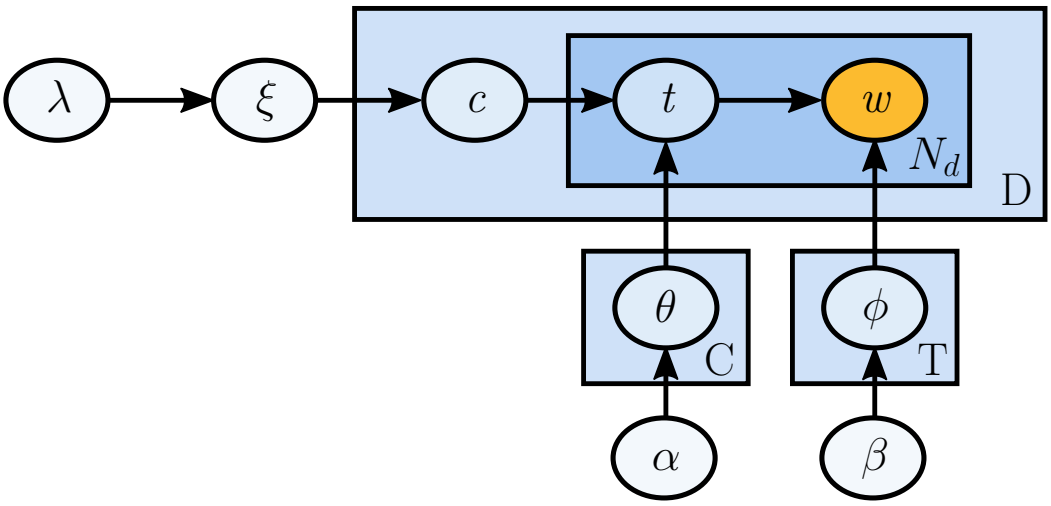
CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG



Topic Analysis for Community Detection

Exploring Topic Models for Community Detection on Social Media

Master's thesis in Applied mathematics

ALGOT JOHANSSON, ERIC GULDBRAND

Department of Mathematical Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2021

Master's thesis 2021

Topic Analysis for Community Detection

Exploring Topic Models for Community Detection on Social Media

ALGOT JOHANSSON
ERIC GULDBRAND

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2021

Topic Analysis for Community Detection
Exploring Topic Models for Community Detection on Social Media
ALGOT JOHANSSON, ERIC GULDBRAND

Supervisor: Johan Jonasson, Department of Mathematical Sciences
Examiner: Petter Mostad, Department of Mathematical Sciences

Master's Thesis 2021
Department of Mathematical Sciences
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Plate notation for one of the proposed and investigated models, LDAC.

Typeset in LaTeX
Gothenburg, Sweden 2021

Topic Analysis for Community Detection
Exploring Topic Models for Community Detection on Social Media
ALGOT JOHANSSON, ERIC GULDBRAND
Department of Mathematical Sciences
Chalmers University of Technology and University of Gothenburg

Abstract

Being able to detect communities in social networks can be an aid in understanding trends, assist moderation efforts and build recommendation systems. In this paper we explore the use of topic models for community detection by proposing two such models, LDAC and LDACS, based off of Latent Dirichlet Allocation (LDA) [1] and the Community Topic Model [8]. These models are compared to LDA and evaluated on datasets collected from Twitter and Reddit. It is concluded that LDACS may be a reasonable and simple model for community detection, but with further study needed, and that LDAC gives some credence to utilizing both topics and communities in a model, but does itself not produce sufficient results to weigh up for its complexity, although training it on more data might remedy this.

Keywords: topic analysis, community detection, community, topic, thesis, lda, ldac, ldacs, ctm.

Acknowledgements

Many thanks to our supervisor Johan Jonasson for his invaluable feedback and help throughout the entire process of writing this thesis. Thanks also to our examiner Petter Mostad for his interest and great advice.

Algot Johansson and Eric Guldbrand, Gothenburg, June 2021

Contents

List of Figures

List of Tables

1
Introduction

Social networks like Twitter are widely used for communicating and interacting with a large number of people. It would be useful to, based on the written messages on these platforms, be able to gain insight into the social network and the communities therein. Such insight could help understand current trends, assist moderation efforts, improve recommendation systems and help counteract the harmful effects of echo chambers or message bots.

Traditional community detection methods have often been link- and graph-based [5], but more recent studies have made use of topic models [8] or the combination of graph and topic modeling[10]. These studies have shown some success, demonstrating that topic modelling is a viable technique for community detection, but that further exploration of the subject may be useful.

1.1 Purpose

The purpose of this paper is to further explore topic analysis for community detection. This will be done by building and evaluating two such models on posts from two social networks: Twitter and Reddit.

1.2 Problem Definitions

First, propose and implement an extension to Latent Dirichlet Allocation (LDA, see Section 2.2). Since simpler models often generalize better with less data [17], this extension should be a simplified version of the Community Topic Model (CTM) [8], but which has topics for each word, communities for each document and does not consider authors. This way, the model will be able to label each document as part of a community and each word as part of a topic.

Second, propose and implement an even simpler model that ignores topics entirely, but still has a notion of community. This means it will be able to label each document but not each word. How well this model performs compared to the first may give indication of how worthwhile more complex models are under our circumstances.

Third, evaluate both models and compare them to LDA. Since this is a clustering problem there are no true answers to measure against. Thus the evaluation will be done partly by visual inspection to see if the model output makes any sense, partly by analyzing model consistency and stability.

Finally, to get around the lack of true answers to compare against, Reddit data will be used as a stand-in for real community labels, to see how well the models'

definition of community compares with that of Reddit.

1.3 Limitations

The models developed in this paper are exclusively of a bag-of-words type. This means that the order of words in each document is not taken into consideration, which leads to a loss of information. For example: ”this is good” and ”is this good” will be considered the same to the model.

Another loss of information is that stop words are removed. This means that both ”happy” and ”not happy” will only register that the word ”happy” has been used, since ”not” is considered a stop word.

Furthermore, our models do not use any traditional link- and graph-based community detection algorithms [5] or the explicit combination of graph and topic modelling approaches [10].

Neither are likes, retweets or similar taken into account for improving the model. While this is very interesting data to consider, especially as retweets have been found to spread information rather quickly[7], this type of data does not seem to be a natural fit for a topic model approach, and would require special attention.

Finally, the models are trained on our personal computers which means that running training for multiple days at a time are difficult to do. This limits the amount of data we are able to use.

1.4 Data

Datasets from two social media platforms, Twitter and Reddit, will be used.

On Twitter, users can write posts using up to 280 characters. Each such post is called a tweet. Tweets are public to anyone who visits a user’s profile, but are also shown in the feeds of users who ”follow” the poster. Each tweet may include one or several hashtags.

A hashtag is any word directly preceded by a hash symbol (#). These let the user mark tweets as belonging to a certain topic, which makes it easier to find tweets on the same topic[3]. As a result, trending topics are often accompanied by a certain hashtag [8]. However, users may not always be consistent with their hashtag usage, and some users may deliberately use trending hashtags on unrelated tweets in an effort to gain more exposure [8].

In addition, Twitter user A may ”retweet” another user B’s tweet. This makes that tweet show up to everyone following user A as well. Users can also ”like” tweets. Both retweets and likes can be seen as measures of how many other users agree with or otherwise think that a tweet should be seen by more people.

The Twitter dataset collected consists of 550k tweets, although only 80k is used for training, each tweet containing at least three words. All tweets have been preprocessed as to be lowercase, have contractions expanded and stopwords (NLTK’s english stopwords[12]), urls, hashtags and all other non-letter characters except space removed.

On Reddit, a web forum, users submit posts to sub-forums called ”subreddits”. Posts can be seen by anyone who visits that subreddit, but subreddits can also be favorited by users, allowing its posts to show up in the users' feeds. Each subreddit is meant as a community for talking about a particular set of topics. Each post typically consists of a title and a link or image, and other users can comment on the post itself (top-level comment) as well as on other users' comments. Users can also up- or down-vote both posts and comments. The total number of up-votes minus down-votes is used as a measure of how popular it is, and these posts and comments are placed more visibly.

The Reddit dataset consists of the 1000 most popular posts on 6 different subreddits. Each post title is considered a document, and so are the (up to) 15 most up-voted top-level comments for each post. In the dataset, a max-length of 500 characters was set per document to reduce model training time, as a few documents were very long.

1.5 Risk Analysis and Ethical Considerations

In this study we collect a fairly large amount of Twitter data. While all this data are publicly posted messages, it is still important to use and handle such datasets responsibly. As such we have taken care in handling the data and not disclosing details like usernames.

Another consideration is for how tweets are collected. Twitter has an official API for collecting tweets, but this only allows collecting tweets from the last week. But since the tweets we are interested in are all public, several third-party libraries exist for collecting tweets from further back. While such libraries can be very convenient, they scrape the Twitter website to collect data, a practice which is sometimes frowned upon[16]. Therefore we have instead applied for and received academic access to the official Twitter API, allowing us to collect more data and from a wider time span without having to use scraping-based tools.

Identifying communities online can have many benefits such as recommending relevant content for users, gain insight into problematic social network phenomenon like the existence of echo chambers or help keep an eye on users commonly involving themselves in pro-violence communities or similar.

Unfortunately each of these benefits may, as most things, instead be used to bring harm. Content recommendation has potential to create echo chambers in the first place, insight in social network structures can be used to manipulate and control opinion or flow of information, and finding users with particular interests could be misused to target political, religious or other minorities.

However, since this is an unsupervised approach, it's impossible to say beforehand what types of communities will be found, or if people are grouped into communities they are not comfortable with. Thus it can be valuable to investigate whether or not the kinds of communities mentioned above can at all be identified by topic analysis. This could help provide insight to what degree the things people say online may be used to classify them.

2
Theory

This section introduces and explains the theoretical concepts relevant to this paper. However, the reader is assumed to have a reasonable understanding of Bayesian statistics beforehand.

A topic model is a statistical model for identifying abstract topics from text. The topics are defined by the model during training as a collection of words that are more likely to appear together. For instance, the words ”car”, ”pedestrian” and ”lane” may all be words associated with a topic which a human might label ”traffic”.

This and the following sections uses a lot of notation. For summary of all such notation, see Table 2.1.

2.1 Dirichlet Distribution

The Dirichlet distribution (Dir) is the probability distribution LDA models are based on and is a random probability vector $\boldsymbol{p} = (p_1, p_2, ..., p_K)$ chosen from the density

$$Dir(\boldsymbol{p}|\boldsymbol{\alpha}) \propto_{\boldsymbol{p}} \prod_{k=1}^{K} p_k^{\alpha_k - 1}, \tag{2.1}$$

where $\alpha_1, \ldots, \alpha_K > 0$ are the parameters and $\sum_{k=1}^{K} p_k = 1$. For an example of sampling from Dirichlet distributions with different $\boldsymbol{\alpha}$, see Figure 2.1. Note that each sampled point $\boldsymbol{p}$ can be seen as a distribution itself. Thus the Dirichlet distribution can be seen as a ”distribution of distributions”.

For the multinomial distribution (Multi) the probability of choosing a vector $\boldsymbol{m} = (m_1, m_2, ..., m_k)$ is given by:

$$Multi(\boldsymbol{m}|\boldsymbol{p}) = \frac{n!}{\prod_{k=1}^{K} m_k!} \prod_{k=1}^{K} p_k^{m_k} \tag{2.2}$$

As can be seen, the Dirichlet distribution is conjugate prior to multinomial distribution, so we get

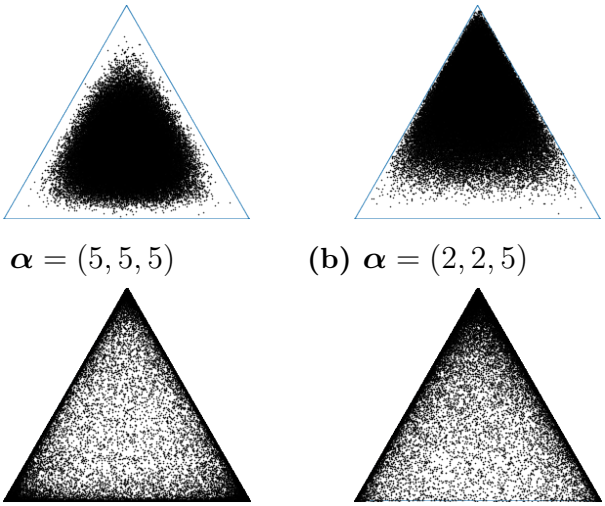
$$P(\boldsymbol{p}|\boldsymbol{m}) \propto_p P(\boldsymbol{m}|\boldsymbol{p})P(\boldsymbol{p}) = Multi(\boldsymbol{m}|\boldsymbol{p})Dir(\boldsymbol{p}|\boldsymbol{\alpha}) \propto_{\boldsymbol{p}} Dir(\boldsymbol{p}|\boldsymbol{\alpha} + \boldsymbol{m}) \tag{2.3}$$

2.2 Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation[1] (LDA) is an unsupervised clustering model that, given documents and a set of topics, determines how much each document belongs to each

Table 2.1: Notation Table. Summary of all model notation.

Notation	Meaning
D	Nr of documents
A	Nr of authors
T	Nr of topics
W	Nr of unique words in all documents
C	Nr of communities
d	Document
w	Word
i	Index of a word in a document
x	Author
t	Topic
c	Community
N_d	Number of words in d
a_d	Author of d
α	Hyper parameter for generating θ
λ	Hyper parameter for generating ξ
β	Hyper parameter for generating ϕ
μ	Hyper parameter for generating ψ
ξ	Distribution of communities
θ	Distribution of topics
ϕ	Distribution of words
ψ	Distribution of publication venues
$n_d(t)$	Nr of times topic t occurs in document d
$m_t(w)$	Nr of times word w is classified as topic t
$l_c(t)$	Nr of times a topic t occurs in community c
$p(c)$	Nr of times a document is classified as c
$o_c(w)$	Nr of times a word w occurs in topic c
$q_d(w)$	Nr of times word w is in document d



(a) $\boldsymbol{\alpha} = (5, 5, 5)$ **(b)** $\boldsymbol{\alpha} = (2, 2, 5)$

(c) $\boldsymbol{\alpha} = (0.1, 0.1, 0.1)$ **(d)** $\boldsymbol{\alpha} = (0.1, 0.1, 0.8)$

Figure 2.1: Illustration of 100 000 samples drawn from Dirichlet distributions with different $\boldsymbol{\alpha}$. Each dot is one sample. Samples ($\boldsymbol{p}$) are the most likely to occur in different regions depending on $\boldsymbol{\alpha}$. As can be seen, when the alphas are equal and bigger they push the samples away from the corners, but when one alpha is bigger than the others, the corresponding corner get the dots closer. Note that each corner represents one of the following values: $\boldsymbol{p} = (1, 0, 0)$, $\boldsymbol{p} = (0, 1, 0)$ or $\boldsymbol{p} = (0, 0, 1)$. (Images generated with [2].)

topic, based on which words appear in the document. The number of possible topics is pre-determined, but the content of each topic is learnt by the model. Since LDA is a bag-of-words model, the order in which words appear in the document is not taken into consideration.

The model uses two sets of distributions: $\theta \in \mathbb{R}^{D \times T}$ and $\phi \in \mathbb{R}^{T \times W}$, where θ_d is how each document d is distributed over all topics and ϕ_t how each topic t is distributed over all words.

LDA's **generative process** is an assumption on how the documents are generated. Figure 2.2 and Algorithm 1 both show this process for LDA. Note that the generative process is just an assumption, we are not interested in actually running it. This can be compared to how linear regression assumes that all its data points have been generated along a straight line.

After choosing the hyperparameters α and β, the distributions are assumed to be generated the following way. For each document d it is assumed that the number of words in the document N_d is fixed and that the distribution over topics θ_d is generated from $\text{Dir}(\boldsymbol{\alpha})$ where $\boldsymbol{\alpha} = (\alpha, \alpha, ..., \alpha) \in \mathbb{R}^T$ and T is the pre-determined number of topics. $\phi_{t_{di}}$ is generated from $\text{Dir}(\boldsymbol{\beta})$ where $\boldsymbol{\beta} = (\beta, \beta, ..., \beta)$, $|\boldsymbol{\beta}| = W$, and W is the number of unique words in all documents combined.

Now the variables are assumed to be generated the following way. For each of the $i = 1...N_d$ word positions in document d, choose a topic $t_{di} \sim \text{Cat}(\theta_d)$. Then choose a word w_{di} from the topic-word distribution $\phi_{t_{di}}$. This generates document d.

Note that the hyperparameter α is used to generate the topic distributions and β for generating the word distributions. If $\alpha = \beta = 1$ the probability distributions

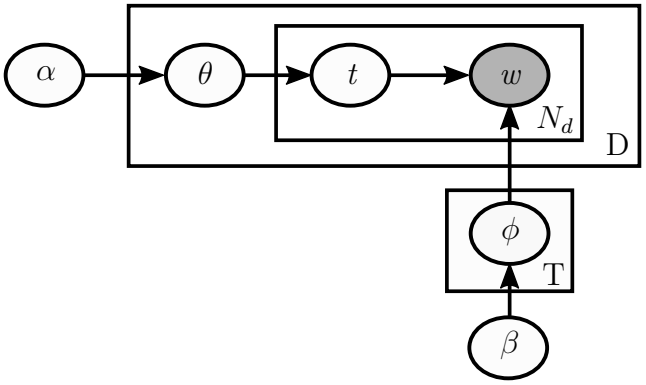


Figure 2.2: Plate notation of the LDA model, describing LDA's generative process. Here θ is the document-topic distribution, ϕ is the topic-word distribution, t is a topic, w is a word, N_d is the number of words in document d, D is the number of documents, T is the number of topics, α and β are hyperparameters. There are D different θ_d and T different ϕ_t, each considering a document with N_d words, in which each word w is generated by its topic t's distribution ϕ_t.

Algorithm 1 Generative process of LDA.

for document d in documents **do**
  $\theta_d \sim Dir(\alpha)$
end for
for topic t in topics **do**
  $\phi_t \sim Dir(\beta)$
end for

for document d in documents **do**
  for word position j in document d **do**
    Set topic $t_{dj} = t$ with probability θ_{dt}
    Set word $w_{dj} = w$ with probability $\phi_{t_{dj}w}$
  end for
end for

θ and ϕ will be generated uniformly. If $\alpha = \beta > 1$ the distributions will be more similar and if $\alpha = \beta < 1$ the distributions will be less evenly distributed [18]. See Figure 2.1.

2.3 Community Topic Model (CTM)

The Community Topic Model (CTM) [8] is a model based on LDA that can identify communities from user created documents. It assigns each person to a community by finding similarity between a persons topic distribution and the topic distribution of a community.

This model was used to detect communities in the scientific world by analysing papers, citations and conferences, but it was also applied to Twitter.

To understand the model, it is helpful to refer to its plate notation in Figure 2.3.

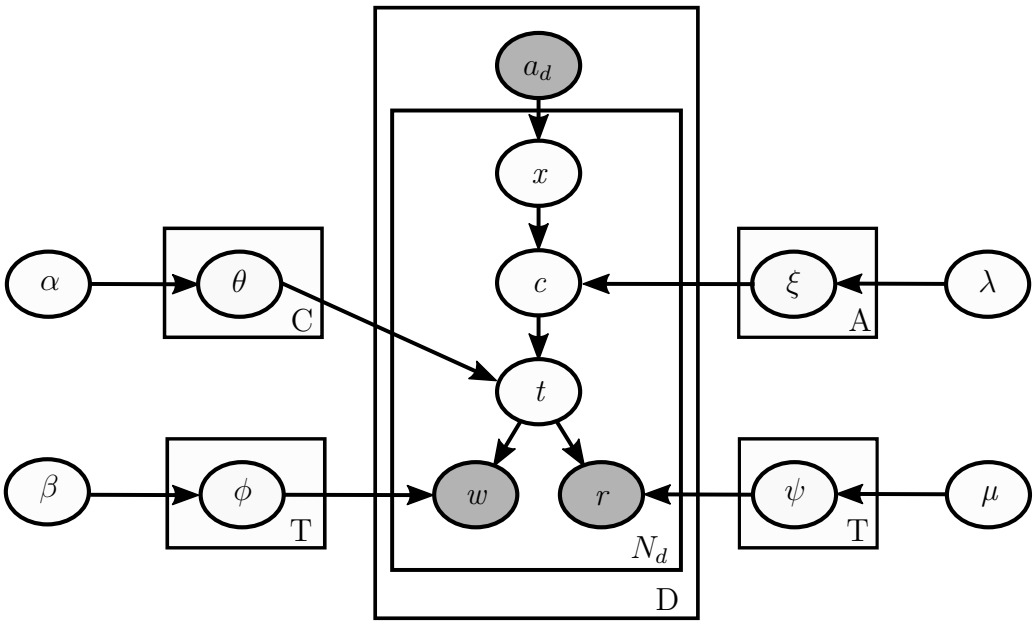


Figure 2.3: Plate notation for the Community Topic Model[8] (CTM). As described in the original paper[8], a group of authors a_d is collaborating on document d. Each author x in a_d selects a community c from the author-community distribution ξ, then selects a topic t from the community-topic distribution θ_c. Finally, each author selects a word w from the topic-word distribution ϕ_t and a conference r related to t from the topic-conference distribution ψ_t.

As described in the original paper[8], the model has the following generative process. A group of authors a_d is collaborating on document d. Each author x in a_d selects a community c from the author-community distribution ξ, then selects a topic t from the topic-community distribution θ_c. Finally, each author selects a word w from the topic-word distribution ϕ_t and a conference r related to t from the topic-conference distribution ψ_t.

2.4 Gibbs Sampling

Gibbs sampling is a general Markov chain Monte Carlo method. It works as follows. Assume you want to sample $\boldsymbol{X} = (x_1, x_2, ..., x_n)$ from a distribution $P(x_1, x_2, ..., x_m)$ which is hard to sample directly from. Instead we can sample one variable at a time conditioned on the other variables. That is, every iteration samples all the variables in order, conditioned on each other, as in Algorithm 2.

Algorithm 2 Gibbs Sampling

```
for number of iterations do
  for j ∈ {1, ..., n} do
    sample x_j from p(x_j|x_1, ..., x_{j-1}, x_{j+1}, ..., x_n).
  end for
end for
```

It is also possible to use **collapsed Gibbs sampling** where you integrate out one or more variables and make an inference sample from the marginal distribution of the remaining variables. For example, if you integrate out x_n in Algorithm 2 you would get Algorithm 3. This can be done to improve computation speed in some cases.

Algorithm 3 Collapsed Gibbs Sampling

for number of iterations **do**
  for $j \in \{1, ..., n-1\}$ **do**
    sample x_j from $p(x_j|x_1, ..., x_{j-1}, x_{j+1}, ..., x_{n-1})$.
  end for
end for

2.5 LDA Inferring Distributions and Topics

We want to use Gibbs sampling to sample the topics. To do this we first fix the topics t and sample θ and ϕ, then we fix θ and ϕ and sample t.

When sampling t we use the probability

$$\begin{aligned} &p(t_{dn} = t|w_{dn} = w, \theta, \phi) \propto \\ &p(w_{dn} = w|t_{dn} = t, \theta, \phi)p(t_{dn} = t|\theta, \phi) = \\ &p(w_{dn} = w|t_{dn} = t, \phi)p(t_{dn} = t|\theta) = \\ &\theta_d(t)\phi_t(w). \end{aligned} \tag{2.4}$$

and when sampling θ we use the distribution

$$\begin{aligned} &p(\theta|t) \propto \\ &p(t|\theta)p(\theta) = \\ &Dir(\alpha + n_d(1), ..., \alpha + n_d(T)) \end{aligned} \tag{2.5}$$

where $n_d(t)$ is how many words are set to topic t in document d.

For ϕ we use the same logic and get $\phi_t \sim Dir(\beta + m_t(1), ..., \beta + m_t(W))$, where $m_t(w)$ is the number of times word v has been assigned topic t.

2.6 Rand Index

Rand index[15] measures how similar two data clusterings are. Assume that we have a set of elements S and two partitionings of S: $X = \{A_1, ..., A_N\}$ and $Y = \{B_1, ..., B_N\}$. We can then define the following:

- a, number of pairs of elements that are in the **same** cluster in X, and in the **same** cluster in Y.
- b, number of pairs of elements that are in **different** clusters in X, and in **different** clusters in Y.

- c, number of pairs of elements that are in the **same** cluster in X, but in **different** clusters in Y.
- d, number of pairs of elements that are in **different** clusters in X, but in the **same** cluster in Y.

The Rand index R can now be defined as

$$R = \frac{a+b}{a+b+c+d}.$$

As such the Rand index is much like an accuracy score, but where it doesn't matter which cluster an element is in, as long as the other elements in that cluster are also the same. If $X = Y$ then $R = 1$.

A variant of Rand index is the Adjusted Rand Score [15, 13] which is a Rand index that is adjusted for chance such that two random clusterings should give on average $R = 0$.

3
Methods

This chapter describes and motivates our data sets and models.

3.1 Selecting Data

Data was primarily collected from Twitter, but as a comparison we also looked at data from the webforum Reddit. This section describes how the data was collected.

3.1.1 Twitter

It has been estimated that Twitter produces 500 million tweets a day[6]. As such, using all tweets is infeasible just from the data size alone. The question is then, how should the data be selected?

Twitter limits the number of tweets one can pull from their API, so to collect a large number of tweets we applied for an academic API license. This procedure was fairly quick, without cost and allowed for collecting 10 million tweets a month. This turned out to be more than sufficient for our purposes.

Trending topics were collected from the `trending` API endpoint. This endpoint lists the 30 most popular topics for the last 24 hours, each topic being a word, a name or a hashtag. Since we were initially interested in being able to relate each tweet to a hashtags, we collected a list of hashtag by querying this endpoint once a day until we had enough unique hashtags. The only query parameter to this endpoint is location, to find what is trending in a specific country, city or worldwide. We decided to only look at the US and the UK in an attempt to get hashtags most likely to be associated with English tweets.

The reason we were initially only interested in hashtags was to be able to use them as a proxy for true data labels, which was done by [8], but this idea was abandoned in favour of looking at Reddit data in addition to Twitter data. This was in large part due to the realisation that there are often several hashtags related to a single community or topic.

Tweets could then be queried from the `tweets/recent` or `tweets/all` endpoints. The first only allows access to the last week, whereas the `tweets/all` allows querying any past date range, but requires an academic license to access. This endpoint allows pulling up to 500 tweets per second, and thus collecting a few hundred thousand tweets can be done in under an hour.

However, pulling tweets requires a query with at least one key word, which is why collecting trending hashtags and using them as keywords seemed like a reasonable

choice. In addition, we limited our queries to only collect English tweets and no retweets, replies or quotes. The reason was to not confuse our models with multiple languages, and because previous studies have excluded retweets[8]. We decided to also exclude replies and quotes (a mix between retweet and reply) to try looking at only 'one kind' of data.

All tweets were first saved to a json format similar to the raw API data, to make it easy to keep an archive of all information that might be relevant. However, to significantly reduce the size of the working dataset, a csv file was created containing only the relevant data.

Tweet texts in the csv file were also pre-processed. First this meant to remove stop words: common words like 'and', 'I', 'be', etc. that provide very little meaning in a bag-of-words model, but which are also so common that they would dominate the word lists for most topics if not removed. We used the same list of stop words as NLTK [12], a well-known natural language processing library for Python. Second in the pre-processing step, hyperlinks and hashtags (hash symbol and word) were removed. It was especially helpful to do this in connection to the original json representation, as that data is already annotated with what parts of each tweet is a link, hashtag etc. Thirdly, all words were normalized by being set to lowercase, expanding all contractions and removing all punctuation. Finally, all tweets that contained less than 3 words in their pre-processed form were removed.

Of the collected 680 thousand tweets, 550 thousand remained after pre-processing. This was more than enough for our purposes and more would only make the dataset more cumbersome.

3.1.2 Reddit

We also decided to look at posts and comments from the webforum Reddit. On Reddit every post is uploaded into a sub-forum called a subreddit, each of which has a particular subject of discussion. We wanted to treat each subreddit as a "real" community label, and compare our results to those to see if our models' definitions of community would be similar to that of Reddit.

Data was collected from six subreddits: "machinelearning", "linux", "programming", "books", "stocks" and "funny", using The Python Reddit API Wrapper (PRAW)[14]. For each subreddit the 1000 highest ranked posts were collected (or rather the post titles, because only a small number of reddit posts have text content, but the titles are often quite long), along with (at most) 15 top-level comments. Then the data was cleaned and pre-processed in much the same way as the Twitter data, except that documents with more than 500 characters were trimmed to roughly 500 characters (without cutting any word short) as to not have documents that take too much time to process.

The reason for collecting post comments as well as the posts themselves is that the comments could be related to their post in two ways. Either the comments of each post is considered documents with no relation to their parent post, except from being in the same subreddit. This results in more but shorter documents. We call this data set **Reddit-Many**. Otherwise the comments can be seen as extensions of their post. That is, their words were appended to the post itself. This results

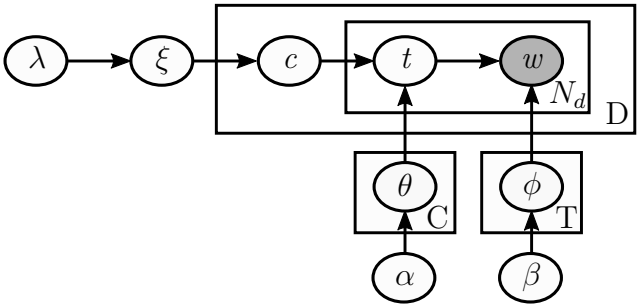


Figure 3.1: Plate notation for the LDA + Community (LDAC) model, describing its generative process. First, a community distribution ξ is sampled from $Dir(\lambda)$. Then a topic-community distribution θ_c is sampled from $\text{Dir}(\alpha)$ for each community c and the word-topic distribution ϕ_t is sampled from $\text{Dir}(\beta)$ for each topic t. For each document d, a community c_d is sampled from ξ. For each of the N_d word positions $n \in \{1, ..., N_d\}$ in d, a topic t is selected from θ_c and a word $w_{n,d}$ is selected from ϕ_t.

in fewer but longer documents. We call this dataset **Reddit-Long**. (We primarily considered Reddit-Many, but also compared the two, see 4.2.4.)

3.2 Defining the LDAC Model

We will call our main model of interest the LDA + Community (LDAC) model (see Figure 3.1), based on LDA, with inspiration from CTM[8], but somewhat simplified. This model tries to predict what community a tweet belongs to by assuming that each tweet is written by a community, not an actual user. As such it is assumed that a community generates topics from a distribution over all topics, each of which generates a word from the distribution ϕ.

Gibbs sampling is used to train the model. To make this process more efficient, collapsed Gibbs sampling is used.

3.2.1 Generative Process

The model assumes that all documents it sees are generated in a particular way, its generative process. For LDAC the generative process is described by Algorithm 4 and Figure 3.1, and is as follows.

We have three distributions: the community distribution $\xi \in \mathbb{R}^C$, the topic-community distribution $\theta \in \mathbb{R}^{C \times T}$ and the word-topic distribution $\phi \in \mathbb{R}^{T \times W}$. Begin by sampling $\xi \sim Dir(\boldsymbol{\lambda})$. Then sample $\theta_c \sim \text{Dir}(\boldsymbol{\alpha})$ for each community c. Now sample $\phi_t \sim \text{Dir}(\boldsymbol{\beta})$ for each topic t. Here $\boldsymbol{\lambda} = (\lambda, \lambda, ..., \lambda) \in \mathbb{R}^C$, $\boldsymbol{\alpha} = (\alpha, \alpha, ..., \alpha) \in \mathbb{R}^T$ and $\boldsymbol{\beta} = (\beta, \beta, ..., \beta) \in \mathbb{R}^W$ are hyperparameters. Now, for each document d, a community c_d is sampled from ξ. For each of the N_d word positions $n \in \{1, ..., N_d\}$ in d, a topic t is selected from θ_c and a word $w_{n,d}$ is selected from ϕ_t. The process concludes when a word has been selected for every word position of every document.

Algorithm 4 LDAC Generative Process

$\xi \sim \text{Dir}(\lambda)$
for community c in communities **do**
  $\theta_c \sim Dir(\alpha)$
end for
for topic t in topics **do**
  $\phi_t \sim Dir(\beta)$
end for
for document d in documents **do**
  Set community $c_d = c$ with probability ξ_c
  for word position j in document d **do**
    Set topic $t_{dj} = t$ with probability $\theta_{c_d t}$
    Set word $w_{dj} = w$ with probability $\phi_{t_{dj} w}$
  end for
end for

3.2.2 Inferring Distributions and Variables

To find out which distributions would be most likely to generate the training data through the generative process, we use Gibbs sampling to, one at a time, sample values of communities, topics, θ, ϕ and ξ, each time given the previously sampled values of the other variables. This process is similar to Gibbs sampling for LDA.

Algorithm 5 LDAC Gibbs Sampling

for number of iterations **do**
  $\xi \sim Dir(\lambda + p(1), ..., \lambda + p(C))$
  for community c in communities $1...C$ **do**
    $\theta_c \sim Dir(\alpha + l_c(1), ..., \alpha + l_c(T))$
  end for
  for topic t in topics $1...T$ **do**
    $\phi_t \sim Dir(\beta + m_t(1), ..., \beta + m_t(W))$
  end for

  for document d in documents **do**
    $c_d = c$ with probability proportional to $\prod_{t=1}^{T} \theta_{c,t}^{n_d(t)}$
    for word position j in document d **do**
      $t_{dj} = t$ with probability proportional to $\theta_{c_d t} \phi_{t w_{dj}}$
    end for
  end for
end for

As can be seen in Algorithm 5, for each iteration of the Gibbs sampling process, ξ is a distribution sampled from $Dir(\lambda + p(1), ..., \lambda + p(C))$ That is, a Dirichlet distribution with as many parameters as there are communities (C) where $p(c)$ is the number of times a document has been classified as community c.

Similarly, θ_c is a distribution sampled from a Dirichlet distribution with as many

parameters as there are topics, each parameter created from α and $l_c(t)$, the number of times topic t is in a document classified as community c. In just the same way, ϕ_t is a distribution sampled from a Dirichlet distribution with as many parameters as there are words in the dictionary, each parameter created from β and $m_t(w)$, the number of times word w is in topic t.

Now, for each document d in the training data, set the document's community c_d to the community c with probability proportional to $\prod_{t \in T_d} \theta_{c,t}^{n_d(t)}$ where T_d is the topics in d. In addition, for each word w in d, set the word's topic t_{dw} equal to t with probability proportional to $\theta_{ct}\phi_{tw}$.

With sufficiently many iterations, ξ, θ and ϕ will approximate the distributions assumed in the generative process for the training data documents, since the Gibbs sampling was conditioned on them; c is the communities for the documents and t is the topics for the words.

3.2.3 Collapsed Gibbs

We wanted to attempt using collapsed Gibbs to learn if that would result in faster computation or better results. To find the topic probability while collapsing the θ and ϕ calculations we need to sample directly from

$$p(t|w,c) \propto_t p(w|t)p(t|c) = E\left[p(w|t,\phi)\right] E\left[p(t|c,\theta)\right] \tag{3.1}$$

where

$$E\left[p(w|t,\phi)\right] = \prod_{t=1}^{T} E\left[\prod_{w=1}^{W} \phi_t(w)^{m_t(w)}\right] \tag{3.2}$$

$$E\left[p(t|\phi,\theta)\right] = \prod_{c=1}^{C} E\left[\prod_{t=1}^{T} \theta_c(t)^{l_c(t)}\right]. \tag{3.3}$$

To be able to sample from Equation (3.1) we have to consider the moments of the Dirichlet distribution. From [9] we have that

$$\begin{aligned} E\left[\prod_{w=1}^{W} \phi_k(w)^{m_t(w)}\right] &= \frac{\Gamma(W\beta)}{\Gamma(W\beta + \sum_{w=1}^{W} m_t(w))} \prod_{w=1}^{W} \frac{\Gamma(\beta + m_t(w))}{\Gamma(\beta)} \\ &\propto_t \frac{1}{\Gamma(W\beta + \sum_{w=1}^{W} m_t(w))} \prod_{w=1}^{W} \Gamma(\beta + m_t(w)) \end{aligned} \tag{3.4}$$

and

$$\begin{aligned} E\left[\prod_{t=1}^{T} \theta_c(t)^{l_c(t)}\right] &= \frac{\Gamma(T\alpha)}{\Gamma(T\alpha + \sum_{t=1}^{T} l_c(t))} \prod_{t=1}^{T} \frac{\Gamma(\alpha + l_c(t))}{\Gamma(\alpha)} \\ &\propto_t \prod_{t=1}^{T} \Gamma(\alpha + l_c(t)). \end{aligned} \tag{3.5}$$

We can now expand Equation (3.1) using Equations (3.2) to (3.5) and get

$$\begin{aligned} p(t|w,c) \propto_t &\left(\prod_{t=1}^{T} \frac{1}{\Gamma(W\beta + \sum_{w=1}^{W} m_t(w))} \prod_{w=1}^{W} \Gamma(\beta + m_t(w))\right) \times \\ &\prod_{c=1}^{C} \prod_{t=1}^{T} \Gamma\left(\alpha + l_c(t)\right). \end{aligned} \tag{3.6}$$

If we ignore the topic of the current word t_{dj} and then set it to topic t, we get that the change in value of Equation (3.6) is

$$\frac{(\beta + m_t^{-dj}(w_{dj}))(\alpha + l_c^{-dj}(t))}{W\beta + \sum_{w=1}^{W} m_t^{-dj}(w)}, \tag{3.7}$$

where c is document d's community. Note that anything to the power of $-dj$ means that we ignore the word in document d, position j.

Moving on to communities we have a similar calculation where the equivalent to Equation (3.1) is

$$p(c|t) \propto p(t|c)p(c) = E[p(t|c,\theta)]E[p(c|\xi)] \tag{3.8}$$

where

$$E\left[p(t|c,\theta)\right] = \prod_{c=1}^{C} E\left[\prod_{t=1}^{T} \theta_c(t)^{l_c(t)}\right] \tag{3.9}$$

$$E\left[p(c|\xi)\right] = E\left[\prod_{c=1}^{C} \xi(c)^{p(c)}\right]. \tag{3.10}$$

Again we have Equation (3.5) but this time it is proportional to c, so we get

$$\begin{aligned} E\left[\prod_{t=1}^{T} \theta_c(t)^{l_c(t)}\right] &= \frac{\Gamma(T\alpha)}{\Gamma(T\alpha + \sum_{t=1}^{T} l_c(t))} \prod_{t=1}^{T} \frac{\Gamma(\alpha + l_c(t))}{\Gamma(\alpha)} \\ &\propto_c \frac{1}{\Gamma(T\alpha + \sum_{t=1}^{T} l_c(t))} \prod_{t=1}^{T} \Gamma(\alpha + l_c(t)). \end{aligned} \tag{3.11}$$

Instead of Equation (3.4) we now have

$$\begin{aligned} E[\prod_{c=1}^{C} \xi(c)^{p(c)}] &= \frac{\Gamma(C\lambda)}{\Gamma(C\lambda + \sum_{c=1}^{C} p(c))} \prod_{c=1}^{C} \frac{\Gamma(\lambda + p(c))}{\Gamma(\lambda)} \\ &\propto_c \prod_{c=1}^{C} \Gamma(\lambda + p(c)). \end{aligned} \tag{3.12}$$

We again consider what happens if we ignore a document d and then classify it as community c. Note that when this is done, $\sum_{t=1}^{T} l_c^{-d}(t)$ in Equation (3.11) can increase by more than 1 and thus we get that $\Gamma(T\alpha + \sum_{t=1}^{T} l_c^{-d}(t))$ increases by

$$\frac{\Gamma(T\alpha + \sum_{t=1}^{T} l_c^{-d}(t) + n_d(t))}{\Gamma(T\alpha + \sum_{t=1}^{T} l_c^{-d}(t))}. \tag{3.13}$$

In addition, $l_c^{-d}(t)$ in Equation (3.11) can increase by more than 1. Here we get that $\Gamma(\alpha + l_c^{-d}(t))$ increases by

$$\frac{\Gamma(T\alpha + l_c^{-d}(t) + n_d(t))}{\Gamma(T\alpha + l_c^{-d}(t))} \tag{3.14}$$

and thus we get the probability of setting document d to c as

$$\left(\lambda + p^{-d}(c)\right) \frac{\Gamma\left(T\alpha + \sum_{t=1}^{T} l_c^{-d}(t)\right)}{\Gamma\left(T\alpha + \sum_{t=1}^{T} l_c^{-d}(t) + n_d(t)\right)} \prod_{t=1}^{T} \frac{\Gamma\left(T\alpha + l_c^{-d}(t) + n_d(t)\right)}{\Gamma\left(T\alpha + l_c^{-d}(t)\right)}. \tag{3.15}$$

We now have the probabilities for updating the topics in Equation (3.7) and communities in Equation (3.15).

3.3 Defining the LDACS Model

We call our second model the LDAC Simple (LDACS), since it is a simpler version of LDAC. It is closely related to LDA, except that for each document you have one community rather than a distribution of topics.

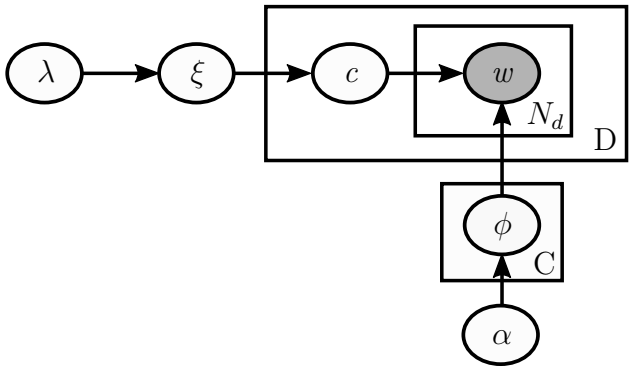


Figure 3.2: Plate notation describing the LDACS model's generative process. For each of the $n = 1...N_d$ word positions in document d, a word w_{dj} is chosen from the community-word distribution $\phi_c \sim \text{Dir}(\boldsymbol{\alpha})$ where $\boldsymbol{\alpha} = (\alpha, \alpha, ..., \alpha) \in \mathbb{R}^W$ and W is the number of unique words in all documents combined. For each document d it is assumed that the distribution over communities ξ is generated from $\text{Dir}(\boldsymbol{\lambda})$ where $\boldsymbol{\lambda} = (\lambda, \lambda, ..., \lambda) \in \mathbb{R}^C$ and C is the pre-determined number of communities.

The generative process is described in Figure 3.2 and Algorithm 6. It has two distributions: the community distribution $\xi \in \mathbb{R}^C$ and the word-community distribution $\phi \in \mathbb{R}^{C \times W}$, where W is the number of unique words in all documents combined and C is the pre-determined number of communities. Then, using our chosen hyperparameters α and λ, we assume that $\phi_c \sim \text{Dir}(\boldsymbol{\alpha})$ where $\boldsymbol{\alpha} = (\alpha, \alpha, ..., \alpha) \in \mathbb{R}^W$ and $\xi \sim \text{Dir}(\boldsymbol{\lambda})$ where $\boldsymbol{\lambda} = (\lambda, \lambda, ..., \lambda) \in \mathbb{R}^C$. Finally, for each of the $n = 1...N_d$ word positions in d, choose a word w_{dj} from ϕ_c. For each document d it is assumed that the number of words N_d is fixed.

Algorithm 6 LDACS Generative Process

$\xi \sim \text{Dir}(\boldsymbol{\lambda})$
for community c in communities $1...C$ **do**
  $\phi_c \sim Dir(\boldsymbol{\alpha})$
end for
for document d in documents **do**
  Set community $c_d = c$ with probability ξ_c
  for word position j in document d **do**
    Set word $w_{dj} = w$ with probability $\phi_{c_d w}$
  end for
end for

Gibbs sampling for LDACS is described by Algorithm 7. The math for collapsed Gibbs for LDACS was calculated but never implemented. Thus it's not that relevant and has been placed in Appendix C instead of here.

Algorithm 7 LDACS Gibbs Sampling

for number of iterations **do**
  $\xi \sim Dir(\lambda + p(1), ..., \lambda + p(C))$
  for Community c in communities $1...C$ **do**
    $\phi_c \sim Dir(\alpha + o_c(1), ..., \alpha + o_c(W))$
  end for
  for document d in documents **do**
    $c_d = c$ with probability proportional to $\xi_c \prod_{w=1}^{W} \phi_{c,w}^{q_d(w)}$
  end for
end for

3.4 Hyperparameters

Our hyperparameters α, β and λ were all set to 0.1 since then the sampled distributions will be closer to a corner or edge, as can be seen in Figure 2.1. That the distributions are close to a corner or edge is a reasonable assumption for the distributions since it is reasonable to assume that each document is focused on few topics, and are very unlikely to be equally consider all topics (which would place it in the center of the triangle).

4
Results

4.1 Evaluating LDAC on Twitter Data

The models presented in this section were trained on a random selection of 80 thousand out of the 550 thousand tweets in the dataset. This since using more data would make training each model take prohibitively long time.

In order to get a better understanding of the model performance, some of the models where trained multiple times with the same hyperparameters but with different random seeds for part of the process. However, figures in this section will, unless otherwise stated, only visualize the first instance of any such model. Corresponding figures for the other instances are generally very similar, and are not included for the sake of brevity.

If nothing else is stated, the models shown here assume that there are 10 topics and 5 communities. These hyperparameters are somewhat arbitrary, but based on the idea that a community should include one or more topics. In addition, early tests that assumed many more communities (upwards 30) showed that some words and hashtags would often be grouped together, but the same groups would occur in several communities, indicating a high overlap between some communities. With only 5 communities, the number of overlapping words is much lower.

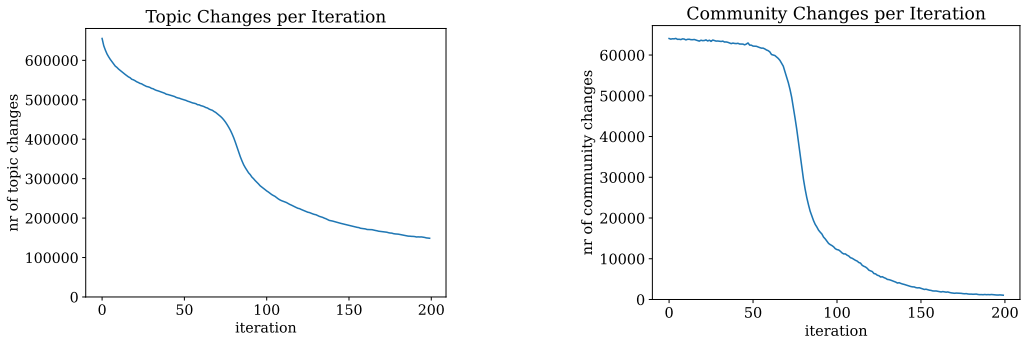
4.1.1 Convergence Evaluation

Our main model of interest is the LDAC model with 10 topics and 5 communities, being run for 200 iterations. A requirement for the model to do anything useful is that it improves during training. We measure this by seeing how much the model variables changes over time. After each iteration the model should on average be a little more stable and thus change a little less every iteration after that.

The problem with Monte Carlo algorithms like Gibbs sampling is that there is no proof that a model has converged. Instead we define convergence as when the change in number of changes per iteration is small in comparison to earlier in the training.

Figure 4.1 shows the number of times a word changes topic, and the number of times a document changes community, in every iteration of training the LDAC model mentioned above. It is clear that the model is at a more stable state at the end of training than at the start, although it does not converge completely. Other instances of the model behaves similarly, but not exactly the same. For the LDA and LDACS models that we will look at further below, the same reasoning about convergence applies, but LDA only has topic changes and LDACS only has community changes.

In addition, Figure 4.2 shows that all communities are utilized by the model, since every community has quite a few documents associated with them. If instead all documents had been classified as one community some problem could be suspected. At the same time, it may also have been suspect if all communities were of the exact same size. Neither of these scenarios seems to be the case from the figure.



(a) Number of words changing topic after each iteration.

(b) Number of documents changing community after each iteration.

Figure 4.1: Graphs showing how LDAC (10 topics, 5 communities) stabilizes during training. The model doesn't converge fully, especially not in topic changes, but reaches a much more stable point than at the start of training.

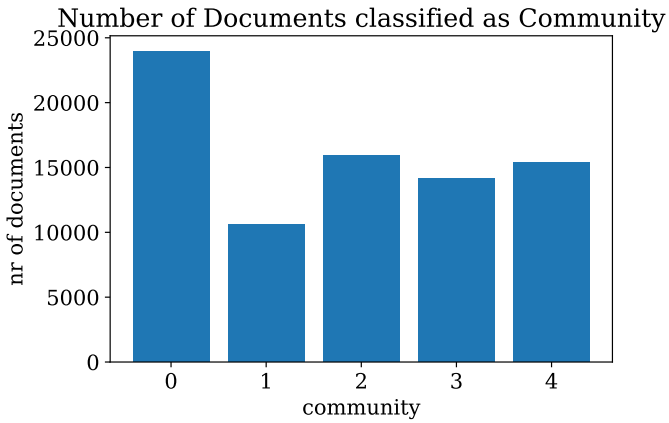


Figure 4.2: Documents classified as each community for LDAC (10 topics, 5 communities). If most documents had been classified as the same community, one could suspect some error, but instead we see a relatively, but not too, even spread.

4.1.2 Stability Evaluation

It is not a given that LDAC is stable for different initial conditions. To determine if it is, the model was trained multiple times for different initial states. Table 4.1 shows the average and standard deviation in adjusted rand score for model instances using two different initial states (IS), both for LDA and LDAC. For each model and initial state (each row in the table), 24 model instances were trained on the same

data, each resulting in a slightly different clustering due to the random steps taken during the training process. The adjusted rand score was then calculated for each pairwise combination of the 24 models, of which the average and standard deviation is presented.

Table 4.2 similarly shows the adjusted rand score for 24 instances of LDA, and of LDAC, but where each is trained with a different initial state. What we can see by comparing LDAC in Table 4.1 to LDAC in Table 4.2 is that the performance seems largely the same between training runs where the initial state is constant, and where it varies, indicating that the model is relatively stable on initial state, even though the standard deviation shows that the result can vary depending on the random steps in the training process. It is possible that there is no entirely stable minimum to find in the data.

If we look at the topic Rand scores, LDAC has a similar average score in both tables, showing that it has a similar stability to LDA, even if the slightly lower average score and higher standard deviation indicates that it may be performing slightly worse. However, the community Rand score is noticeable higher than the Topic Rand score for both LDAC and LDA. This could be because each community has more data than each topic, and is thus able to give more consistent results.

LDA seems to get a higher topic Rand score than LDAC. This is likely because LDA only calculates topics, whereas LDAC uses more parameters at the same amount of data, which could make its results less stable. Since LDAC is more complex it may need more data to perform at its peak, which LDA is able to do at less data.

Table 4.1: Average and standard deviation of adjusted Rand score for training the same model 24 times with the same initial state (but random steps afterwards), calculated on all combinations of clusterings for each model and initial state.

Model	Topic Rand	Community Rand
LDA, IS=1	$\mu = 0.2243, \sigma = 0.0233$	–
LDA, IS=2	$\mu = 0.2220, \sigma = 0.0324$	–
LDAC, IS=1	$\mu = 0.1875, \sigma = 0.0469$	$\mu = 0.2898, \sigma = 0.0610$
LDAC, IS=2	$\mu = 0.1972, \sigma = 0.0446$	$\mu = 0.3112, \sigma = 0.0710$

Table 4.2: Average and standard deviation of adjusted Rand scores for training the same model 24 times with different initial state each time, calculated on all combinations of clusterings for each model.

Model	Topic Rand	Community Rand
LDA	$\mu = 0.2098, \sigma = 0.0265$	–
LDAC	$\mu = 0.1813, \sigma = 0.0500$	$\mu = 0.2706, \sigma = 0.0576$

4.1.3 Visual Inspection of Communities

While model convergence is needed, it does not guarantee a good result. To get a more intuitive idea of how well the model performs, we take a look at which

words LDAC most strongly relate to each community and topic. The words which most strongly relates to each community can be seen in Table 4.3, those that most strongly relate to each topic can be seen in Table 4.5 and the hashtags that most strongly relate to each community can be seen in Table 4.4.

The most ”strongly related” words are those that, as a percentage of their total number of occurrences, most often are assigned to each community or topic. Each word has occurred at least 80 times in total. Without this limit, words that only occur a few times would easily get a full score without actually characterizing the community or topic very well. (Consider the extreme case where a single word would always get maximum score, since it could only be assigned to a single community or topic.)

To note here is also that visual inspection is prone to bias. It is easy to quickly get an idea of what a group of words are about and then fit the remaining words to that. To counteract this somewhat, we have tried to not use too specific search terms when searching for a connection to these words online, as many terms and word combinations are guaranteed to appear in some location. We have also made sure to verify the dates of all articles and other search hits, so that the connections we found would have been relevant during the time frame when the data was collected.

Let us now study these tables closer. If we look at community 0 in Table 4.3, we see some German words ”wir” (we), ”sind” (are), ”gegen” (against) and ”rassismus” (racism). This is probably because while we only looked for trending hashtags in the UK and US, and user accounts marked as using English, it is likely that some hashtags trended in another country too. If look at the top hashtags for this community (Table 4.4) we can see three similar hashtags: ”RassismusBeiBayern3”, ”Bayern3Racist” and ”RacismBayern3”. Looking into these, they seem to originate from a scandal where a host on the German radio channel ”Bayern 3” was accused of making racist remarks[11]. It makes sense then that Germans who usually write English tweets, could have used some German to tweet about this local issue. These particular words ”wir sind gegen rassismus", may also have been some kind of slogan and therefore been used more widely. These hashtags and several of the words seem to have a very strong correlation as they appeared together in all instances of the LDAC model that we have looked at. In this community are also several other words that do seem likely to be used in discussions of racism, such as ”racial”, ”perceptions” and ”condemn”, although it is difficult to say for sure.

Community 1 has several words that seem connected to celebrating and toasting, which would seem to relate well to the hashtags ”NationalMargaritaDay” and ”NationalToastDay” in this community. If it makes sense for all or only some of these words to belong here is hard to determine. This community may be a bit weaker since it did not seem so visible in other instances of the model.

In Community 2, many of the words seem related to the Marvel Avenger’s spin-off episodic tv-series ”Wanda Vision” which was running at the time of data collection. The first two words ”olsen” and ”elizabeth” refer to Elizabeth Olsen, the actress portraying the shows main character Wanda Maximoff (”maximoff” being one of the other words). Similarily, Monica is the name of another character in the show, and Kathryn Hahn another actress. Most of the other words also seem to have a connection to the show or to the Marvel Avenger’s franchise, but the ”SnowfallFX”

Table 4.3: The 15 words that are the most strongly related to each community and occurred at least 80 times in the training data. Each community has been given a label that we think describes what the community is about.

Community 0	**Community 1**	**Community 2**	**Community 3**	**Community 4**
Bayern 3 Scandal	*Celebration?*	*Wandavision*	*?*	*Cricket*
disrespectfulapologize	shining	olsen	demoralizing	axar
rtrep	margarita	elizabeth	crossing	sharma
dnpthree	pearl	monica	fingers	motera
bhookha	diamond	emmy	phenomena	pope
dheere	margaritas	kathryn	tmobile	ashwinravi
condemn	toast	westview	plague	patel
wir	spring	hahn	ashram	bowler
sind	zack	finale	crude	lbw
gegen	national	nigga	satlok	foakes
rassismus	weather	maximoff	kabir	bowled
respected	celebrate	skully	persevering	bumrah
tolerated	trailer	avenger	win	sibley
hoye	celebrating	cried	mantra	ishant
racial	roadmap	manboy	chapter	leach
perceptions	dip	episode	destroyed	ind

hashtag instead seems to relate to an entirely different TV-series whose fourth season premiered during our data collection. This community about Wanda Vision or tv-series has, just as community 0, been present in all LDAC models we've inspected.

For Community 3 we have not been able to determine a clear connection between the different words and hashtags.

Community 4 has also been present in virtually all our models, and seems to mainly concern cricket, but perhaps also some football. The hashtags "INDvENG" and "channel4cricket" both seem to relate to one or several games between India and England. Almost all of the words in this community also seems relevant to cricket. Here, "bowled" and "bowler" are cricket terms, "sibley" is the name of an Indian cricket stadium, and Ravichandran Ashwin ("ashwinravi"), Foakes, Axar Patel, Ishant Sharma, Bumrah, Sibley, Pope and Leach all seem to be names of current Indian or English cricket players. This community, just as community 0 and 2, has been consistently present in the instances of the LDAC model we've inspected.

The words and hashtag clusters are not perfectly consistent between models, as demonstrated by the Rand scores in 4.1.2. However, some words do repeatedly appear together and do seem to belong, even if they don't immediately make sense.

Figure 4.3 shows how relevant each topic is in each community. By comparing this figure with Tables 4.3 and 4.5) we can see some similarities between the topics and the communities. Most importantly, Figure 4.3 shows that each community has a fairly distinct topic distribution, with fairly little overlap between communities, but also that the communities generally correspond to more than a single topic. For

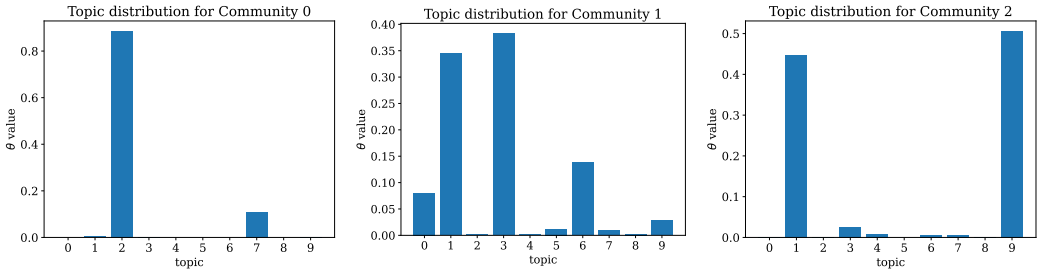
Table 4.4: The 5 hashtags that are the most strongly related to each community. Each community has been given a label that we think describes what the community is about.

Community 0	**Community 1**	**Community 2**	**Community 3**	**Community 4**
Bayern 3 Scandal	*Celebration?*	*Wandavision*	*?*	*Cricket*
RassismusBeiBayern3	PurpleFriday	SnowfallFX	5Gsfor5G	INDvENG
Bayern3Racist	FMQs	WandaVision	talkswithAsh	channel4cricket
RacismBayern3	NationalMargaritaDay	RIPTwitter	RIPTwitter	FreeBetFriday
FreeSpideyPS5	NationalToastDay	FreePapaJohns	VaccinePassports	Lionesses
talkswithAsh	Roadmap	idontsteal	WednesdayMotivation	UELdraw

Table 4.5: The 15 words that are the most strongly related to each topic.

Topic 0	**Topic 1**	**Topic 2**	**Topic 3**	**Topic 4**
margarita	weeks	apologize	celebrating	demoralizing
digital	tonight	tolerated	toast	tmobile
boris	credits	respected	borisjohnson	fingers
lockdown	remakes	justifying	shining	crossing
garden	elizabeth	disrespectfulapologize	zack	phenomena
goals	hahn	rtrep	trailer	plague
arceus	snap	condemn	johnson	crude
pearl	house	antiasian	february	satlok
increase	week	kim	schools	ashram
link	drawing	wir	national	kabir
pokmon	art	sind	march	win
football	wait	gegen	celebrate	destroyed
report	parents	rassismus	diamond	mantra
path	vibes	perceptions	market	persevering

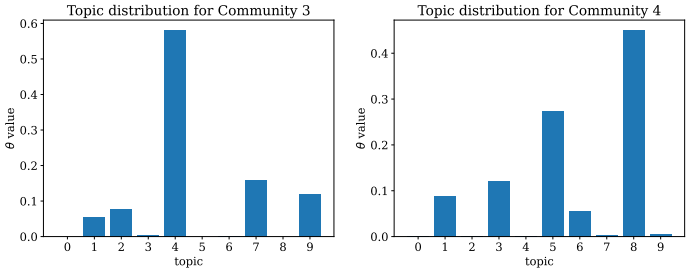
Topic 5	**Topic 6**	**Topic 7**	**Topic 8**	**Topic 9**
manchester	roadmap	disgusted	bcci	visions
motera	lovely	virus	milan	cw
bowling	weather	viewed	axar	villain
stadium	saturday	station	rohit	avengers
joe	april	description	sharma	monica
fiverr	food	hurt	ashwin	manboy
runs	alex	sorry	batsman	avenger
congratulations	etsy	maharaj	stokes	disneyplus
tests	health	tired	ashwinravi	maximoff
umpires	challenge	germany	akshar	westview
indias	showreel	calling	lbw	cried
anderson	data	matuschik	foakes	spoilers
largest	including	band	bowled	wandavision
broad	brilliant	anymore	crawley	marvel



(a) Topic distribution for community 0.

(b) Topic distribution for community 1.

(c) Topic distribution for community 2.



(d) Topic distribution for community 3.

(e) Topic distribution for community 4.

Figure 4.3: Topic distribution for the different communities. Each community has a distinct topic distribution, with fairly little overlap between communities. The communities generally correspond to more than a single topic.

instance, community 0 does rely heavily on topic 2, but no other community uses topic 2 much at all. One exception is that community 1 and 2 both use topic 1, but both communities also have one other topic they rely on even more.

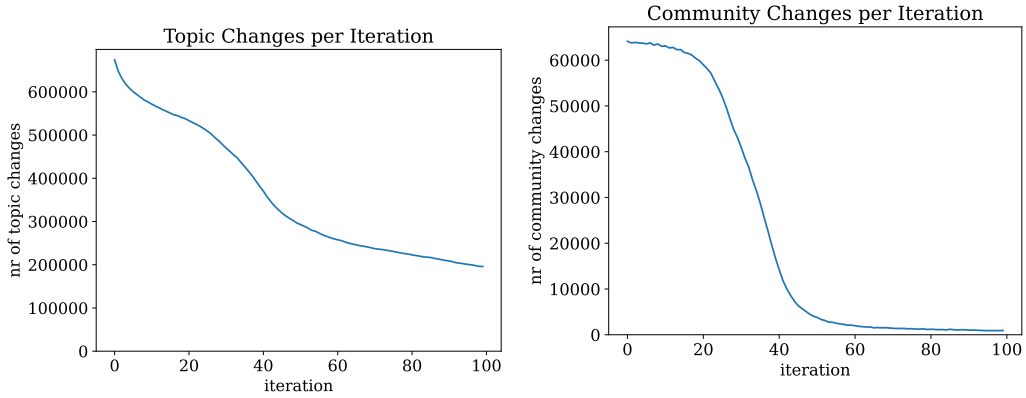
4.1.4 Collapsed Gibbs Comparison

Whether or not to train the model using collapsed Gibbs sampling or non-collapsed Gibbs sampling could affect the result. Unfortunately there was not enough time to test this extensively.

Collapsed Gibbs sampling allowed LDAC to converge in fewer iterations. Figure 4.4 shows that using collapsed Gibbs allows the model to converge approximately as much in 100 iterations as non-collapsed does in 200 (compare to Figure 4.1).

However, it is valuable to note that our implementation of collapsed Gibbs, in real time, took over eight times as long to train for those 100 iterations, compared to the 200 iterations of non-collapsed Gibbs. This difference lies primarily in the optimizations we were able to do for the non-collapsed Gibbs, but which were not made for the collapsed Gibbs due to a lack of time and a greater difficulty. It is likely that with proper optimizations, collapsed Gibbs could run faster than or at least as fast as non-collapsed Gibbs.

Due to collapsed Gibbs taking much longer to train, it was not possible to study it as closely. However, when comparing one collapsed Gibbs LDAC to 24 non-collapsed Gibbs LDAC, the adjusted Rand score suggests that the collapsed Gibbs



(a) Changes over time for topics.

(b) Changes over time for communities.

Figure 4.4: Changes over time for LDAC trained with collapsed Gibbs on Twitter data. Collapsed Gibbs approaches convergence in noticeably fewer iterations, here trained for 100 iterations compared to the 200 iterations for other LDAC.

model achieves somewhat different results. This can be seen in Table 4.6 where the adjusted Rand score is significantly lower when comparing the collapsed to the non-collapsed than when comparing the non-collapsed against itself. However, since only one non-collapsed instance is being considered here, it is difficult to draw any general conclusions.

Table 4.6: Adjusted Rand scores of comparing a single collapsed Gibbs LDAC trained over 100 iterations to 24 non-collapsed Gibbs LDAC trained over 200 iterations, and Rand scores of comparing 24 non-collapsed Gibbs LDAC pairwise against themselves. The model trained with collapsed Gibbs seems to overlap less with the non-collapsed than the non-collapsed does with itself, suggesting there is some difference in result.

Model	Topic Rand	Community Rand
Collapsed vs Non-collapsed	$\mu = 0.0977, \sigma = 0.0141$	$\mu = 0.1818, \sigma = 0.0273$
Non-collapsed vs Non-collapsed	$\mu = 0.1875, \sigma = 0.0469$	$\mu = 0.2898, \sigma = 0.0610$

4.2 Evaluating Models on Reddit Data

Next we present the results of training the models on Reddit data. Table 4.7 shows the adjusted Rand score of comparing the model clusters against the ”real” community clusters that the subreddits represent. This differs from the Rand score measures on models trained on Twitter data where Rand score could only be used to compare the consistency between multiple runs of the model. This time each model was only run three times, due to a much longer training time on the Reddit data. Figures in this section only show the first run unless otherwise stated as those figures are quite similar between runs.

Table 4.7: Adjusted Rand scores of comparing the clusters created by the models to the ”real” clusters in the Reddit data. Due to long training times, these models were only run three times. This table shows the adjusted Rand score for each run, and the average of those three times.

Model	Topics	Communities	Rand 1	Rand 2	Rand 3	Rand Avg
LDA	6	-	0.2165	0.1690	0.2243	0.2033
LDAC	5	6	0.1122	0.1605	0.1395	0.1374
LDAC	10	6	0.1604	0.1675	0.1638	0.1639
LDAC	15	6	0.1615	0.1261	0.1506	0.1461
LDAC	20	6	0.1473	0.0987	0.1425	0.1295
LDACS	-	6	0.2074	0.2318	0.2457	0.2283

4.2.1 LDA with Reddit-Many

Since LDA has no built-in concept of community, but a community label would be needed to be given to each document for comparison with the other models, classification was made by labeling each document by the most common topic classification for its words. The adjusted Rand-score is the comparison between the clustering done by LDA and the subreddits.

For LDA, 150 iterations seemed to be enough to make the changes over time converge (see Figure 4.5 for the convergence of the first run). The adjusted Rand score was 0.2165 when comparing to the real communities (see Table 4.7).

The ”real” community clusters from the Reddit data all have approximately the same number of documents. Thus we might expect to see a fairly even spread in the model clusters as well. This is what we do see for the LDA model in Figure 4.6a.

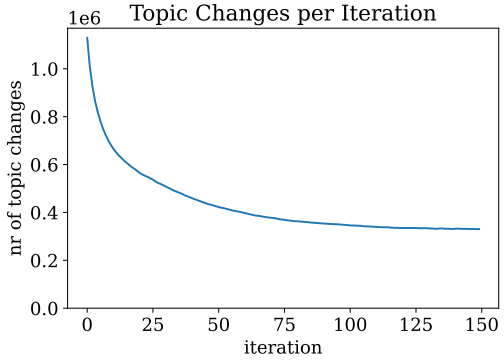
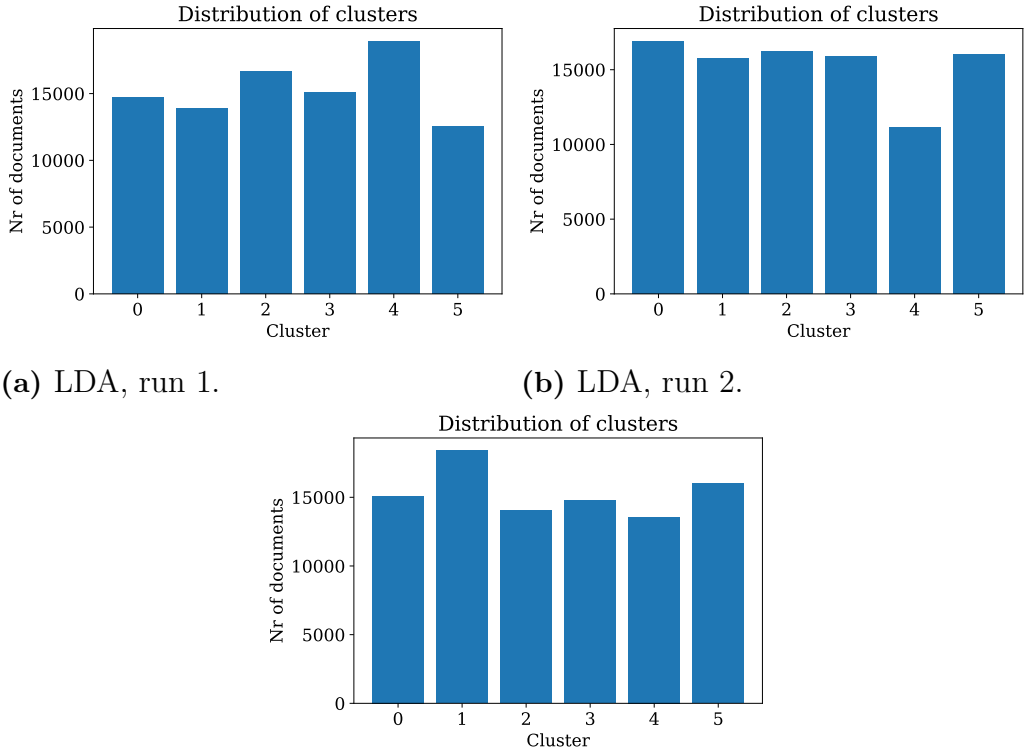


Figure 4.5: Topic changes over time for LDA on the Reddit data for run 1.



(a) LDA, run 1.

(b) LDA, run 2.

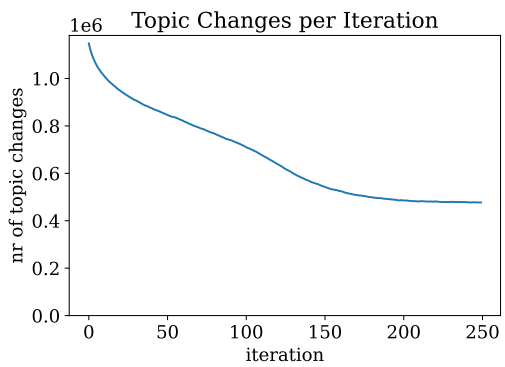
(c) LDA, run 3.

Figure 4.6: Number of documents classified as each community for LDA on the Reddit data. There is approximately the same number of documents in each cluster, which is what might be expected given the training data.

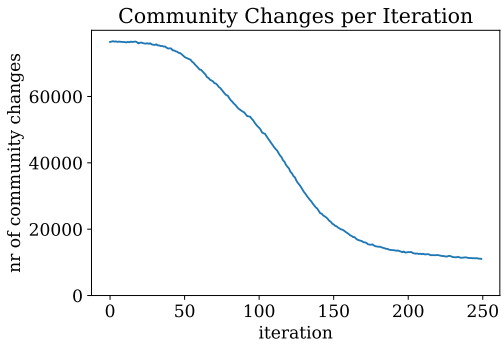
4.2.2 LDAC with Reddit-Many

To try finding communities in the Reddit data using LDAC, models using different numbers of topics were tried (5, 10, 15, 20), all assuming 6 communities. According to Table 4.7, assuming 10 topics performed the best (although this is fairly uncertain due to the low number of runs). This is the LDAC instance we will be referring to for now. All the instances were trained over 250 iterations, which seemed to be enough for the model to converge.

Figure 4.7 shows the changes over time for topics and communities and Figure 4.8 shows the number of documents for each community for LDAC run 1 with 10 topics. In Appendix A Figures A.1 and A.2 shows the topic and community changes over time for all topics for run 1.

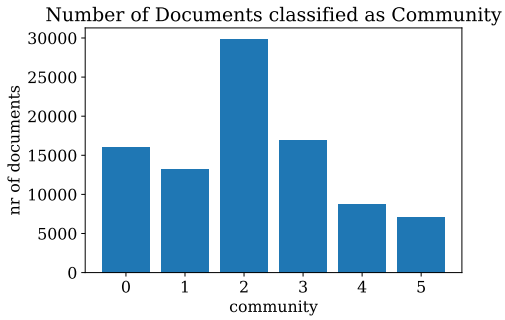


(a) Changes over time for topics.

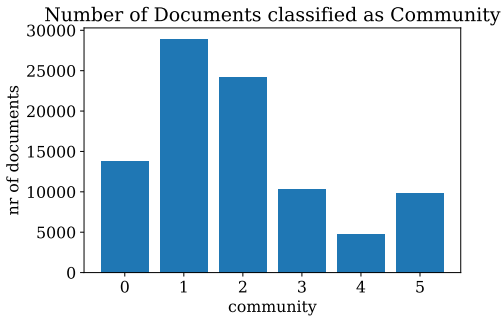


(b) Changes over time for communities.

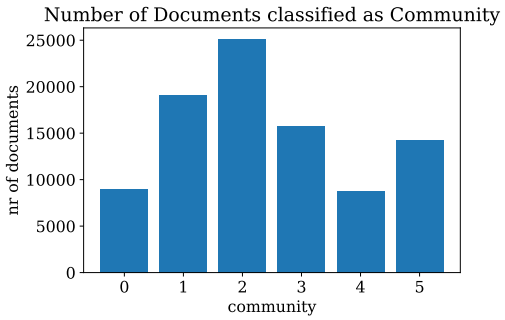
Figure 4.7: Changes over time for LDAC run 1 with 10 topics and 6 communities on the Reddit data. Both topic changes and community changes seem to level out towards the end of training, but the number of topic changes per iteration is still quite high.



(a) LDAC with 10 topics, run 1.



(b) LDAC with 10 topics, run 2.



(c) LDAC with 10 topics, run 3.

Figure 4.8: The number of document classified as each community for LDAC with 10 topics. The Reddit data has 6 ”real” communities with an equal number of documents for each. Here we see a consistently uneven spread of documents over communities. This is an indication that LDAC does not find the ”real” community clusters.

4.2.3 LDACS with Reddit-Many

For LDACS, 80 iterations seems to be enough to make the community changes over time converge. See Figure 4.9 for the convergence on run 1. The adjusted Rand score compared to the Reddit communities can be seen in Table 4.7.

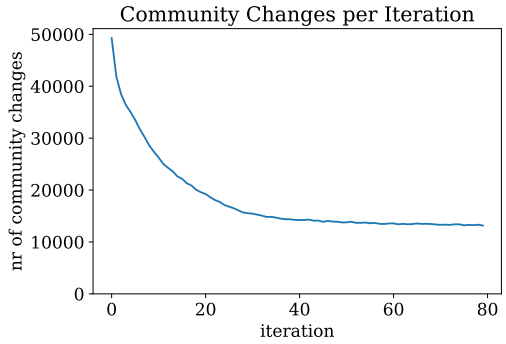


Figure 4.9: LDACS community changes over time for run 1. The curve is similar for run 2 and 3.

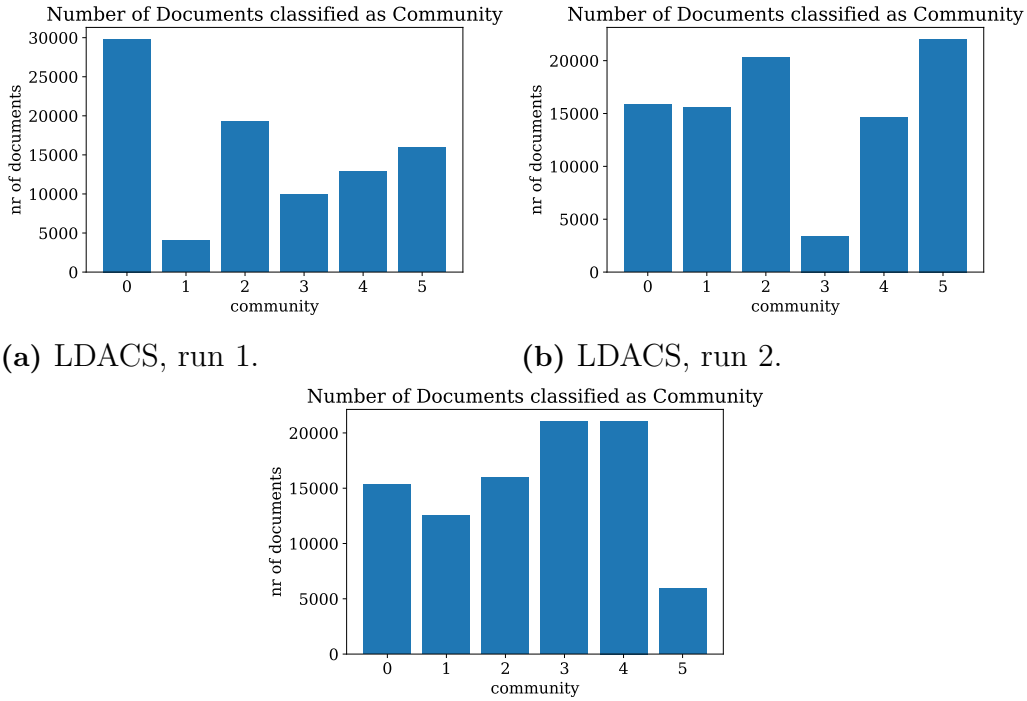
4.2.4 Testing Reddit-Long

We also tested the Reddit data when each post was appended to the post-titles they were part of. The models are the same as for Reddit-Many and evaluated the same way as in previous sections. The scores are summarized in Table 4.8. We see that LDA achieved a very high score compared to on Reddit-Many, whereas LDACS achieves a very low score. For LDA this is probably because there are more words per document and thus more data to help cluster them.

Table 4.8: Adjusted Rand scores of comparing the clusters created by the models to the ”real” clusters in the Reddit-Long dataset. Due to long training times, these models were only run once. We see that LDA achieves a very high score compared to on Reddit-Many, whereas LDACS achieves a very low score.

Model	Topics	Communities	Rand Score
LDA	6	-	0.6297
LDAC	5	6	0.2829
LDAC	10	6	0.1902
LDAC	15	6	0.1156
LDAC	20	6	0.1258
LDACS	-	6	0.0018

The changes over time for topics and communities for the models can be seen in Figure 4.11 for LDA, Figure 4.12 for the LDAC model with 5 topics (since it performed the best) and in Figure 4.13 for the LDACS model.



(a) LDACS, run 1.

(b) LDACS, run 2.

(c) LDACS, run 3.

Figure 4.10: The number of documents classified as each community for LDACS. The Reddit data has 6 ”real” communities with an equal number of documents for each, here we see a consistently uneven spread of documents over communities. This is an indication that LDACS does not find the ”real” community clusters.

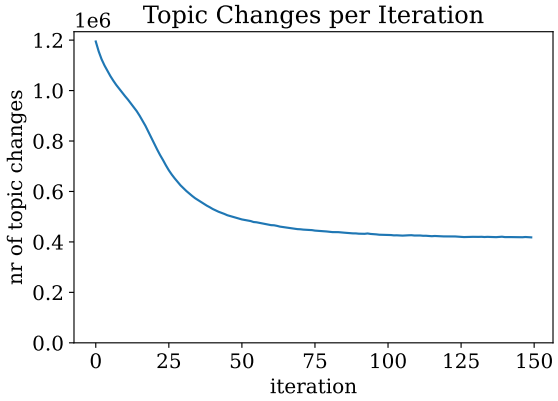
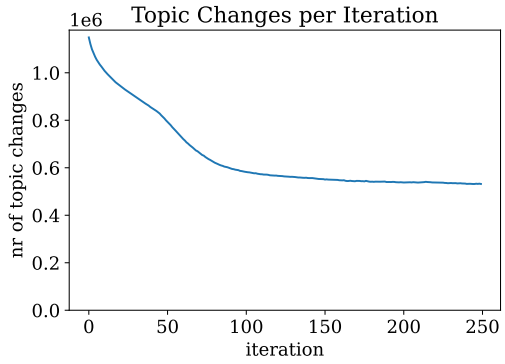
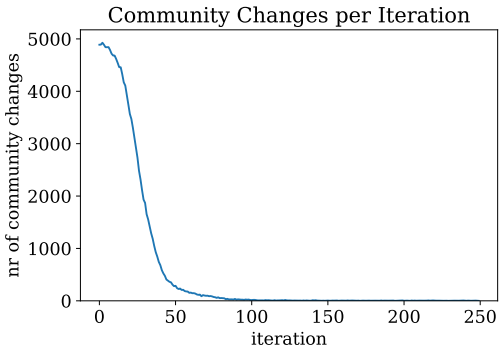


Figure 4.11: Topic changes over time for LDA on the Reddit-Long data.



(a) Changes over time for topics.



(b) Changes over time for communities.

Figure 4.12: Changes over time for LDAC with 5 topics and 6 communities on the Reddit-Long data. Both topic changes and community changes seem to level out towards the end of training, but the number of topic changes per iteration is still quite high.

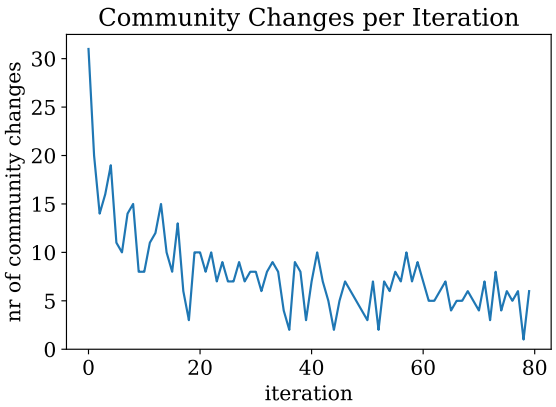


Figure 4.13: Community changes over time for LDACS on the Reddit-Long dataset.

5
Discussion

5.1 Visual Inspection

Section 4.1.3 presented the words most strongly related to each community and topic, showing that LDAC identifies some communities that can be given reasonable labels, although there is a fair bit of noise. However, one question that arises is whether or not topics have a meaningful place within communities. Are there subdivisions within communities considering distinct but related topics? Are some topics shared between communities, and if so, are those communities meaningfully similar in some way?

To try answering these questions we need to take a closer look at how the LDAC topic distributions (Figure 4.3) for each community relates to the words most strongly related to each community (Table 4.3).

In community 0 ”Bayern 3 Scandal”, looking at the distribution of topics for the first community (Figure 4.3a), we see that it consists almost entirely of topic 2, and the top words for topic 2 are indeed very similar to those of community 0.

In community 1 ”Celebration”, Figure 4.3b shows that it has connections to several topics, primarily topic 1 and 3. Looking at Table 4.5, topic 1 seems to partly concern movies and tv (”elizabeth hahn”, ”credits”, ”remakes”), whereas topic 3 seems to refer to celebration, more similar to the top words in the community.

In community 2 ”Wandavision”, the topic distributions in Figure 4.3c shows that the community mostly contains topic 1 and 9. Topic 1 is reasonable since it is about movies and tv-series and mentions two actors from WandaVision, and Topic 9 seems more focused on the characters and the Marvel franchise in general, rather than actors. It makes sense for this community to have two topics, one for tv and one for the particular franchise. Interestingly, topic 1 is also shared with community 1, suggesting that community 1 and 2 have similarities.

For community 3 we can see in Figure 4.3d that it almost entirely overlaps with topic 4. Looking at the words we can see that they are very similar.

For community 4 ”Cricket” we can see from the topic distribution in Figure 4.3e that the community primarily uses topic 8 and 5. Topic 8 is almost entirely about cricket players, and topic 5 seems to concern cricket and sports in more general terms. The fact that the model has kept these topics separate, yet grouped them into a single community does speak to that there is something being gained from considering both communities and topics.

In conclusion there seems to be some meaningful subdivision within the communities, most communities makes use of more than one topic, and one topic is being

utilized by more than one community. This indicates that the relationship between topics and communities are somewhat meaningful. However, there also seems to be noise, or at least a number of words for which we are unable to determine why or why not they belong. This makes drawing general conclusions difficult, but at the very least there doesn't seem to be a trivial relationship between communities and topics, such as each community only relating to a single topic, in which case the community-topic distribution would have seemed unnecessary.

5.1.1 Twitter Hashtags

As was mentioned in Section 3.1.1, we initially planned to use hashtags as a proxy for real data labels, but abandoned this idea in favour of looking at Reddit data, mainly because single hashtags can not reasonably be said to approximate a community label. This is demonstrated by Table 4.4 where amongst others, there are three barely-distinguishable hashtags about the Bayern 3 scandal and two apparent variants of the "NationalToastDay".

At the same time, it is evident that some of the hashtags do not seem to belong to their community. This can be helpful to determine where some of the less obvious words in Table 4.3 come from, but is probably also a side-effect of the limited number of hashtags in total. There simply doesn't seem to be 5 relevant hashtags to find for each community.

5.1.2 Investigation of a Strange Word

As a last note on the Twitter data, interesting but somewhat unrelated to our models, one of the most common words under community 0 in Table 4.3 and topic 2 in Table 4.5 was "disrespectfulapologize". This seemed odd as that shouldn't be a common word. After some investigation it turned out that multiple accounts had posted similar tweets, each with slight variations of the sentence *"That is very disgusting and disrespectful.APOLOGIZE TO BTS"*, but all of them forgetting the space after the dot. See Appendix B for 4 examples of these tweets. In the dataset were a total of 936 documents containing the word "disrespectfulapologize".

When cleaning the data, words were recognized as series of letters separated by white space, and special symbols were removed, so in this case were a dot is not followed by a space, "disrespectfulapologize" shows up as a single word. Then, since many accounts were spamming very similar tweets, "disrespectfulapologize" ended up as one of the top words in it's own community/topic. What is particularly interesting is how many comments had the exact same typo, possibly indicating some automated process behind these tweets.

5.2 Reddit data

Comparing model performance on Reddit data was interesting to see how well they would agree with Reddit's definition of communities: the subreddits. We also looked at two different ways of using the Reddit data. First, in form of the Reddit-Many

dataset, where comments were considered their own documents. Second, in the form of the Reddit-Long dataset, where comments were appended onto their post.

The reason these datasets were created, which don't differ in data, but only in how the data is structured, was to explore the difference between using long documents, and using many documents. Due to the time restrictions, it was easier to use the same data for both datasets than to find two sets of data that would have to fulfill different requirements.

Instead the assumption was made that the words in comments and the words in posts are of the same distribution. This seemed like a reasonable assumption because while comments are about a post, and the post in turn is about a community, the comments could be considered both part of the community as their own documents, or as an extension to the post itself.

5.2.1 Reddit-Many

Table 4.7 shows that LDAC's clusterings don't overlap that well with the subreddits, compared to LDA's and LDACS's. However, at the same time (although on Twitter data) Tables 4.1 and 4.2 indicates that for communities, LDAC clusters have good overlap with itself over multiple runs. It thus seems like LDAC produces clusterings that are stable, yet differ from the subreddits, but which may be reasonable in some other way.

On the Reddit-Many data, LDA is the only model that seems to classify a similar number of documents into each cluster (compare Figures 4.6, 4.8 and 4.10). Since the Reddit data consists of 6 subreddits, each with the same number of documents associated, this is what one might expect from all models. However, some of the subreddits may also be very similar, such as the programming subreddit and the machine learning subreddit. It is thus possible that our models, with good reason, considers some of the communities as one.

To note here is that a more complex model like LDAC may need more training data than a simpler model to perform at its best. We can even see from Table 4.7 that LDACS, which is less complex than LDA, seems to perform slightly better.

5.2.2 Reddit-Long

For the Reddit-Long data, LDA is the best at finding the original clusters with a score of 0.63 while LDACS the worst with a score of 0.0018. For LDACS, as can be seen in Figure 4.13, not many communities change at all. This is probably the reason for the poor performance.

The reason for LDACS not changing communities is that there are not so many documents while those that do exist are more effective at defining what the community is. For example if we have some words, $(w_1, ..., w_{N_d})$, in a document classified as community c, when we resample the document's community we try to see what community is similar to that document. That community is most likely going to be c since it also has the words $(w_1, ..., w_n)$ since they are part of the document.

For a more mathematical description of this, Algorithm 7 describes when we sample ϕ_c from a Dirichlet distribution. If we count the occurrences of the words

$(w_1, ..., w_{N_d})$ from document d, ϕ_c will on average have a higher value for those words compared to if we didn't. Similar to how the samples are closer to the corners corresponding to the bigger α:s in Figure 2.1.

The process in Figure 4.13 could potentially enter a better state after a really long time since once a document manages to change to a more relevant community, even if it's with a really small probability, it would be an even smaller probability to change back. However this would probably take a much longer time. One potential way to mitigate this might be to use collapsed Gibbs sampling for LDACS as well since it ignores the current document when updating a community. See Appendix C.

LDAC performance is between LDA and LDACS. This is probably because it incorporates both communities and topics. So it has a bit of both. Again, it would be interesting to see how well the collapsed version performed on these documents.

5.3 Conclusions on Model Performance

To begin, LDAC converges fairly well and seems stable for initial conditions, but its results have a lower Rand score average and greater Rand score variance for topics than LDA. This is to be expected since LDAC is a more complex model.

On Rand score for communities, LDAC seems to get a fairly good result on the Twitter data, indicating that it clusters communities more consistently than topics. This seems reasonable when looking at the word lists the model has produced, but is difficult to compare to anything else since neither LDA nor LDACS were used to identify communities in the Twitter data.

At the same time, on Reddit data, LDAC achieves a significantly lower Rand score than LDA and LDACS, indicating that it produces clusters that correspond less with the subreddits than LDA and LDACS. This suggests that LDACS could have performed better than LDAC on the Twitter data as well, but it is also possible that subreddits are an insufficiently good stand-in for the "true" community clustering to allow for that conclusion.

LDACS was unfortunately not tested on the Twitter data due to time constraints, and thus there are no results that are directly comparable to those of LDAC in terms of stability and self-consistency. However, given its simplicity, performance on the Reddit-Many dataset, and its similarity to LDA and LDAC, we hypothesise that it is if anything more stable than LDAC.

On the Reddit-Many data set, LDACS seems to slightly outperform LDA. This is probably because it is slightly simpler and thus generalizes better, but also because it utilizes the concept of a community directly. In comparison, for LDA we needed to artificially create a community label for each document after the model was finished (using the majority of topics within each document).

On the Reddit-Long data set LDA has by far the best self-consistency (Rand score against itself). This is probably because it is hard for LDAC and LDACS to change the community for a dataset with long documents since that document is such a big part of its community and thus helps define it.

It would therefore be interesting to try LDAC and LDACS with collapsed Gibbs on Reddit-Long, since that could potentially solve this issue. Collapsed Gibbs could also potentially help improve the score of LDAC and LDACS for the Reddit-Many data

set, since it might have this problem as well, albeit less serious since the documents are shorter and less defining of the communities.

Curiously however, LDAC trained using collapsed Gibbs seems to produce results that overlaps less well with results from it's counterparts trained with non-collapsed Gibbs. If this difference is good or bad is unclear and more runs to compare its self-consistency would be useful, but was not done due to time constraints.

In conclusion, LDACS seems like a good and simple alternative to LDA for detecting communities but might need more study. LDAC gives some credence to the usefulness of utilizing both topics and communities, but does not produce sufficient results to weigh up for its increased complexity, and it might be necessary to train it on more data.

6
Future Work

The LDACS model seems promising, but due to time constraints it was not possible to investigate this model as thorougly as we would have liked. It would therefore be useful to test its stability directly, similar to as was done for LDAC, and more importantly, do the same kind of visual inspection of the communities as was done for LDAC.

Since Reddit data was a late addition to this paper, there was also not enough time to do a thorough visual inspection of the subreddits compared to the clusterings created by LDAC and LDACS. This might be interesting to test further.

Furthermore it would be interesting to investigate if or how our models could be improved by incorporating word embeddings. This approach may be inspired by one that has been made previously[19].

There are also several possibilities for putting more focus on the authors of documents. It would be interesting to investigate a model which incorporates the author of each datapoint in a similar vein as was done in the CTM[8]. However, in contrast to CTM, the intention would be to not let tweets have multiple authors, since that makes much less sense for tweets than it does for academic papers. This could allow for trying to identify which communities users belong to.

How authors are distributed within communities is also an interesting subject to investigate. Are authors usually active within a single community, or several? Are there key authors within communities?

Finally, it would be useful to look into more refined methods of evaluation. Visual inspection is always a reasonable thing to do, but relying too much on it is also prone to bias. It would be possible to use some evaluation methods for topic models[20], specifically Chib-style [4] estimation and left-to-right evaluation on document completion to evaluate the underlying topic model.

Bibliography

[1] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[2] Thomas Boggs. A script to generate contour plots of Dirichlet distributions, 2014. [Online; accessed 26. May 2021].

[3] Twitter Help Center. How to use hashtags. *Twitter Help Center*, Sep 2020.

[4] Siddhartha Chib. Marginal likelihood from the gibbs output. *Journal of the American Statistical Association*, 90(432):1313–1321, 1995.

[5] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

[6] Twitter Usage Statistics - Internet Live Stats, Mar 2021. [Online; accessed 10. Mar. 2021].

[7] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, page 591–600, New York, NY, USA, 2010. Association for Computing Machinery.

[8] Daifeng Li, Ying Ding, Xin Shuai, Johan Bollen, Jie Tang, Shanshan Chen, Jiayi Zhu, and Guilherme Rocha. Adding community and dynamic to topic models. *Journal of Informetrics*, 6(2):237 – 253, 2012.

[9] Jiayu Lin. On the dirichlet distribution. *Department of Mathematics and Statistics, Queens University*, 2016.

[10] Yan Liu, Alexandru Niculescu-Mizil, and Wojciech Gryc. Topic-link lda: Joint models of topic and author community. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 665–672, New York, NY, USA, 2009. Association for Computing Machinery.

[11] Kaela Malig and Shai Lagarde. Bts fans decry racist comments from german radio host; '#bayern3racist, racism is not an opinion' trend, Feb 2021.

[12] Natural language toolkit.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[14] The python reddit api wrapper.

[15] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[16] Edward Roberts. Is web scraping illegal? depends on what the meaning of the word is: Imperva, Sep 2018.

[17] Seema Singh. Understanding the Bias-Variance Tradeoff - Towards Data Science, Oct 2018.

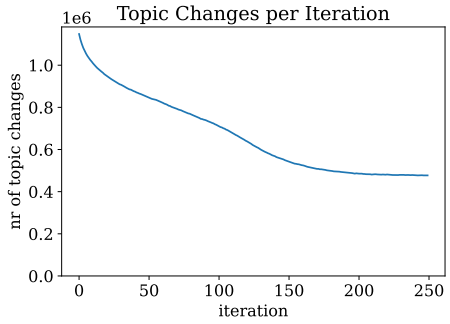
[18] ThoughtVector. LDA Alpha and Beta Parameters - The Intuition, Dec 2019. [Online; accessed 29. Jan. 2021].

[19] Vladimir Vargas-Calderón and Jorge E. Camargo. Characterization of citizens using word2vec and latent topic analysis in a large set of tweets. *Cities*, 92:187–196, Sep 2019.

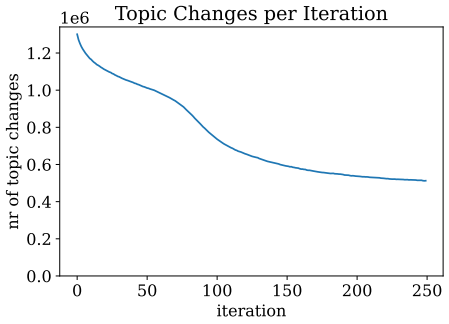
[20] Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. *Evaluation Methods for Topic Models*, page 1105–1112. Association for Computing Machinery, New York, NY, USA, 2009.

A

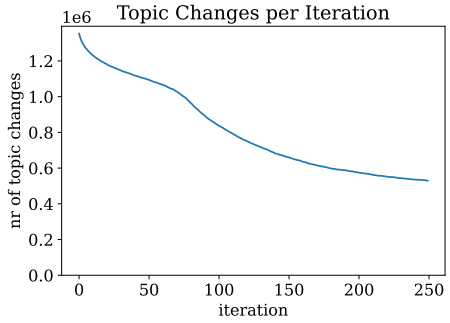
Reddit LDAC converging



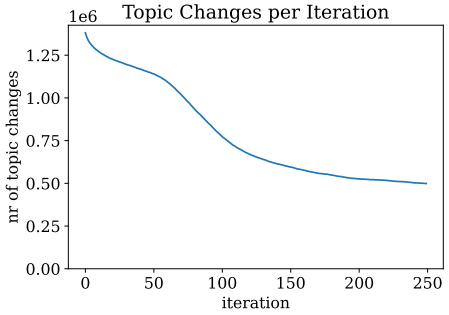
(a) Topic changes over time for LDAC with 5 topics.



(b) Topic changes over time for LDAC with 10 topics.

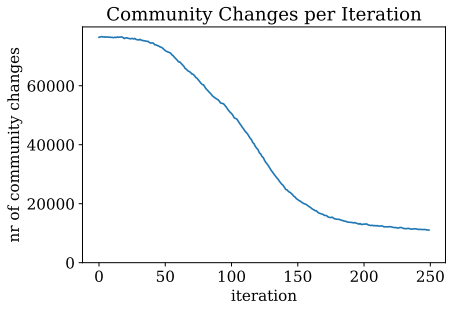


(c) Topic changes over time for LDAC with 15 topics.

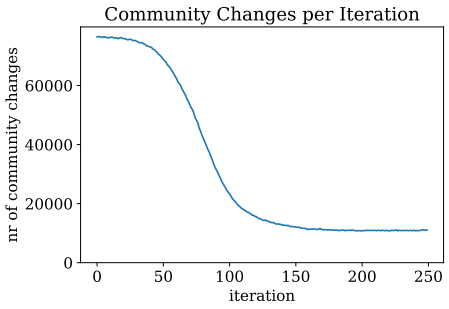


(d) Topic changes over time for LDAC with 20 topics.

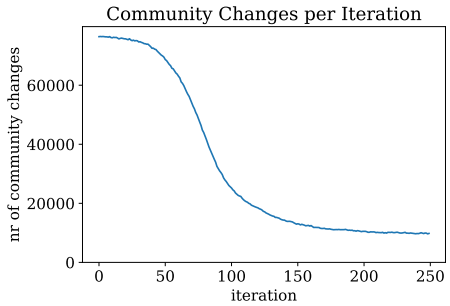
Figure A.1: Topics changes over time for the first run LDAC on the reddit data.



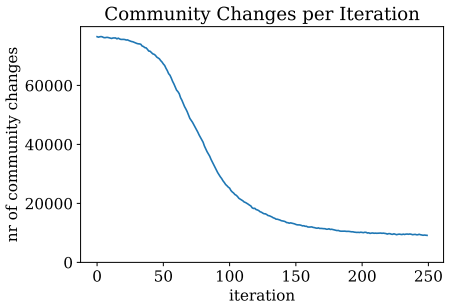
(a) Community changes over time for LDAC with 5 topics.



(b) Community changes over time for LDAC with 10 topics.



(c) Community changes over time for LDAC with 15 topics.



(d) Community changes over time for LDAC with 20 topics

Figure A.2: Community changes over time for the first run LDAC on the reddit data.

B

Similar tweets

Here is a list of urls to very similar tweets that we found when processing the data:

- https://web.archive.org/web/20210226052720/https://twitter.com/jinphipany92/status/1365171521254166531
- https://web.archive.org/web/20210226062350/https://twitter.com/GdV3onrTRXJKclR/status/1365185692897210368
- https://web.archive.org/web/20210226062749/https://twitter.com/kcyc613528/status/1365186663626928129
- https://web.archive.org/web/20210226061408/https://twitter.com/IJkwife/status/1365183288566931457

C

Collapsed Gibbs for LDACS

Collapsed Gibbs was not implemented in code and thus will be presented here in appendix instead.

$$p(c|w) \propto p(w|c)p(c) = E[p(w|c,\phi)]E[p(c|\xi)] \tag{C.1}$$

where

$$E\left[p(w|c,\phi)\right] = \prod_{c=1}^{C} E\left[\prod_{w=1}^{W} \phi_c(w)^{o_c(w)}\right] \tag{C.2}$$

$$E\left[p(c|\xi)\right] = E\left[\prod_{c=1}^{C} \xi(c)^{p(c)}\right]. \tag{C.3}$$

Now we get

$$\begin{aligned} E\left[\prod_{w=1}^{W} \phi_c(w)^{o_c(w)}\right] &= \frac{\Gamma(W\alpha)}{\Gamma(W\alpha + \sum_{w=1}^{W} o_c(w))} \prod_{w=1}^{W} \frac{\Gamma(\alpha + o_c(w))}{\Gamma(\alpha)} \\ &\propto_c \frac{1}{\Gamma(W\alpha + \sum_{w=1}^{W} o_c(w))} \prod_{w=1}^{W} \Gamma(\alpha + o_c(w)). \end{aligned} \tag{C.4}$$

and

$$\begin{aligned} E[\prod_{c=1}^{C} \xi(c)^{p(c)}] &= \frac{\Gamma(C\lambda)}{\Gamma(C\lambda + \sum_{c=1}^{C} p(c))} \prod_{c=1}^{C} \frac{\Gamma(\lambda + p(c))}{\Gamma(\lambda)} \\ &\propto_c \prod_{c=1}^{C} \Gamma(\lambda + p(c)). \end{aligned} \tag{C.5}$$

We again consider what happens if we ignore a document d and then classify it as community c. Note that when this is done, $\sum_{w=1}^{W} o_c^{-d}(w)$ in Equation (C.4) can increase by more than 1 and thus we get that $\Gamma(W\alpha + \sum_{w=1}^{W} o_c^{-d}(w))$ increases by

$$\frac{\Gamma(W\alpha + \sum_{w=1}^{W} o_c^{-d}(w) + n_d(w))}{\Gamma(W\alpha + \sum_{w=1}^{W} o_c^{-d}(w))}. \tag{C.6}$$

In addition, $o_c^{-d}(w)$ in Equation (C.4) can increase by more than 1. Here we get that $\Gamma(\alpha + o_c^{-d}(w))$ increases by

$$\frac{\Gamma(\alpha + o_c^{-d}(w) + q_d(w))}{\Gamma(\alpha + o_c^{-d}(w))} \tag{C.7}$$

and thus we get the probability of setting document d to c as

$$\left(\lambda + p^{-d}(c)\right) \frac{\Gamma\left(W\alpha + \sum_{w=1}^{W} l_c^{-d}(w)\right)}{\Gamma\left(W\alpha + \sum_{w=1}^{W} l_c^{-d}(w) + n_d(w)\right)} \prod_{w=1}^{W} \frac{\Gamma\left(\alpha + o_c^{-d}(w) + q_d(w)\right)}{\Gamma\left(W\alpha + l_c^{-d}(w)\right)}. \quad \text{(C.8)}$$