity measures. *Web Intelligence and Intelligent Agent Technology, IEEE/WIC/ACM International Conference on*, 1:266–273, 2012.

[3] Venkatesh Ramanathan and Harry Wechsler. phishgillnet–phishing detection methodology using probabilistic latent semantic analysis, adaboost, and co-training. *EURASIP Journal on Information Security*, 2012(1):1, 2012.

[4] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 50–57, New York, NY, USA, 1999. ACM.

[5] Chuntao Hong, Yurong Chen, Weimin Zheng, Jiulong Shan, Yurong Chen, and Yimin Zhang. Parallelization and characterization of probabilistic latent semantic analysis. In *Parallel Processing, 2008. ICPP '08. 37th International Conference on*, pages 628–635, 2008.

[6] Raymond Wan, VoNgoc Anh, and Hiroshi Mamitsuka. Efficient probabilistic latent semantic analysis through parallelization. In GaryGeunbae Lee, Dawei Song, Chin-Yew Lin, Akiko Aizawa, Kazuko Kuriyama, Masaharu Yoshioka, and Tetsuya Sakai, editors, *Information Retrieval Technology*, volume 5839 of *Lecture Notes in Computer Science*, pages 432–443. Springer Berlin Heidelberg, 2009.

[7] Ayman Farahat. F.r.: Improving probabilistic latent semantic analysis using principal component analysis. In *In: Eleventh Conference of the European Chapter of the Association for Computational Linguistics (EACL*, 2006.

[8] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 50–57, New York, NY, USA, 1999. ACM.

[9] Anna Potapenko and Konstantin Vorontsov. Robust plsa performs better than lda. In *ECIR*, pages 784–787, 2013.

[10] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[11] Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. Evaluation methods for topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1105–1112, New York, NY, USA, 2009. ACM.