BOOSTING ARTM WITH PYTORCH

A PREPRINT

Ilya Diyakov
Department of Mathematical Methods of Forecasting
Moscow State University
Moscow
s02210378@gse.cs.msu.ru

Konstantin Vorontsov
Department of Mathematical Methods of Forecasting
Moscow State University
Moscow
vokov@forecsys.ru

December 20, 2024

ABSTRACT

Topic modelling is a fast and efficient technique for text analysis. Currently, the most popular approaches for topic modelling are BPTM (bayesian probabilistic topic models) and ARTM (additive regularization for topic modelling). Recently, a new emerging field, *neural topic modelling*, became an increasingly popular research area. ARTM is a newer approach to this problem, compared to LDA, and it has shown recent success in solving many applied problems in this area. The state-of-the-art model for ARTM is currently BigARTM, which uses expectation-maximization algorithm. In this paper, we propose a different approach for solving the ARTM optimization problem, making use of popular autograd framework PyTorch for efficient implementation.

Keywords Clusterization · Topic Modelling · ARTM

1 Introduction

Topic modelling is a relatively new approach for text analysis and clusterization, which has shown success in many applications in these areas. Approaches emerged during the researches in NLP area also proved useful in a wide variety of problems from another domains like analysis of banking transaction data, image annotation or search for motifs in nucleotide and amino acid sequences. It makes topic modelling an efficient and universal approach for summarization, generalization, and extraction of specific features from unstructured data.

BPTMs, that generates the data from the pre-specified distributions, have been the most popular and successful representation of topic models. However, such approach has several significant drawbacks: **1)** Due to the use of specific pre-defined distributions in each problem, success in one problem cannot be easily generalized to the whole area or transferred to another area. **2)** The solution stability of BPTMs cannot be guaranteed. **3)** The scaling of trained model's inference or making use of parallel computing and GPUs is a challenging task. **4)** There is no modular realization of proposed models, so it is hard to add regularization to an already solved problem.

One of the main advantages of ARTM over BPTM is that its theoretical basis is easy to understand. ARTM successes in creating topic modelling algorithms which can be easily interpreted and scaled. There is also no need for a prior knowledge of the structure of data we are working with. These advantages make it a good alternative for Bayesian approach. Our research is aimed at improving speed and quality of the state-of-the-art model for ARTM, BigARTM. Our new model outperforms BigARTM in speed, maintaining the quality. For better understanding of our new approach, we first outline some theoretical background.

2 Theoretical Background

Let us define the problem ARTM states and overview the solution of this problem proposed in BigARTM model. Then, we will propose a different approach for solving the same problem and compare these solutions.

2.1 The classic problem of topic modelling

Given the collection of documents D with vocabulary W, described as a conditional distribution

$$p(w|d) = \frac{n_{dw}}{n_d} \quad w \in W, d \in D, \tag{1}$$

where n_{dw} is the number of occurrences of the word w in the document d, and n_d is the total number of words in the document d. This distribution can be rewritten as

$$p(w|d) = \sum_{t \in T} p(w|t)p(t|d) = \sum_{t\,T} \phi_{wt}\theta_{td} \quad w \in W, d \in D, \tag{2}$$

where T is the set of topics we want to extract. As can be seen from the equation, the number of topics can be varied and the content of each topic will change accordingly. The aim is to find conditional distributions $p(w|t)$ and $p(t|d)$ in the form of matrices Φ and Θ given the distribution $p(w|d)$.

2.2 ARTM Problem Statement

The proposed problem is the problem of stochastic matrix factorization. For it to have a unique solution, it has to be regularized as follows:

$$\sum_{d,w} n_{dw} \ln \sum_{t} \phi_{wt}\theta_{td} + R(\Phi, \Theta) \to \max_{\Phi,\Theta} \tag{3}$$

Where $R(\Phi, \Theta$ — additive regularization term. This is the classic problem of ARTM. BigARTM authors proposed a solution using expectation-maximization algorithm:

$$\begin{cases} p_{tdw} \equiv p(t|d,w) = \underset{t\in T}{norm} \\ \phi_{wt} = \underset{w\in W}{norm} \left(n_{wt} + \phi_{wt}\frac{\partial R}{\partial \phi_{wt}}\right), \quad n_{wt} = \sum\limits_{d\in D} n_{dw}p_{tdw} \\ \theta_{td} = \underset{t\in T}{norm} \left(n_{td} + \theta_{td}\frac{\partial R}{\partial \theta_{td}}\right), \quad n_{td} = \sum\limits_{w\in d} n_{dw}p_{tdw} \end{cases} \tag{4}$$

Authors have proposed several technical optimizations for the algorithm, but the approach remained the same. Note, that on the maximization step, we have to find partial derivatives of regularization term.

Now, we can notice that the function we want to maximize 3 can be easily optimized directly with any of the autograd frameworks. So our first new approach for solving this problem is to use PyTorch for automatic differentiation. Among other advantages of this approach (see 4), it allows using custom user's regularization functions.

3 Implementation Details

The program provides compatibility with scikit-learn functions and provides the same interfaces for user. We tried to make our implementation easy-to-use for beginners and provide a clear and flexible framework for advanced users.

	Proposed model	BigARTM
Parallel calculation on cpu	No	Yes
GPU enabled	Yes	No
Input types	torch.Tensor, pandas.DataFrame	Vowpal Wabbit, UCI bow

Table 1: Brief comparison of BigARTM and proposed model

It operates with more common torch `Tensor` and pandas `DataFrame` types, unlike BigARTM, which uses `Vowpal Wabbit` or `UCI bow` formats for input data. The main goal of the program is to provide functionality for future research in the field and draw attention of other researchers who needs a fast and reliable software for their work.

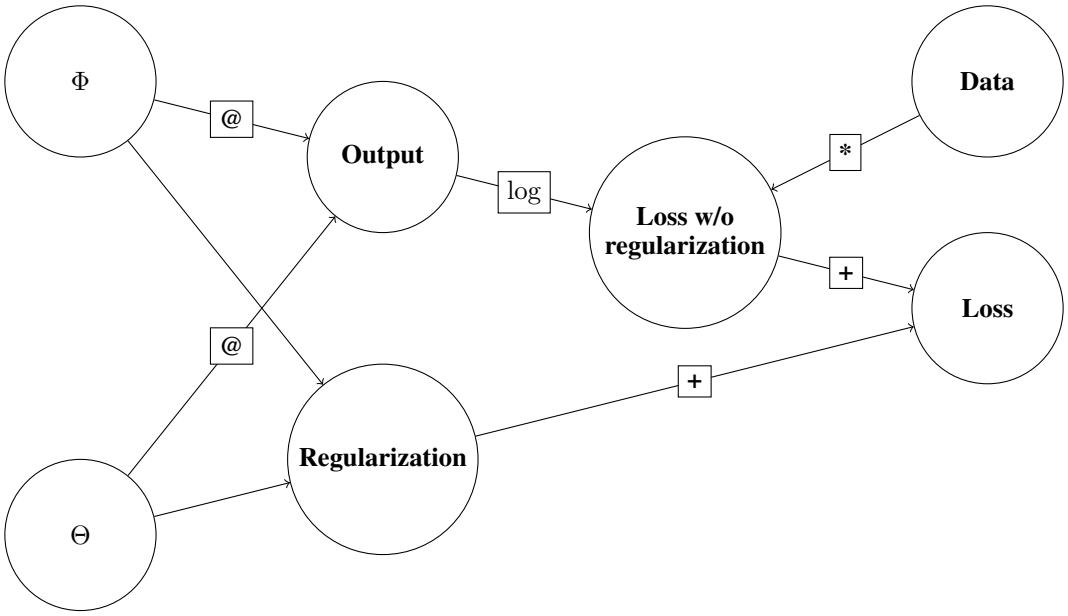

Figure 1: Calculation graph

Our implementation of the proposed algorithm is lighter and easier to maintain, because most of the optimization falls on the PyTorch module. Also, with the support of PyTorch, we can utilize GPU for training. This makes inference much faster compared to BigARTM 4.

4 Experiments

4.1 Data

We evaluate the proposed model on the 20newsgroups dataset, which contains a collection of approximately 20,000 newsgroup documents, partitioned nearly evenly across 20 different newsgroups. We extracted approximately 70,000 unique words as vocabulary in bag-of-words format.

4.2 Quantitative Results

We report our quantitative results in Table 2, training both models on 20 steps. Both models were compared without regularization (PLSA). To measure topic similarity, we used the Hungarian algorithm to find similar topics using cosine metric between top-3 words as distance between topics. Then we calculated the mean distance across all distances between the most similar topics. All metrics were aggregated over 100 runs.

Metric	Ours, 10 topics	BigARTM, 10 topics	Ours, 120 topics	BigARTM, 120 topics
Perplexity@10step	3750	2600	2780	1170
Perplexity@20step	2800	2520	1310	1120
Perplexity@30step	2670	2500	1190	1110
Training time CPU (30 steps, sec)	400	30	1400	130
Training time GPU (30 steps, sec)	40	—	40	—
Topic similarity	0.65		0.4	

Table 2: Quantitative results training BigARTM and proposed models

Overall, our implementation shows very similar results with BigARTM, but successes in speeding up the inference with GPU. For both models, only 20 steps are sufficient to generate interpretable topics from the given data. Note that both models are producing many similar topics, which shows that the algorithm's principle remains the same.

The results show that our model's training time cannot compete with efficient CPU implementation of BigARTM, but on GPU it gains better results. Training time on GPU does not depend on number of topics because of efficient parallelization. This demonstrates the potential of the proposed model in large datasets.

4.3 Qualitative Results

Here, we analyze our results in terms of top k words, representing the topic, focusing on models with 10 topics. These particular examples demonstrate the same logic of clustering in both models.

BigARTM, top-5 topic words

topic 1: max, q, r, g, p
topic 2: one, would, say, people, write
topic 3: game, team, line, subject, organization
topic 4: would, people, write, gun, article
topic 5: god, hell, atheist, line, subject
topic 6: x, file, use, window, program
topic 7: say, armenian, people, one, go
topic 8: line, subject, get, organization, car
topic 9: space, organization, subject, line, db
topic 10: line, subject, organization, use, university

Ours, top-5 topic words

topic 1: max, q, r, g, p
topic 2: line, one, get, subject, use
topic 3: god, would, say, one, people
topic 4: line, subject, organization, university, article
topic 5: game, team, line, drive, subject
topic 6: say, armenian, one, go, people
topic 7: x, file, line, use, subject
topic 8: use, line, subject, organization, window
topic 9: would, people, write, get, article
topic 10: use, key, system, data, space

5 Conclusion

The proposed model seems to be a promising direction of development, offering accessibility and lightness to the user, the ability to flexibly configure system components and integrate custom regularizers. One of the most important advantages of the developed program is its efficiency. Also, the proposed algorithm can become the first step of ARTM's integration with deep neural networks. This can solve several crucial problems of modern neural networks. Among them, we can highlight the selection of the learning rate (ARTM does not need this parameter, 2) as well as the interpretability of the inner layers as a set of «topics», which model extracts from data.