TEXT PROCESSING FOR THE EFFECTIVE APPLICATION OF LATENT DIRICHLET ALLOCATION

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Alexandra Kathryn Schofield

May 2019

TEXT PROCESSING FOR THE EFFECTIVE APPLICATION OF LATENT DIRICHLET ALLOCATION

Alexandra Kathryn Schofield, Ph.D.

Cornell University 2019

Distributional semantic models such as LDA [18] are a powerful method to extract patterns of word co-occurrences for exploration of a textual corpus. This is of particular interest to social scientists and humanists, who may wish to explore large collections of text in their fields of expertise without specific hypotheses to test. However, to use topic models effectively relies on choices about both text processing and model initialization. Without prior experience in machine learning and natural language processing, these choices may be challenging to navigate. I focus on two primary challenges in establishing datasets for effective topic models: pre-processing and privacy. In the first part, I share a number of experiments to discover the effects of standard text pre-processing steps on the learned topic models. My work shows common practices in text cleaning, including stemming, stopword removal, and text de-duplication may be less necessary than conventionally assumed. In the second part, I discuss a workflow to apply differential privacy through randomization for Poisson factorization models, a broad class of distributional models of count data including LDA. My work includes multiple methods for efficient inference of private Poisson factorization models on large datasets, including approximations to an MCMC algorithm and a new variational inference (VI) algorithm. I also discuss approaches to introduce of randomness to privatize unigram frequencies to better preserve the sparse, correlated structure of the true data.

BIOGRAPHICAL SKETCH

Alexandra Schofield was born and raised in Bellevue, Washington. Prior to her Ph.D., Alexandra received her B.S. from Harvey Mudd College in Computer Science and Mathematics with great distinction and honors in both Computer Science and Humanities. After one year as a software engineer in search quality at Yelp, Inc., Alexandra started her in Ph.D. Computer Science at Cornell University, where she was advised by Prof. David Mimno. During her time as a Ph.D. student, she received numerous awards, including the NDSEG and NSF GRFP Fellowships, the Microsoft Research Graduate Women's Scholarship, and the Anita Borg Memorial Scholarship. She has been fortunate to complete two internships at Microsoft Research during this time: one in the Machine Teaching Group in Redmond with Jina Suh and one in the Fairness, Accountability, Transparency, and Ethics group in New York City with Dr. Hanna Wallach. In Fall 2019, Alexandra will return to her alma mater to start as an Assistant Professor in Computer Science at Harvey Mudd College. When not doing computer science, she practices Aikido, bakes computationally-inspired cookies, and posts on blogs and social media about computer science education. She is also passionate about supporting women in computer science through a variety of volunteering roles.

To my father, Kevin, who inspired me to pursue computer science and supported me every step of the way.

ACKNOWLEDGEMENTS

Obtaining a Ph.D. has required a great deal of patience, perseverance, good humor, and editing. For all of these things, I would not have been able to manage without the enormous team of people who have supported me.

It would be ridiculous not to start by thanking my tireless, perfect advisor, David Mimno. Not everyone would be willing to take on a PhD student with no prior college-level coursework in statistics, machine learning, or NLP. The first paper in this dissertation was an early project I was meant to complete on my own in a month, but instead it took two years, two papers, and innumerable hours of his support and guidance. David's patience, sense of humor, and attention to the well-being of all of his students has been instrumental in my completion of my Ph.D. I like to tell people that David was my advisor from the moment I got to Cornell, and that I have not once regretted it. I am immensely grateful to have such a positive supporter, role model, and lifelong colleague in David.

Next, I would like to thank my other collaborators. Work in this dissertation was co-authored with Måns Magnusson, David Mimno, Aaron Schein, Laure Thompson, Hanna Wallach, Steven Wu, Gregory Yauney, and Mingyuan Zhou. During my time at Cornell, I have also been grateful to write and collaborate with Rishi Bommasani, Thomas Davidson, Alicia Eads, Jack Hessel, Lillian Lee, Fauna Mahootian, and Leo Mehr. Whether as colleagues, mentors, or mentees, they have widened my research horizons.

I would also like to thank my research group, including Maria Antoniak, Jack Hessel, Moontae Lee, Grant Storey, Laure Thompson, and Gregory Yauney. Drinking tea and discussing research with you all has been a privilege, and I look forward to continuing to do so for a long time yet, as I count you among my favorite friends. The Cornell NLP community has provided many guides,

particularly Cristian Danescu-Niculescu-Mizil and Lillian Lee, who mentored and challenged me in coursework and seminars to think about social processes in new ways. The many other amazing faculty at Cornell have been inspirational, but I would particularly like to thank Éva Tardos and Ken Birman for supporting me through numerous applications and efforts at my own personal development. I am also grateful to my MSR mentors, including Jina Suh, Carlos Garcia Jurado Suarez, Steven Wu, and of course, Hanna Wallach, who has cheered me on with unshaken enthusiasm from the first time she met me.

A number of friends at Cornell have supported me through thick and thin, even from a different research area. Though I'm grateful to all my fellow CIS graduate students, I'd like to specifically thank Rediet Abebe, Isaac Ackerman, Nathan Adara, Rebecca Bernstein, Ethan Cecchetti, John Eom, Molly Feldman, Jacob and Katharine Gardner, Matthew Hin, Andrew Hirsch, Michelle Kelley, Vera Khovanskya, Yolanda Lin, Andrew Loeb, Sofia Mota, Fabian Muehlboeck, Geoff Pleiss, Eston Schweickart, Tobias Schnabel, Jonathan Shi, Sharifa Sultana, Nathaniel Stetson, Wil Thomason, Anna Waymack, and Loki White. Your friendship got me through a lot of hard days.

I also want to acknowledge the amazing humans who have supported me from afar in my Ph.D. work, whether through social media or the occasional conference. While I can't hope to list all of you, I see you all and celebrate your support. Thank you for your collegiality, your trips to get coffee, your couches when I traveled, your frankness, and your comments of support in times when I was discouraged. The internet is messy, but I'm glad you're there with me.

Finally, I want to honor my family. My mother, Dr. Melissa Ganus, has been an eager and patient supporter of my journey through higher education. My father, Kevin Schofield, has been an energetic advocate for my success even

if he sometimes knows more about my research than I expect or want to talk about. My twin sister, Elizabeth Schofield, has edited, advised, and cheered from afar while pursuing her own brilliant career in education. These three have taught me so much of what I know and care about when it comes to teaching, communication, and the pursuit of knowledge. Taken with my astoundingly accomplished grandparents, aunts, uncles, and cousins, I consider myself truly lucky to have won the family lottery. Thank you, and I love you all.

The majority of my PhD was supported through a three-year fellowship from the Department of Defense (DoD) through the National Defense Science and Engineering Graduate Fellowship (NDSEG) Program, and from a two-year fellowship from Cornell University. I was also supported in part by two NSF grants, #1526155 and #1652536, a Sloan Fellowship, the Microsoft Research Graduate Women's Scholarship, and the Anita Borg Memorial Scholarship. I am grateful for travel awards through the Association of Computational Linguistics, the Women in Machine Learning workshop, and Cornell University.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In the early 21st century, ever-growing quantities of available digitized text have empowered experts experts in humanistic fields to new discoveries about human behavior. For sufficiently large text collections, however, an expert humanist or sociologist may not be able to wade through all of the relevant texts available. To cope with this challenge, scholars in these areas may apply methods from statistics, natural language processing, and machine learning can provide help in building understanding at scale, in search of broad patterns or interesting anomalies across a text corpus. At the intersection of these computational fields and the humanities is discipline of *cultural analytics*, or the study of sociocultural phenomena through the use of computational analysis of digital data.

Work in classical NLP tasks usually begins with a clearly-defined task. For example, given the text of a movie review, can one predict its sentiment or rating? Given a sentence describing an event, can you determine which words depend on which others? Given an instance of the pronoun "she" in a news article, can you determine the name of the person to whom it refers? In machine learning, these tasks are generally undertaken using *supervised learning*, in which some model is given both inputs and the correct outputs and optimizes learned parameters to get as close as possible to recovering the correct outputs for those inputs. However, digitized collections for a humanistic investigation do not always have clear labels or supervision available related to the investigator's subject of interest. Nor do the investigators themselves necessarily have a predefined hypothesis for how to pin labels to such a corpus, much less the time available to do so objectively and accurately. In these cases, *unsupervised learning* models

may be more fruitful: instead of using given labels, the model is simply given the observed text input and attempts to recover some kind of structure over that input. Though the model still has some objective, such as maximizing the likelihood of the observed text given the model or minimizing the interaction between separate components of the model, there is no known optimal structure to which the model may be compared to assess its accuracy.

Distributional semantic models are a particularly popular type of unsupervised learning for cultural analytics. These models use the statistics describing the frequencies of words and their co-occurrences in different spans of text to infer measures of similarity of meaning in documents and words. Examples of distributional semantic models include topic models [17, 18, 69, 89, 134, 154] and word embeddings [36, 38, 85, 107, 121]. Both of these types of models produce vector representations of words in a compressed numerical space such that one may compare words based on *distributional* observations of their frequencies to learn the relative similarity of those words' meaning (or *semantic* similarity) .

Not only limited vector representations of words, topic models also provide natural representations of the documents containing these words on the same latent "topic" dimensions as the words themselves. Perhaps the most popular topic modeling framework, latent Dirichlet allocation (LDA) [18], expresses both word and document vectors using a Bayesian model of distributions of words given a latent set of topics describing which words arise together. LDA has seen applications in diverse fields such as literature [58, 133], archaeology [108], classics [109], history [114], and political science [53]. The work in this thesis focuses on LDA, with a richer description of the statistical model, inference strategies, evaluations, and pitfalls of LDA models given in Chapter 2.

In order for models like LDA to help in the work of cultural analytics, these models must be easy to deploy for experts in culture. Ideally, such models should be simple to use as tools without requiring deep knowledge of how to implement their inference or the calculus behind their statistical meaning. Compared to other machine learning models of text, topic models provide straightforward affordances for interpretation: topics can be characterized by lists of their highest probability words and samples of documents in which that topic has a high proportion. Experts can then read these word lists and sample documents to determine their own label for the meaning of that topic in their corpus. None of this required deep understanding of the statistics behind the model inference, merely the ability to feed the text of interest into the "black box" topic model and store the output topics and document-topic proportions.

Even as a black box, however, developing the knowledge to deploy topic models effectively can be a challenge. Prior work already demonstrates that how an LDA model is initialized may affect the conclusions drawn: the choice of hyperparameters [165], number of topics used [10], and lengths of documents [102] can all affect inferred model quality. Particularly time consuming is the choice of appropriate pre-processing decisions for text, in which one fixes which word *types*, or unique vocabulary terms, will be considered by the model, and how to split apart *tokens*, or instances of those types. Topic modeling practitioners must determine where to segment, trim, combine, and remove words to form a reasonable vocabulary for representing the text content [21].

The need for expertise in pre-processing to learn a topic model is concerning for two primary reasons. First, the terminology associated with the choices made in processing and initializing a topic model exclude people outside of computa-

tion. When acquisition of a text collection is already a significant undertaking for many humanists due to the limitations of OCR and expense to acquire data rights, it can be a prohibitively high hurdle to learn the necessary "folk knowledge" about how to turn digitized text into a meaningful topic model input. This may have a disproportionately discouraging effect on women and people of color, who are less likely to have encountered computational terminology formally in early stages of their education [163]. Second, much of the gathered wisdom found in existing work is based on the experience of machine learning experts focused on so-called "canonical" datasets, such as 20 Newsgroups [131], Enron emails [78], or Reuters newswires [86]. Though these datasets are well-known benchmarks, and some are even used in this thesis, they alone only represent miniscule portions of the wealth and diversity of human language. Practitioners need reliable methods to obtain recommendations of pre-processing methods that extend beyond standard English or high-resource language collections into new domains with different properties.

The challenge of processing data to produce the best model results grows more complex with the need to protect the contents of these data themselves. Natural language text typically emerges from social processes, which may contain personally-identifying information, intellectual property, or simply embarrassing content that ought to be kept private. While some of these text collections may already be legally constrained through privacy policies, terms of service, or copyright, even openly available data may break reasonable privacy expectations due to poor human reasoning about privacy in an online setting [117, 161]. In these settings, it may still be beneficial to learn topic models, but it is important to find ways to do so without the models themselves revealing private information about its authors or subjects. The field of differential privacy [41] offers a

framework to provide guarantees about how much privacy is released by the introduction of random noise to computation and data queries, which can be incorporated into the inference of algorithms such as LDA [119]. When there is no trusted server on which to store data, and when it may be necessary to distribute data itself for reproducibility, it may be necessary to use the local model of privacy [169], in which random noise is added directly to the original data representations. In the case of bag-of-words features, this leaves significant questions about how to still learn a meaningful distributional model when the distributions of interest have been randomly perturbed.

This thesis is split into two parts in investigation of better methods of processing data for topic models and, by extension, other distributional semantic models. In the Part I, I consider several canonical pre-processing steps and empirically investigate their impact on the resulting learned model. After providing initial background for LDA in Chapter 2, I provide results on the effects of stemming, lemmatization, and stopword removal in Chapter 3 and of text de-duplication in Chapter 4. Though these experimental results focus on only LDA topic models on English corpora, many of these results give intuitions about distributional semantic model inference that should extend to other distributional semantic models, as well as methodologies to test collections beyond the high-resource languages considered here. This work leverages a wide array of different evaluations, including new techniques to resolve challenges in evaluation when different processing varies the underlying dataset itself.

In Part II, I investigate the problem of learning Poisson factorization models — a class of models which includes not only LDA but also network and event models — when data processing must also introduce privacy to the representations of

the text. I begin this part with a supplementary background section in Chapter 5 to describe existing privacy methods and relevant prior work in Bayesian models. I discuss the specific challenges caused by attempting to introduce local privacy to document-word occurrence statistics in Chapter 6, and consider alternative methods of adding random noise to compressed representations of these texts to produce higher quality models while preserving co-occurrence statistics. I also introduce work in Chapter 7 that builds on an early idea for inference of models under these privacy constraints [141] to make it scale to larger datasets, including modifications to an MCMC algorithm and a new variational inference protocol. I close in Chapter 8 with plans for future directions of both the pre-processing and privacy work in shaping concrete tools and workflows for novices practitioners to apply in investigation of new text collections.

Part I

Pre-Processing for Topic Models

CHAPTER 2

AN INTRODUCTION TO LATENT DIRICHLET ALLOCATION

In this dissertation, we focus on modeling topics using Latent Dirichlet Allocation (LDA) [18]. This section is intended to act as a primer to the important topics in understanding LDA. Below, I outline some of the basic structure and history of the model, common strategies of inference and evaluation, and a few challenges that may be encountered when setting up LDA inference.

2.1 The LDA Model

LDA presents a *generative model*, a type of statistical story for how data might be generated. In the domain of text, the goal is to explain the statistical frequencies of observed words in text documents. A text collection is described as having a vocabulary of possible words, called *types*, that might occur in documents. Documents are then comprised of sequences of *tokens*, or individual instances of words in use. As an example, the phrase "my cat likes my other cats" taken at face value contains six tokens, but only five word types: both instances of the word "my" are unique tokens but share a type. A generative model for texts describes a way to sample from statistical distributions in order to generate documents in a text collection as sequences of tokens.

LDA's generative model insight is that a text collection is defined by a mixture of latent *topics*, or multinomial probability distribution over words. Each document is also characterized by a multinomial distribution over topics. The name Latent Dirichlet Allocation comes from the assumptions that these latent

multinomial distributions are in turn drawn from Dirichlet distributions.[1]

The following notation for the full LDA generative model is presented below and summarized in the plate diagram in Figure 2.1:

- Conventionally, a text collection or *corpus* contains D documents (indexed $d \in 1 \dots D$), a vocabulary of V distinct word types ($w \in 1 \dots V$), and K latent topics ($k \in 1 \dots K$).
- $\vec{\alpha}$ is a length-K Dirichlet parameter. The sum of this vector is a scalar α_0; for a symmetric Dirichlet prior, the kth entry α_k is defined as α_0/K.
- β is a scalar representing the smoothing of each word in a Dirichlet prior.
- Each topic k has a categorical topic-word distribution over V outcomes $\phi_k \sim \text{Dirichlet}\left(\vec{\beta}\right)$, where ϕ_{kv} is the probability of drawing word w.
- Each document d has a categorical document-topic distribution $\theta_d \sim \text{Dirichlet}\left(\vec{\alpha}\right)$ over K outcomes, where θ_{dk} is the probability of drawing topic k. Document d also has some number of words N_d in it; this can be drawn from a Poisson distribution or assumed to be fixed.
- For each word $i \in 1, \dots N_d$ in document d, the word has topic assignment $z_{di} \sim \text{Categ}\left(\theta_d\right)$, and word type $w_{di} \sim \text{Categ}\left(\phi_{z_{di}}\right)$.[2]

LDA provides a hypothesized generative process for text data or, more generally, sequences of discrete count items. It makes the simplifying assumption that

[1] The original LDA model [18] only specified a Dirichlet prior over document-topic distributions, using a maximum likelihood estimate of the topic-word distribution instead of specifying a prior. The model with a Dirichlet prior on both multinomial distributions, referred to as "smoothed LDA" in the original paper, was popularized by the introduction of the collapsed Gibbs sampler shortly after LDA's introduction [62]. As is common, we use LDA to refer to the model with Dirichlet priors on both the document-topic and topic-word distributions.

[2] These distributions are often described as multinomial distributions, reflecting multiple draws from the same distribution. However, each z_{di} represents only a single draw, so we describe it as a categorical or Multinoulli distribution.

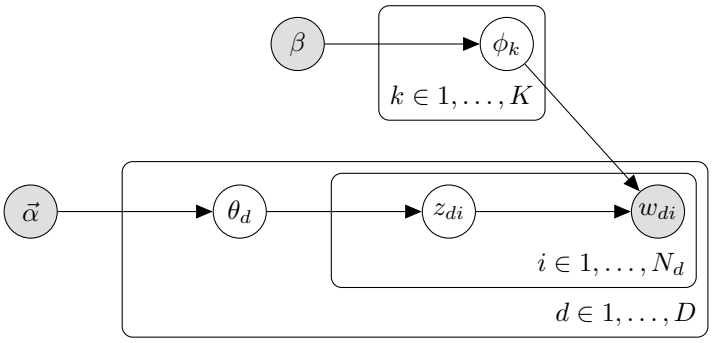


Figure 2.1: Graphical model for smoothed LDA.

drawing words randomly from probability distributions is sufficient to generate text. We refer to this as a *bag-of-words* assumption: the order of words does not effect the probability of any of the observed or latent features of these models, only the combined frequencies of each word in each document. Though using this model to actually generate text through random draws would be highly unlikely to generate meaningful prose due to its ignorance of syntax, we can learn something more meaningful by attempting to infer which topics could have generated the text we see given these probability distributions. That is, what choice of topics and topic proportions — ϕ and θ, respectively — would be most likely to generate the text we observe given this generative process?

LDA is a simplistic model of text, in which the only latent structure present is topics. However, LDA can also serve as the base of more complex models describing dynamics of time [17], author [134], syntax and word sequences [63, 164], author behavior [126], and levels of text contextuality [29]. These models may also be made hierarchical [89, 154], partially supervised [74, 127], or even automatically dimensionally reduced [72]. Some of these models may also vary distributional assumptions about text, such as using biterms instead of documents [173] or using an additive model of Gaussian topic mixtures to keep

sparse representations with respect to a background distribution [43].

2.2 LDA Inference

Even with a bag-of-words assumption, the question above of which model parameters are optimal proves to be computationally challenging to answer. First, the analytic solution for an optimal ϕ and θ cannot be solved exactly; to compute the optimal topic requires integration over ϕ and θ for topic assignments z_{di} conditioned on both, which does not have a closed form [18, 62]. This problem is not solved merely by a numerical solution, such as using gradient descent to follow the slope of an objective function to a local optimum. Topic models are *nonconvex*: there are a number of stable, locally-optimal points in parameter space describing high-probability topics and topic proportions. Trivially, if we have some optimal θ and ϕ, we can simply permute the order of topics in both sets of parameters to generate an equally likely model, providing at least $K!$ different local optima. In practice, the space is much more complicated than this, so gradient descent alone even in a stochastic form is not guaranteed to find even a particularly good local optimum, let alone the global optimum.

Instead, to solve for optimal topics and topic proportions, we use iterative methods to approximate the posterior distribution. This thesis considers two primary strategies which are arguably the most popular for LDA inference: MCMC and variational inference. Though not used in this work, there also exist two more recent spectral inference algorithms for topic models, one using the method of moments [6] and one using a selection of "anchor words" for fast topic inference [8, 9]. Additional methods also learn topic models as communities in a

network of co-occurrences [80], from information-theoretic optimization [48], or from statistically-inspired neural methods [105, 167, 176]. These models do not leverage the same core generative assumptions as LDA.

2.2.1 MCMC

A popular approach for LDA inference is the use of a *Gibbs sampler*, a type of MCMC or *Markov Chain Monte Carlo* method. Starting from a random initialization, in each iteration, every latent quantity is resampled from the posterior distribution conditioned on the current observations of all other sampled and observed quantities. After thousands of iterations over all latent quantities, this converges to estimates of samples from the true posterior distribution. The simplest version of this for implementation is a *collapsed Gibbs sampler* given by Griffiths and Steyvers [62]. Rather than resampling θ, ϕ, and z values in every iteration, a collapsed Gibbs sampler writes the z_{di} sampling formula to be conditioned only on the observations of other z variables. In each Gibbs sampling iteration, for each token in the text, a new topic assignment for the token z_{di} is sampled based on the counts of the topic assignments $z_{d\neg j}, j \neq i$ of tokens the same word type w_{di} or in the same document d:

$$P(z_{di} = k | \{z_{j \neq i}\}, w_{di}, d) \propto \frac{n_{dk}^{-i} + \alpha_k}{(N_d - 1) + \alpha_0} \cdot \frac{n_{w_{di}k}^{-i} + \beta}{n_k^{-i} + V\beta}, \tag{2.1}$$

where w_{di} is the observed word, z_{di} is the latent topic assignment associated with token i in document d, n_{dk}^{-i} is the number of tokens in document d assigned to topic k aside from token i, $n_{w_{di}k}^{-i}$ is the number of tokens with word type w_{di} assigned to topic k aside from token i, and n_k^{-i} is the total number of tokens

assigned to topic k aside from z_{di}. The relative simplicity of this sampling formula comes from the conjugacy of the Dirichlet distribution of θ and the multinomial distribution of the various counts of observed z_{ij} counts, such that their joint likelihood leads many terms to cancel. Iteration usually progresses token-by-token through the text corpus for some number of initial iterations of "burn-in" until the procedure arrives at a stationary distribution of samples from the true posterior. While in traditional Gibbs sampling, one collects a number of different samples after burn-in and aggregates over them, it is popular in LDA simply to use a single random sample of the latent variable topic assignments z_{di} to compute estimates of the document topic proportions θ and topics ϕ.

The compactness of the sampling formula in Equation 2.1 makes implementation of the iterative process straightforward: one need only track arrays storing n_{dk}, n_{wk}, and n_k. Though it is difficult to measure when this method converges, as the random samples at each iteration may still vary considerably, with a sufficient number of burn-in iterations the samples will be exact samples from the true posterior. Sampling from Equation 2.1 can also be made very efficient through the observation that the corpus-wide statistics, $n^{-i}_{w_{di}k}$, and n^{-i}_{k}, change slowly, while n^{-i}_{dk} is local to individual documents. This has produced a number of different efficient sampling strategies [87, 109, 174] which can also be used to generate parallel and distributed implementations of LDA [26, 67, 88, 113, 168].

2.2.2 Variational inference.

Variational inference or VI methods estimate the true model posterior by iteratively optimizing *variational distributions*, or distributions that approximate likelihood

for latent quantities. In the case of estimating parameters for a known structure of model, these algorithms are typically described as *variational Bayes expectation maximization*, or VBEM, as the underlying algorithm reflects the expectation maximization style of updating parameter and latent variable estimates based on estimated likelihoods from current estimates. Usually, the variational distributions are found using the *mean-field approximation*, in which latent quantities are assumed to have independent priors to ensure easy factorization of the joint model posterior to find analytic parameter updates.

To develop a variational inference algorithm requires two derivational steps. First, one must find *Q distributions* to approximate each parameter and latent variable. Second, one must derive the *evidence lower bound* (ELBO), an objective function that describes how close the posterior probability of the variational distribution is to the true distribution. Maximization of the ELBO corresponds to minimization of the KL-divergence between the variational and true posterior distributions. A coordinate-ascent variational inference (CAVI) algorithm iteratively alternates between analytical updates for the optimal parameters of the approximation (the *variational parameters*) and estimates of the true posterior latent quantities given the approximation until the ELBO converges.

Variational inference algorithms can be much more efficient for model inference than MCMC. First, variational inference tends to converge much more quickly than Gibbs sampling approaches. The ELBO itself aids this, as it should monotonically increase during inference, and can be used to track the rate of convergence. In MCMC, no such clear metric of convergence exists, and instead empirical methods are often employed to determine an appropriate number of iterations of burn-in (usually thousands or tens of thousands). Second, the

computation of parameters may sometimes be more efficient than sampling, as parameter updates are computed using a closed form instead of needing to form full probability distributions from which to draw for each latent quantity in each iteration. Third, while MCMC methods only provide samples from the posterior distribution, VI methods provide the full parameterization of their approximating distributions with expectations and variances. In contrast, MCMC requires many samples after burn-in to estimate mean and variance. This may be why LDA was originally introduced with a variational inference algorithm [18].

However, variational inference also has its drawbacks. Optimal closed-form variational parameter updates often require that the variational distribution priors are conjugate to the likelihood model, or that they belong to the same distributional family. Variational distributions are thus often restricted to *exponential family* distributions, whose probability mass functions can be written in the form

$$p(x|\theta) = h(x) \exp\left(\eta(\theta)T(x) - A(\theta)\right) \tag{2.2}$$

where θ describes the *natural parameters* or latent quantities of the distribution, $T(x)$ gives the *sufficient statistics* as some function of the observed data x, and A and h are additional functions to describe independent effects of the natural parameters and sufficient statistics on the probability.

More recent work on black-box variational inference [128] aims to avoid manual update inference; however, this method is still limited to distributions with general derivations for the black-box framework. Analytic solutions for the optimal variational parameters may also scale poorly to large datasets. Stochastic variational inference [68] provides an alternative to direct solutions by using estimates of statistics from mini-batches of observed data in place of true analytical solutions. However, stochastic VI still relies on the existence of a CAVI algorithm.

Additionally, the resulting distribution is still learned only via an approximation of the true distribution, which is known to often underestimate the variance of distributions [56]. Because of the nonconvex space and the discrete probability distributions involved in LDA, variational inference may also do a poor job of exploring the full probability space of topic distributions, leading to worse topics than observed from MCMC inference. These issues may be mitigated by borrowing ideas from MCMC; for instance, one may use a collapsed VI algorithm that only infers topic assignments [155], or apply MCMC inference within a VI update to explore probability space and improve variance estimates [136].

2.2.3 Bayesian Poisson Factorization

While LDA is named for its Dirichlet-multinomial latent structure, that same statistical generative story may also be expressed using Poisson distributions instead. This falls into the family of *Bayesian Poisson factorization models* [139]. Bayesian Poisson factorization is a generative refinement of classic Poisson factorization [59], a non-negative matrix factorization model in which some matrix or tensor of observed counts Y is derived from samples of Poisson distributions whose parameters μ are the product of low-rank latent components $\theta_1, \ldots, \theta_m$.

In Bayesian Poisson factorization (BPF), the Poisson distributed latent parameters are given an additional conjugate Gamma prior, mirroring the relationship of multinomial distributions and a conjugate Dirichlet prior. In the case of LDA, we may define the prior μ as a product of two latent parameter matrices θ_1 and θ_2: θ_1 describes the relative expectations of each topic in each document, and θ_2 describes the relative expectation of each word in each topic. These are analogous

to θ and ϕ from LDA. However, where the LDA product $\theta\phi$ gives the expectations of the probability of each word in each document as per-document stochastic vectors, $\theta_1\theta_2$ gives the expected number of instances of each word in each document. Outside of this difference, these describe equivalent generative processes, and conversions can be made between the two models through normalization with respect to document lengths. BPF also permits both MCMC and VI inference algorithms very similar to those outlined above for LDA.

2.3 Evaluation of LDA Models

As stated in Chapter 1, topic models such as LDA are unsupervised models: there are no labels or known structures that the model aims to predict for new data. Without a clear objective for what parts of the observed data are most important to recover in the latent structure, evaluation of a mode becomes nontrivial. One must consider both the power of the model to accurately reflect the data and the interpretability of how that model reflects phenomena of interest. In the work that follows, we consider three main categories of topic model evaluations: model fit metrics, coherence metrics, and information metrics.

2.3.1 Model fit metrics

Perhaps closest to the standard notion of accuracy in an unsupervised statistical model is model fit, which describes how well the inferred model represents the observed data. Canonically, a statistical model's fit is described one of two ways:

- *Likelihood* gives the combined probability of the corpus by taking the product of the estimated probabilities of each token. Because this product is extremely small, this is often computed instead as the *log likelihood*, $\mathcal{L}$:

$$\mathcal{L} = \sum_d \sum_i \log p(w_{di}|\theta, \phi, \{w_{j\neq i}\}\vec{\alpha}, \beta), \tag{2.3}$$

  where $p(w_{di}|\dots)$ represents the inferred probability of the ith token observed in the dth document given the listed model parameters. Because each probability is below 1, this will be a sum over negative numbers, with a better likelihood being a higher (or less negative value).

- *Perplexity per word* is an entropy-based measure of how many bits on average is required to represent the observed type of each word given the probability distribution of the model parameters. It can be expressed as:

$$\text{perplexity per word} = \exp\left(\frac{-\mathcal{L}}{\sum_d N_d}\right), \tag{2.4}$$

  where $\mathcal{L}$ is the log likelihood from Equation 2.3. The sign is opposite that of likelihood: a lower positive number reflects a better model fit.

On the data used for model inference, the topic proportions θ_d are inferred for a given document, which can be used for monitoring convergence of an MCMC model at inference time. However, to ideally evaluate the ability of an LDA model to generalize, one would estimate either the likelihood or perplexity on *held-out* text data, or text from the same source that was not used in inference of the topic model. Altering notation from [166], the held-out likelihood of test data Y' given model ϕ, θ, and α learned from training data Y is given as

$$P(\mathcal{Y}'|Y) = \int P(Y'|\phi, \vec{\alpha})P(\phi, \vec{\alpha}|Y)d\vec{\alpha}d\phi. \tag{2.5}$$

This can be factored into the probability of generating ϕ given the inference data and the product of the probability of generating each document given the corresponding ϕ. Though this integral is intractable, it is possible to use sampling methods to estimate these probabilities [166]. Our choice strategy for estimation of this likelihood follows the recommendation of this work, applying a left-to-right sequential Monte Carlo method[3] to iteratively resample topic assignments for tokens in order to estimate topic proportions θ_d for a held-out document d.

Describing likelihoods of models penalizes significantly for each individual token with low probability. In practice, text often exhibits *burstiness*, in which a rare word will appear either not at all in a document or several times at once [97]. Likelihood measurements may overly emphasize the likelihood of rarer words over common ones, favoring models that tend towards assigning nontrivial probabilities to all possible outcomes. In these cases, it may be preferable to use a metric like mean absolute error (MAE) to compare the average absolute difference between the observed data Y and the expected frequencies of terms acquired from the matrix product $\theta\phi$. Again, however, this requires inference to generate estimates of the topic proportions θ for each held-out document.

2.3.2 Coherence metrics

Though log likelihood describes the statistical likelihood of the topic model generating the corpus, it does not necessarily indicate whether topics match recognizable concepts when observed by a user. In fact, models with higher likelihood do not necessarily produce more coherent topics from a human perspective [27]. Change et al [27] demonstrate this result by asking humans to

[3]We use the implementation of left-to-right estimation from MALLET [100].

distinguish "intruders" in a series of questions based on an inferred model, either by identifying which word intruder from a set is not from the set of high probability terms of one topic, or by selecting which topic "intruder" is not a high-probability topic for a given document.

There are two popular automatic metrics aimed at replicating the intent of the human "intruder word" test by observing how well the most probable words of each topic reflect a single unified theme. Both of these metrics use some notion of mutual information between top words in a topic based on observations of word co-occurrence in a reference text collection.

- *Topic coherence*, proposed by Mimno et al. [111], is defined for a given topic k, a list of the top M words of a topic $v_1^k, \ldots v_M^k$, and some reference text collection as

$$C(k) = \sum_{m=2}^{M} \sum_{l=1}^{m-1} \log \frac{p(w_l, w_m)}{p(w_l)} \tag{2.6}$$

  where $p(w_l, w_m)$ is the smoothed probability that words w_l and w_m will occur in the same document in the reference text collection.

- *Normalized pointwise mutual information* [3, 20, 83] has a similar form to topic coherence. However, it normalizes this expression using both independent word probabilities and their joint log probability:

$$NPMI(k) = \sum_{m=2}^{M} \sum_{l=1}^{m-1} \frac{\log \frac{p(w_l, w_m)}{p(w_l)p(w_m)}}{-\log p(w_l, w_m)}. \tag{2.7}$$

In both cases, the reference corpus used to determine $p(w_l)$ and $p(w_l, w_m)$ may be either the same corpus as was used for training, or some external reference such as Wikipedia. In practice, Mimno et al. [111] suggest that using the training data is often effective to capture the domain-specific co-occurrence information

about the vocabulary. In both cases, it is beneficial to use a sliding window over the text instead of full documents to compute $p(w_l)$ and $p(w_l, w_m)$.

2.3.3 Information metrics

Another relevant category of evaluations of topic models comes from information theory, with a focus on how much information topics provide with respect to what is already observable about the corpus. For instance, it may be useful to measure entropy of a distribution directly to see if it has has learned a predictable structure [157], or to measure pointwise mutual information of two types of inferred variables to see if they are connected. One particularly useful metric for comparing partitions of data points into clusters is *variation of information* [103] or VOI,[4] which can quantify how much two different sets of clusters of the same data points agree on which points should be grouped together in a way independent of the number of clusters. For two partitions $S = \{S_1, \ldots, S_\ell\}$ and $T = \{T_1, \ldots, T_m\}$ over a set of n data points, this is given as

$$VOI(S,T) = \sum_{i=1}^{\ell} \sum_{j=1}^{m} \frac{|S_i \cap T_j|}{n} \left(2\log(|S_i \cap T_j|) - \log(|S_i|) - \log(|T_j|)\right). \tag{2.8}$$

VOI bears similarity to the mutual information (MI) of partitions S and T. Unlike MI, though, VOI is a proper metric which obeys the triangle inequality.

Another way to express quality is to compare them to degenerate classes of topic which do not help shed light on structure within the corpus. AlSumait et al. [5] introduce three types of "junk" topic that might arise, summarized below:

[4]While variation of inference is typically expressed as VI, we write it as VOI to avoid confusion with variational inference, which is also often expressed as VI.

- *Uniform* topics are those whose topic-word probability distribution ϕ_k is close to a uniform distribution over the vocabulary. For some topic k, the extreme case would assign equal likelihoods to each word type:

$$\forall w.\phi_{kw} \sim \frac{1}{V}. \tag{2.9}$$

  These topics fail to recover any specific content correlated with a topic.

- *Vacuous* topics are similar to uniform topics, but instead of a uniform distribution, the observed distribution over word types ϕ_k is close to the observed distribution of unigrams across the whole corpus. If n_w denotes the total number of tokens of type w in the collection, an vacuous topic is:

$$\forall w.\phi_{kw} \sim \frac{n_w}{V}. \tag{2.10}$$

  These topics may initially appear more coherent than uniform topics, as the most probable words in a text collection may be thematically related, but they still fail to reveal new structural information about the collection.

- *Background* topics, unlike the other two classes, are defined by their topic proportions across documents θ_k. Concretely, a background topic has close to the same probability mass in each document in the collection:

$$\forall d.\theta_{dk} \sim \frac{\alpha_k}{\alpha_0}. \tag{2.11}$$

  While near-vacuous may contain unique content compared to other topics, they do not meaningfully distinguish one document or context from another. These topics may describe the "background" distribution of terms in a corpus, the shared language and terminology associated with the collection's domain that one expects in every document.

The original presentation of these three distributions describes a topic significance rank score, or TSR, as a weighted combination of several dissimilarity

measures across all three of these metrics that can be used to rank topics in a given model [5]. This weighted scheme is meant to ensure that each dissimilar measure and type of junk distribution is weighed equally in its contribution to a single score. In practice, this score is somewhat challenging to reproduce, and it is often beneficial to know what kind of junk topic one might have. This can be done by simply examining the KL-divergence of each topic from the extreme distributional cases given by Equations 2.9, 2.10, and 2.11 above. We can also use this intuition to describe other distributions over documents or terms that we find uninformative in order to compare inferred topics to that condition.

2.4 Practical Concerns in Topic Models

Existing implementations of efficient LDA inference [82, 100, 130] make it possible to perform topic model inference over novel text collections without having to re-implement inference. However, even with these tools available, providing the correct inputs to these models may still prove challenging for a number of reasons. I outline some of the common challenges faced in setting up inference of an LDA model below. While each of these concerns may have solutions available to address it, it can be difficult to distinguish which of these concerns, if any, might have triggered an apparently uninformative topic model.

Irregular text. The development of a vocabulary for a text collection and a reasonable tokenizer to place words in that vocabulary is an crucial step in inference of a model. Methods of vocabulary curation are discussed in more detail in Chapter 3, but a challenge in this process is that digitized data is quite often

messy. Social media data is rife with new words and slang, code switching, and typos, all of which can complicate the definition of a meaningful vocabulary of words [115]. Some archival text collections may have more editorial oversight, but the conversion of scanned images to text through optical character recognition (OCR) is imperfect, and may produce both systematic and random errors in transcription of text to a digital format. In both of these cases, it can take considerable manual effort to correct the text sufficiently for computational work. A significant amount of prior effort has been invested into statistical and rule-based methods to correct text after OCR, such as work by Tong [159] and Reynaert [132]. Though LDA models may be hard to interpret over irregular text, the models themselves can help to discover systematic issues in a collection, as text in a different language than the supposed language of the collection or a repeated orthographic error will often emerge as a distinct topic.

Dirichlet priors as hyperparameters. In LDA, topic proportions are expected to be *sparse*, with most of the probability mass in some document-topic distribution θ_d assigned to a small number of topics in each document. This desirable property is specified in the model through Dirichlet hyperparameter $\vec{\alpha}$. This hyperparameter vector, often referred to as a Dirichlet prior or pseudocounts, can affect both the relative likelihood of each topic with respect to the others and the combined tendency of the model towards sparsity. This effect can be intuited through examination of the topic term in Equation 2.1:

$$\theta_d k \propto \frac{n_{dk}^{-i} + \alpha_k}{(N_d - 1) + \alpha_0}.$$

In the symmetric case, if for all k, α_k is the same, then absent the observed

topic assignments n_{dk}, topics are equiprobable. However, on observing these assignments, the magnitude of the prior for a topic α_k becomes important. If α_k is much smaller than 0, then topic k is very unlikely to be sampled from this distribution unless some word in the document is already assigned to that topic. This encourages sparse distributions: in a Gibbs sampler, as sampling proceeds, any topic that ceased to be observed in the document would be unlikely to be sampled unless it is for a word often assigned to that topic in the rest of the corpus. In contrast, if an α_k is on the order of N_d, then unless the document is quite long, resampling from this distribution will usually look similar to a uniform distribution even when most words in document d are assigned to a single topic. The relative values of α_k are also meaningful, as a higher value of α_k will give a topic a larger probability of being sampled when $n_{dk} = 0$.

Work by Wallach et al. [165] demonstrates that for document-topic distributions, it can be beneficial to use an *asymmetric* $\vec{\alpha}$ hyperparameter, or one where not all α_k values are equal. This work suggests initializing $\vec{\alpha}$ with a sparse symmetric value (e.g., 0.05), then optimizing the values asymmetrically periodically starting partway through model inference. This permits some topics to be more prominent than others in the corpus, which has positive effects like consolidation of the background distribution of words as described in Equation 2.11 into a single topic to avoid dilution of other topics. The authors recommend a sparse, symmetric β prior (e.g., 0.01) over the topic-word distributions, as asymmetric β may cause topics to overfit to global word frequencies, effectively encouraging topics close to the vacuous distribution described in Equation 2.10.

Tools for topic model inference such as MALLET [100] and gensim [130] support both asymmetric Dirichlet priors and automatic inference of these priors.

However, the defaults for these models are symmetric, and the specific correct way to initialize automatic inference or asymmetric priors is often difficult to find in documentation; for instance, in MALLET, one must specify the "optimization interval" in order to enable hyperparameter inference. Determining how to best initialize these parameters can be challenging for a newcomer to topic models.

Data density. Bag-of-words features can be represented using a matrix of count observations, with each row corresponding to a document, and each column corresponding to a unique word type in the corpus vocabulary. To infer meaningful topics from these features relies on a tradeoff between data sparsity and correlation in this matrix. To analyze this, we consider again the LDA collapsed Gibbs sampling formula given in Equation 2.1.

The sampling equation has two topic-specific terms of interest: $n_{dk}^{-i} + \alpha_k$ and $n_{k_w}^{-i} + \beta_w$ The term $n_{dk}^{-i} + \alpha_k$ signifies the probability of a word belonging to topic k given observations of that topic in its document d. If documents are too short, the model may quickly assign all tokens in a document to one topic k $n_d = n_{dk}$. If this occurs early in inference, documents may adhere to their early topic distributions, which can impede the model's ability to learn topic distribution distinctions between words that co-occur in documents. If documents are too long, this term may push the sampler towards the uniform distribution, making it challenging for the model to converge to meaningful topics. This becomes particularly important in interaction with another term, $n_{k_w}^{-i} + \beta_w$, which signifies the probability of a word belonging to topic k given observations of the same word w across the corpus. For this second term to be meaningful, the word must appear frequently enough to have variation across k observations of n_{k_w}, but not so frequent that the word is inferred to be distributed evenly across all topics.

Overly long documents and overly frequent words may slow convergence and produce poor topics; overly short documents and rare words may too quickly converge and produce similarly poor topics.

The next chapter explorers this subject further through consideration of how choices of vocabulary shape model inference.

CHAPTER 3

VOCABULARY CURATION IN TOPIC MODELS

Topic models are known to be sensitive to pre-processing because of their dependence on a sparse vocabulary [76]. In practice, however, pre-processing methods are often neither described in detail nor justified, which can inhibit reproducibility [47]. Drawing inspiration from prior studies of how stemming and stopword removal affect other text mining tasks and models [65, 64, 75, 104, 129, 147], this work[1] applies rule-based stemmers and classic stopword removal methods to a variety of corpora to test their effect on topic models. This work specifically focuses on the development of quantitative comparisons of different pre-processing treatments that respect how pre-processing biases standard evaluation metrics. In order to study the relationship between text treatments and models, we compare models of pre-processed text to those where processing was applied *after* training. Because typical LDA evaluations are sensitive to vocabulary reduction and corpus modification, we present new and modified metrics to evaluate topic model quality in the presence of such confounding factors. We find that in pre-processing for vocabulary curation, less is often better: many steps may be performed post-hoc on the corpus without risking a potential adverse effect on model inference. Though focused on English morphology, these results and experimental methodologies should help guide future researchers as to how to select stemmers and stopword removal methods for a given task and corpus.

[1] The work of this chapter reproduces text and results from two existing published papers on pre-processing [143, 144].

3.1 Mechanisms of Vocabulary Curation

In English, the fields of information retrieval and natural language processing have provided a variety of different methods for vocabulary curation, or the selection of which terms to include in a model vocabulary. This section outlines the standard methods of English vocabulary curation used in this study.

3.1.1 Stemming and Lemmatization

Stemming is a popular way to reduce the size of a vocabulary in natural language tasks by conflating words with related meanings. Specifically, stemming aims to convert words with the same "stem" or root (e.g "creative" and "creator") to a single word type ("create"). Though originally developed in the context of information retrieval (IR) systems, stemmers are now commonly used as a pre-processing step in unsupervised machine learning tasks. Although stemmers are commonly used prior to topic model inference [92, 93, 112, 77, 150, 73], these works rarely indicate the reasoning behind the choice to stem.

One could devise several reasons to stem prior to inference of a topic model. First, conflation of semantically related words could improve model fit by intelligently reducing the space of possible models. Given that randomly reducing the feature space is already known to be potentially beneficial in other machine learning applications [50], doing so in an etymologically-tuned way might be even better. Second, stemmers could reduce the effect of small morphological differences on the stability of a learned model. As an example, reducing the words "happy", "happily", and "happier" to one token may result in fewer

possible models with divergent topics that signify happiness or related themes. Third, stemmers approximate intuitive word equivalence classes, so language models based on stemmed corpora inherit that semantic similarity, which may improve interpretability as perceived by human evaluators.

This work focuses specifically on the limited area of English morphological conflation. This choice of limitation has several reasons: English is a standard benchmark language, and while it has some diversity of rules due to its many source languages, most English is morphologically simple. Nonetheless, the language has encouraged a diverse ecosystem of different stemming and lemmatization procedures over the history of information retrieval. The methodologies used for evaluation here, however, only depend on the existence of some meaningful tokenization scheme, and thus could be extended to any language. Work on how best to infer LDA for morphologically richer languages such as Russian [99] and Hebrew [44] encourages morphological conflation but also indicates the complexity of evaluation of their effects in the context of topic models.

English stemmers have particularly strong potential to be confusing, unreliable, and possibly even harmful in language models. Many stemmers produce terms that are not recognizable English words and may be difficult to map back to a valid original word, such as "stai" as the Porter stem of "stay". Although stemming aids document retrieval for many languages, English is a notorious exception [65]. The complexity of compound affixes with meaning can lead to over-stemming: for instance, the words "recondition" and "recondite" have distinct roots, but under a Porter stemmer, they share the same stem, "recondit." These complexities can also lead to the incorrect conflation of words with the same etymological root but divergent meaning, such as "absolutely" and "ab-

solution." Third, and most troubling, there are cases in which morphological variants of the same stem carry significantly different meanings. Conflating "apple" and "apples" should occur in every canonical morphological treatment of English, but loses the distinction between a device manufacturer and a type of fruit. Similarly, "god" and "gods" may be another obvious case of singular and plural, but belong to vastly different contexts in terms of the religions they imply.

In this work, we consider nine different methods of English word normalization, given below with two-letter labels. This includes five popular rule-based stemmers, two baseline stemmers, and two morphologically aware treatments. We will sometimes use the more general term *conflation treatments* or simply *treatments* to refer collectively to stemmers and lemmatizers. These treatments are compared to the control, "no-stemmer" treatment, **NS**.

Rule-Based Stemmers

Rule-based stemmers are methods governed by a set of rules that convert one affix to another. Most classic stemmers fit into this category, including the famous Porter stemmer. These methods are quick, but also limited: no concise rule set captures every English morphological exception. The systems also cannot use the surrounding context to resolve ambiguous word types. Instead, they are consistent for each token independent of context: if word type a maps to stem b in one location, it will do so in every location that word type a arises. Treatments of this type are effectively equivalence relations over words, with a *conflation class* being an equivalence class of word types under a conflation treatment T. A small example of such rules is shown in the S stemmer [65] in Table 3.1.

If word ends with:	...and does not end with:	...replace this ending with:
-ies	-aies, -eies	-y
-es	-aes, -ees, -oes	-e
-s	-ss, -us	-

Table 3.1: The "S" stemmer consists of three simple rules in order. Only the first rule applicable in the first column is applied.

Jivani [75] refer to these as "truncation stemmers" or "affix removal stemmers." However, this naming may be confusing: stemmers rarely strictly truncate, and almost all stemmers aim to remove affixes. Instead, the term "rule-based stemmers" reflects the core similarity of these methods: all of the language-specific information used is encoded directly into the stemmer's rules. This work applies five different algorithms for rule-based stemming, given below.

Truncation Stemmers. k-truncation stemmers [14] remove all but the first k characters of a word. As an extremely fast, language-ignorant, high-strength method, raw truncation serves as a good baseline of the relative effect of even the most basic vocabulary reduction on model evaluations. This work compares both five-truncation (**T5**), which has strength close to a strong rule-based stemmer, and four-truncation (**T4**), which is the shortest truncation option for English where the resulting conflation rules still often correlate with word roots.

"S" Stemmer. The S-removal stemmer or "S" stemmer (**SS**) removes S-based endings using only three rules. Harman [65] introduced the "S" stemming algorithm as a simpler counterpoint to more standard rule-based stemmers. The rules listed in Table 3.1 are good representatives of both the typical syntax and morphological impact of rules employed by the other stemmers in this section.

Lovins Stemmer. The Lovins stemmer (**LS**) is a rule-based stemmer that applies a two-step stemming algorithm [94]. Although each steps iterates through a long list of rules, the method is still fast and easy to implement. It is among the oldest regularly-deployed stemmers and one of the strongest named stemmers.

Porter and Porter2 Stemmers. The Porter stemmer [122], one of the most popular in current use, is a slightly less strong and more intricate stemmer than the Lovins stemmer. It uses five phases of rules and conditions that can match not only exact letter sequences, but also more general patterns of vowels and consonants. Porter later created a slightly improved version of the Porter stemmer for Snowball, a programming language for rule-based stemmers [123]. As both stemmers are still regularly used, this work evaluates both the original Porter stemmer (**P1**) and the modified Snowball version (**P2**).

Paice/Husk Stemmer. The Paice/Husk stemmer (**PH**), or Lancaster stemmer, iterates indefinitely over the same rule list. To constrain iteration, some rules can only apply to unmodified words, while others always terminate iteration [116]. More concise in rules but more complex in implementation, the Paice/Husk stemmer balances out to be similar to the Lovins stemmer in strength.

Context-Based Treatments

While the methods above are fast, they are imprecise, as a limited set of rules cannot account for all possible morphological exceptions. Subtleties, such as the difference between the gerund "frosting" as applied to windows and cake "frosting," are lost without contextual information. The methods below use tools such

as dictionaries, inflectional analysis, and part-of-speech inference to determine the correct conflated form of a word. As such, they may not consistently reduce the same word type to the same form. However, these tools also demand more computational resources; for the corpora used, WordNet lemmatization of each corpus consistently took more computational time than training the topic model even when executed in parallel across multiple fragments of the text.

Krovetz Stemmer. The Krovetz stemmer [79] (**KS**) uses inflectional analysis and a dictionary to determine correct forms of words before removing word endings. Though this process is complex, the stemmer itself is weak, as it focuses only on normalizing verb forms and removing pluralization instead of navigating morphology that might transcend parts of speech. The dictionary itself is crucial for implementation; this work uses the Lemur Project implementation.[2]

Lemmatizer. Lemmatizers use a database of lemmas, or standardized word forms, in order to find the best normalized word form for a given token. While the method is orders of magnitude slower than rule-based stemmers, it is also much more principled and extremely unlikely to over-conflate. This work uses the popular WordNet-based lemmatizer (**WL**) implemented in the Natural Language ToolKit [15]. This lemmatizer relies on part of speech information, which is provided here via the Stanford POS Tagger [160] applied to the unmodified text.

The comparative strength of these treatments, evaluated based on the change in vocabulary size and average word length, is summarized in Figure 3.1 for four corpora. Character-token ratios vary less between corpora than type-token

[2]The source code contributed by David Fisher and is available at `https://sourceforge.net/p/lemur/wiki/KrovetzStemmer/`.

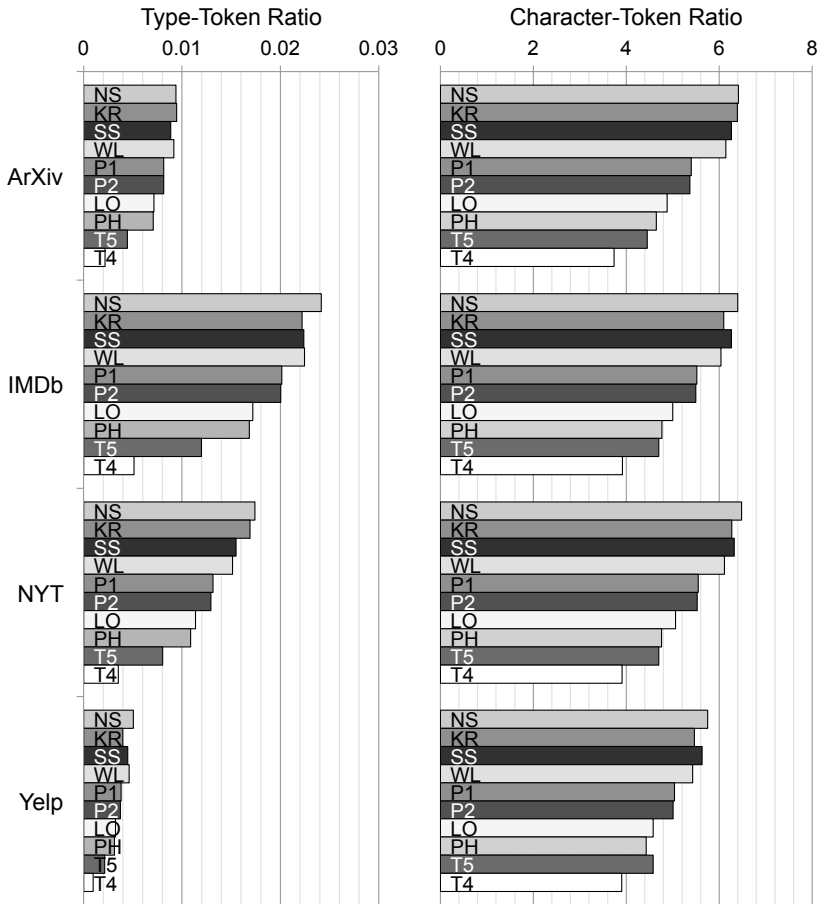


Figure 3.1: Type-token ratio and character-token ratio vary substantially across training corpora and conflation treatments. Due to the context-sensitive stemming done by the Krovetz stemmer, one untreated word type may map to multiple stemmed types, producing a greater type-to-token ratio for the ArXiv version of the Krovetz stemmer than for the original untreated corpus.

ratios. These results demonstrate that stemmer strength can depend heavily on the type of corpus on which it is applied. For instance, the Krovetz stemmer actually increases the size of the vocabulary of ArXiv, whereas it produces more vocabulary reduction than the lemmatizer on both IMDb and Yelp. The three different light stemming methods — the S-stemmer, the Krovetz stemmer, and the WordNet lemmatizer — perform similarly on the IMDb corpus, but vary

substantially across the other three corpora. Though both truncation methods are quite strong in terms of the resulting type-token ratio, **T5** produces average word lengths close to those of the Paice-Husk stemmer.

3.1.2 Stopword Removal

In LDA, a common pre-processing step is the removal of *stopwords*, or very frequent words not specific to a subject such as "the", "and", or "she". Language-specific canonical stopwords can be compiled into shared *stoplists* of words, which will provide a guide for tokens to ignore in a later task. Stoplists may also be corpus-specific: for instance, "government" might be considered a stopword in a corpus of political speeches. Corpus-specific stopwords can be identified as word types that occur above a threshold proportion of documents in the corpus. The assumption behind stopword removal in topics models is that, with stopwords present, we will not be able to learn as high-quality a language model. In the corpora tested, a preset list of approximately 500 stopword types accounts for 40-50% of the corpus. If these words uncorrelated with topics, one would expect stopwords to only hinder inference of meaningful topics.

The use of stoplists comes with several costs in both effort and persuasiveness. Constructing a good stoplist is difficult and time consuming, and often cannot be transferred to new corpora. Custom stoplists can also call into question the validity of a model: if an analyst is too aggressive in removing words, the resulting models may be biased towards what the analyst views as important in a corpus. Finally, while removing stopwords appears to produce more interpretable topics, this effect may be an illusion. As topic interpretability is typically judged by the

most frequent terms in the topic, post-hoc stopword removal from a model can substantially increase interpretability without modifying the model.

LDA may sometimes partially accommodate separating out stopwords without explicitly removing them. Wallach et al. [165] show that a parsimonious asymmetric Dirichlet prior inferred for θ allows model inference to isolate stopwords into fewer low-quality topics, leaving the remaining topics largely unaffected. These low-quality topics resemble the background distribution and vacuous distribution of AlSumait et al. [5] and summarized in Section 2.3. However, even if most of a stoplist is sequestered in such a topic, the most common stopwords, such as "the" in English, are so frequent they are still likely to be prominent in many topics. The latter set of extremely frequent terms may overwhelm the model and reduce how well the model fits topic-oriented terms.

This work assesses three plausible hypotheses about what effect stopwords have on the inference of coherent and high-likelihood topic models:

1. **Stopwords harm inference.** Noise from frequent words prevents the algorithm from recognizing patterns in content-bearing words.
2. **Stopwords have no effect on inference.** Noise from frequent words does not meaningfully alter inference on non-stopwords.
3. **Stopwords improve inference.** Topics assignments inferred for frequent words echo and reinforce patterns in content-bearing words.

To assess these, we consider both the removal of stopwords based on canonical stoplists (GENERIC) and based on a threshold of document frequency (IDF). As with stemming, the evaluations applied are based on English text collections: one of State of the Union speeches, and one of New York Times article text[137].

3.2 Evaluation Practices for Vocabulary Changes

In order to evaluate the differences between morphological conflation treatments and stopword removal treatments, we want to consider a variety of different types of evaluation of topic models. Unfortunately, standard evaluations of topic quality such as held-out likelihood and coherence are implicitly affected by the size of the vocabulary and number of tokens retained of a text collection. Human judgment also prefers topics that do not appear to have redundant or inconclusive information in its topic summaries. Consequently, both automatic and human comparison using classic versions of these evaluations across different processing treatments will often automatically favor more heavily processed text. To be able to compare different treatments without simply favoring the maximum possible vocabulary or corpus reduction, this work introduces modified versions of several existing classic evaluations, as well as new metrics for understanding differences in models at the level of word types instead of topics.

3.2.1 Comparing Pre-Processing and Post-Processing

One of the reasons classic quantitative evaluations fail for pre-processing is that reducing vocabulary size critically alters probability distributions over word types. To avoid biasing towards the smaller vocabularies of stronger pre-processing, this comparison uses a post-hoc transformation to ensure the compared corpora are equivalent. This is achieved by taking the assignments of tokens to topics, z_{di}, from different treatments and applying them to the corresponding tokens in some shared reference version of the corpus. When inference is performed with a collapsed Gibbs sampler, these z_{di} assignments

and the model hyperparameters are sufficient to specify the full parameters of a model. The resulting models are therefore topic models of the same underlying processed corpus, and may be compared more directly.

For conflation treatments, the reference version of the corpus is the no-stemmer (NS) treatment. For each other treatment, the pre-processing workflow in Tables 3.3 and 3.4 is applied to all of the text prior to inference of a topic model. After inference, each stemmed token is replaced with its unstemmed form. This allows us to observe whether conflation treatments map tokens to the same topic in a more coherent way than untreated corpora would. For stopword removal, the reference version of the corpus is one with stopwords removed. In models inferred with stopwords still present, topic assignments to stopped words are discarded. This evaluation strategy may be used to compute both likelihood metrics and coherence metrics of topic models, as described in Section 2.3.

3.2.2 Normalized Per-Token Likelihood

Strong stemmers can improve the joint probability of documents occurring without improving the quality of the model. As we reduce the size of the vocabulary, each topic-word distribution is spread over fewer possible words. At its extreme, the probability of any corpus under a zero-truncation stemmer (where each word type is reduced to an empty string) would be 1.0, as all tokens would be conflated. For morphological conflation, the standard held-out likelihood score $\mathcal{L}$ of the test corpus based on the trained model orders stemmers exactly by how much they reduce the vocabulary, assigning the highest likelihood to those treatments with the smallest vocabularies. This includes

strong performance from both truncation stemmer treatments T4 and T5, even though neither of these methods has explicit knowledge of English affixes.

To account for the likelihood improvement caused by reducing vocabulary size, we normalize a model with K topics by the likelihood of a smoothed unigram language model with the same β parameter. We calculate from the normalized log likelihood $\mathcal{L}_{\text{norm}}$ a normalized per-token log likelihood, or negative per-token perplexity $\text{PTLL}_{\text{norm}}$, to put corpora of different lengths on a comparable scale. We compute the unigram model probability as a smoothed multinomial with prior β, number of instances of word type w in a corpus n_w, vocabulary size W and total number of tokens in the corpus N:

$$\mathcal{L}_{\text{unigram}} = \prod_j \prod_i \frac{n_{w_{ij}} + \beta}{N + W\beta} \tag{3.1}$$

$$\mathcal{L}_{\text{norm}} = \mathcal{L}/\mathcal{L}_{\text{unigram}} \tag{3.2}$$

$$\text{PTLL}_{\text{norm}} = \frac{\log(\mathcal{L}_{\text{norm}})}{N} = \frac{\log \mathcal{L}}{N} - \frac{log(\mathcal{L}_{\text{unigram}})}{N}. \tag{3.3}$$

Our resulting metric measures how much the introduction of multiple topics improves the probability of each token occurring. One can think of this as normalizing by the relative benefit provided by the change in global distribution of words independent of whether local distributions still capture something meaningful. In this sense, this metric measures not the total overall ability of a topic model to fit its respective data, but rather the improvement that a conflation treatment gave to the fit of multiple topics in its automatic combining of words.

3.2.3 Influential Words

The evaluations discussed in Section 2.3 are summary statistics, or quantitative measures that attempt to provide concise comparable values for overall topic model quality. However, to understand why these metrics differ across treatments, we also need some way to examine the individual components we have affected: the word types available in our documents.

We use two heuristics to identify words that are most affected by a given morphological treatment. The first uses inferred token probabilities in the test corpus to determine if the estimated joint probability of tokens of a particular pre-treatment type increases after treatment. Applying notation from Section 2.1, for a given word type $v \in V$ from the untreated corpus and some conflation treatment $t(\cdot)$, we compute the *word type probability* TP_{wt} as

$$\sum_{d=1}^{D} \sum_{i=1}^{N_d} I[w_{di} = v] \log(P(t(w_{di}) = t(v)| \dots)), \tag{3.4}$$

where $P(t(w_{di}) = t(v)| \dots)$ is the estimated held-out likelihood of treated token $t(w_{di})$ having type v given the model parameters.

We average the quantity in Equation 3.4 across all topic models of the same corpus, topic count, and treatment to get $\overline{TP}_{wt}$. In order to compute a relative score of the amount of probability improvement of an individual treatment for a word type from the no-stemmer treatment t_0, we take the difference between topic probabilities, weighted by inverse document frequency (idf) to favor words that are specific to particular documents. Our final score function is

$$TPscore_{wt} = (\overline{TP}_{wt} - \overline{TP}_{wt_0}) \log \left(\frac{D}{D_w} \right), \tag{3.5}$$

where D_w is the number of documents containing at least one token of type w. Lowe negative scores indicate that the unstemmed form of the token had

higher probability and importance to the model, while high positive scores indicate higher probability and importance of the stemmed form. While this does not produce a symmetric distribution, as a smaller vocabulary increases the probability of every word, the heuristic provides a sorting order for word types based both on probability of occurring has changed between treatments and how much that word affects the corpus as a whole.

The second heuristic tests whether stemming increases or decreases certainty of the topic assignment for each stemmed word type. In this case, for a given word type w, we use the topic assignments from the final iteration of Gibbs sampling to compute the number of instances of w assigned to each topic k. To preserve the sparsity inferred by the algorithm, we use this to generate a maximum-likelihood estimate of the probability distribution of w being assigned to each topic, from which we can compute the Shannon entropy:

$$H_{wt}(k) = -\sum_{k=1}^{K} \frac{N_{wk}}{N_w} \log\left(\frac{N_{wk}}{N_w}\right), \tag{3.6}$$

Intuitively, good conflation should reduce the information entropy across tokens of a given conflation class by treating them as a single word in inference.

For each treated form of a word w by a treatment t, we also consider the inverse image $t^{-1}(w)$, or the set of all words that stem to have form w. We therefore compute a change in entropy using average $\bar{H}_{wt}$ across all trials with treatment t and control t_0 for a given corpus and topic count,

$$\Delta H_{wt}(k) = \bar{H}_{wt}(k) - \bar{H}_{t^{-1}(w)t_0}(k), \tag{3.7}$$

where $\bar{H}_{t^{-1}(w)t_0}$ is the information entropy for the topic-word counts summed across all untreated types that conflate to type w under treatment t.

3.2.4 Topic-document mutual information

The three stopword removal hypotheses described in Section 3.1.2 focus on differences between the topic distribution of stopwords in a given document and the topic distribution of content-bearing words in that document. One way to assess this effect in a model is to study the mutual information between documents and topics. Using the topic assignments of tokens inferred via Gibbs sampling, we can examine the mutual information between the document-topic distribution and the topic assignment of the token. We compare the $MI(d, k)$ before and after stopword removal to measure the effect of removal on the posterior. If there is no semantic information in a set of tokens (such as stopwords) the $MI(d, k)$ should be close to 0. If the stopwords have a negative effect on inference (hypothesis 1) removing these words before inference (*pre*-removal) should result in a higher $MI(d, k)$ than removing them afterwards (*post*-removal). The opposite should be true if stopword improve inference (hypothesis 3).

3.2.5 Classification with key terms

A metric of the quality of representative terms for a topic is their ability to identify documents with a high proportion of that topic. Inspired by the approach of Dredze et al. [40], we use classification of documents by predominant topic to assess how well key terms represent their corresponding topics. We train multinomial naïve Bayes models with the token counts of top representative terms as features and the most present topic of each document as labels.

3.3 Experiments

In all experiments, we train topic models using MALLET [100] for number of topics K = 10, 50, and 200. To ensure results are consistent across different random initializations and samples, nine to ten models are inferred for each combination of corpus, pre-processing treatment, and number of topics K.

3.3.1 Stemming

In order to test the various word normalization treatments, we used an existing Python library for the Lovins, Paice/Husk, and both Porter algorithms [28], modified to correct errors in implementation. We implemented our own truncation stemmers and S-removal stemmer. We applied each stemmer to each word token in four corpora: articles from ArXiv in early 2015,[3] articles from The New York Times in 2007 [137], biographies from IMDb,[4] and reviews from the Yelp Dataset Challenge.[5] Corpora were partitioned into 75% training documents, 25% test documents and lower-cased before conflation, which was performed per-sentence on lower-cased text. After treatment, we remove stopwords, digits, and punctuation. Table 3.2 shows details of the corpora, and Tables 3.3 and 3.4 shows example processes for the application of each treatment.[6]

	Training Data		Evaluation Data	
Corpus	# docs	# toks	# docs	# toks
ArXiv articles	17.1K	58.4M	5.7 K	19.5M
IMDb bios	84.6K	9.13M	28.2K	3.05M
NYT articles	29.4K	8.81M	9.79K	2.98M
Yelp reviews	844K	43.1M	281K	14.4M

Table 3.2: Training and test corpora for morphological conflation represent considerable variance in content, size of corpus, average length of document, and proportion of training to test data.

Original	This location does not have good service. Went through drive-through and they forgot our drinks and our sides. While they were preparing what they forgot, we could see another girl who had her back to us and it was obvious that she was on her phone. Any other KFC would be better.
Tokenized	this location does not have good service went through drive through and they forgot our drinks and our sides while they were preparing what they forgot we could see another girl who had her back to us and it was obvious that she was on her phone any other kfc would be better
Stopped	location good service drive forgot drinks sides preparing forgot girl back obvious phone kfc

Table 3.3: Sample steps in the stemming/morphological treatment experiments for tokenization and stopword removal on a Yelp review.

Held-Out Likelihood

Using our normalized log likelihood measure from Equation 3.3, we can compare likelihoods across all our different treatments, as shown in Figure 3.2. With normalization, we observe that all standard rule-based stemmers provide little likelihood benefit. Both Porter stemming treatments have significantly lower normalized log likelihoods than the unstemmed treatment. Statistically, the Porter stemmer is not helping to fit the model better; they are merely reducing

[3]Retrieved from ArXiv (`http://www.arxiv.org`).
[4]Courtesy of IMDb (`http://www.imdb.com`).
[5]Retrieved from Yelp (`http://www.yelp.com/dataset_challenge`).
[6]Our code can be found at `https://github.com/heraldicsandfox/stemmers`.

NS	location sides	good preparing	service forgot	drive girl	forgot back	drinks obvious
T4	loca side	good prep	serv forg	driv girl	forg back	drin obvi
T5	locat sides	good prepa	servi forgo	drive girl	forgo back	drink obvio
LO	loc sid	good prepar	servic forgot	dr girl	forgot back	drink obv
P1	locat side	good prepar	servic forgot	drive girl	forgot back	drink obviou
P2	locat side	good prepar	servic forgot	drive girl	forgot back	drink obvious
PH	loc sid	good prep	serv forgot	driv girl	forgot back	drink obvy
SS	location side	good preparing	service forgot	drive girl	forgot back	drink obvious
KR	location side	good prepare	service forgot	drive girl	forgot back	drink obvious
WL	location side	good prepare	service forget	drive girl	forget back	drink obvious

Table 3.4: Following the same example of Table 3.3, samples of different preprocessing treatments on a Yelp review.

the possible unigrams it could generate in a moderately principled way. As stronger stemmers, both Paice/Husk and Lovins encounter a more extreme version of the same problem: because these both conflate terms liberally, the result seem to even worsen the normalized model likelihood.

More surprising, however, is the mediocre performance of the WordNet lemmatizer. The fact that both Yelp and IMDb do not see an improvement with use of the lemmatizer is easy to explain away: these corpora contain slang, misspellings, and plenty of proper names, enough to make lemmatization a challenge. However, we see the same result in the case of New York Times articles, a theoretically ideal corpus for text mining. While there are still many named entities, they arise in carefully-edited text with standardized journalistic vocabulary. However, it may be that the context conveyed by morphology is still resulting in loss of useful model information from lemmatization.

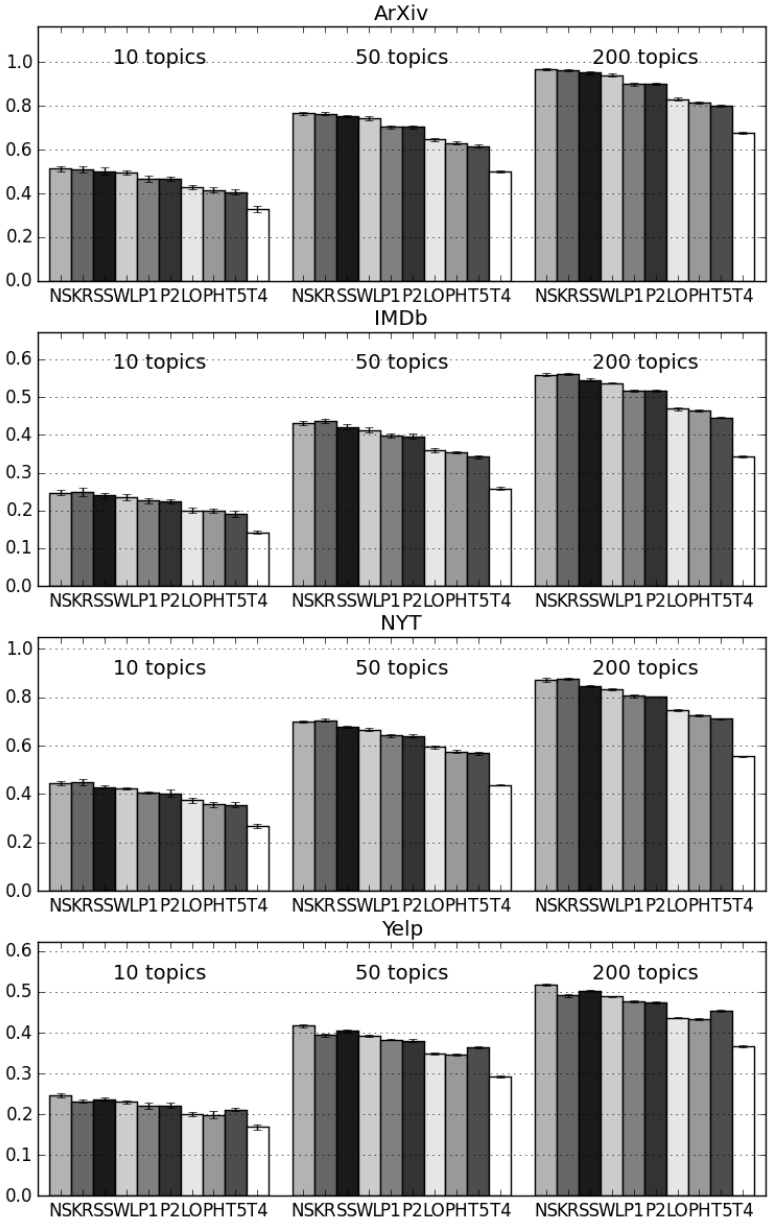


Figure 3.2: While light conflation treatments may help particular corpora, word conflation generally decreases the statistical fit of a topic model proportionally to its strength as measured in normalized log likelihood. Confidence intervals are the $p = 0.99$ range of belonging to the distribution of that treatment's normalized log likelihoods across at least 9 samples each. Higher values of normalized log likelihood represent better model fit.

Other observations are less surprising. Five-truncation produces likelihoods comparable to the stronger Lovins and Paice/Husk stemmers, and significantly better than either for the 50-topic Yelp model. This may relate to the irregularities of review text: words elongated for emphasis (e.g., "helloooo") and other oddities of online informal English are hard for rule-based suffix stemmers to handle but still benefit from naïve forms of conflation. The Porter and Porter2 stemmers are not significantly different in any case, which serves as comforting validation that those not using the new generation of the Porter stemmer are not losing much.

Topic Coherence

Log likelihood measures do not necessarily tell us about the actual apparent coherence of the model in terms of conceptual similarity of words in a topic. In Figure 3.3, we display the negative average coherence scores from Equation 2.6 for each treatment. The hypothesis we test is that using a conflation treatment should map morphologically different words with a shared concept to the same word, automatically constraining the topic model to ensure closely-related words are proportionally present in the same topics.

Our results do not conform to this intuition. The majority of treatments are statistically indistinguishable from the untreated control with respect to coherence. The relative effects of these treatments on coherence are magnified as the number of topics increases; while no ArXiv treatment differs significantly in coherence at 10 topics, at 200, the four strongest treatments (Lovins, Paice-Husk, five-truncation and four-truncation) are significantly worse. Four-truncation suffers a similar effect on IMDb at 50 and 200 topics. In contrast, four-truncation actually improves in coherence compared to other treatments on Yelp as the

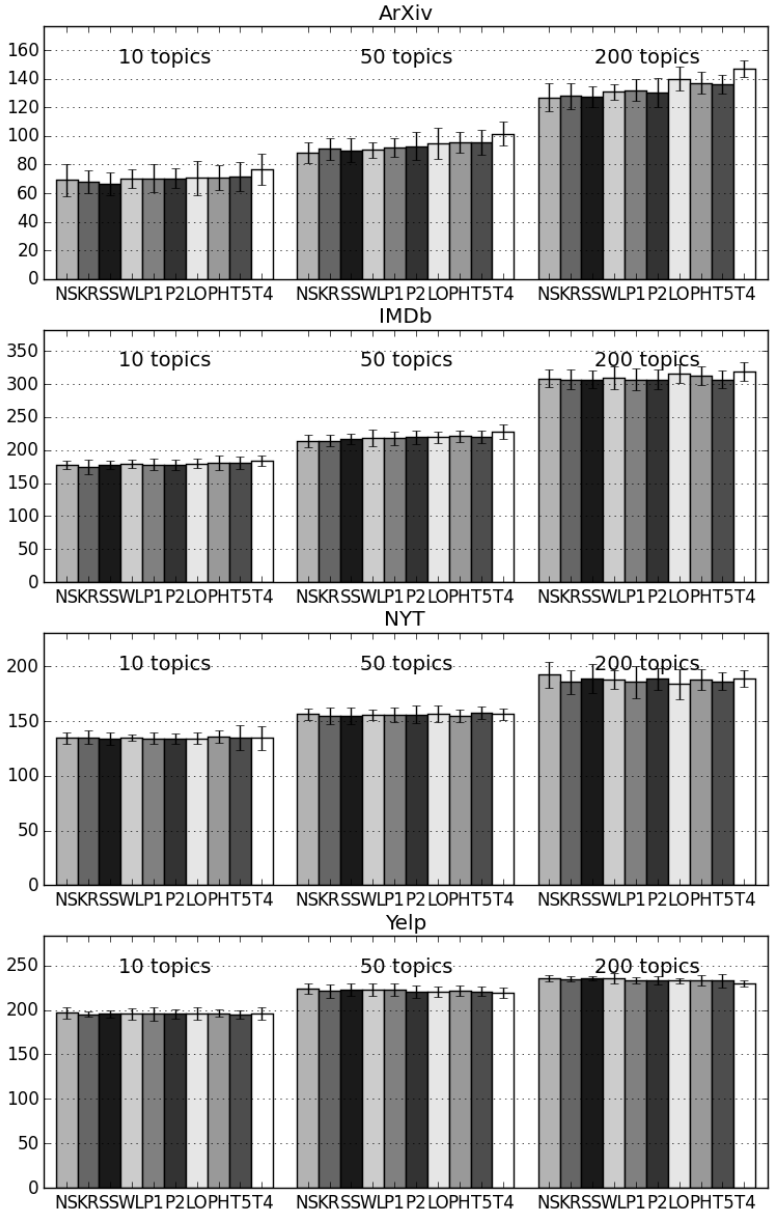


Figure 3.3: Conflation treatments introduce no significant difference in almost all cases in the resulting average negative topic coherence of each model according to token assignments. Smaller values indicating better coherence, and error bars represent the $p = 0.99$ range of possible mean values.

number of topics increases, reaching a significant level at 200 topics. Given the lack of substantial statistical difference across a variety of treatments, it seems

safe to conclude that the use of stemmers is not substantially improving the encoding of word similarities in these topics. The topic model itself on the untreated corpus is perhaps already doing as good a job ensuring that words in the same conflation class have statistically similar topic distributions.

Unstemmed topics sometimes contain words from the same conflation class (e.g. "restaurant" versus "restaurants"). While these might give a slight advantage in coherence measures, this case implies that the stemmers are not necessary for topic models like LDA to place words with shared roots in the same topics.

Clustering Consistency

Another hypothesized effect of stemming is that it will produce more consistent results by reducing the sensitivity of related words to random initialization. We can use variation of information (VOI) to understand how these models differ from each other relative to how much they vary between random initializations. We summarize these comparative stability results in Figure 3.4.

Within statistical error bounds, intra-treatment VOI is always less than or equal to the variation across treatments, and VOI increases as the number of topics increases. On ArXiv, the light treatments — the Krovetz stemmer, S-stemmer, and WordNet lemmatizer — behave indistinguishably from the untreated corpus. The intra-treatment VOI trend shows that stronger treatments generally result in less consistent models. This contradicts the intuition that stemming will help place words with similar meaning into the same topic. While stemming constrains all conflated word types to share one probability in each topic, it does not ensure that those probability distributions will favor few topics.

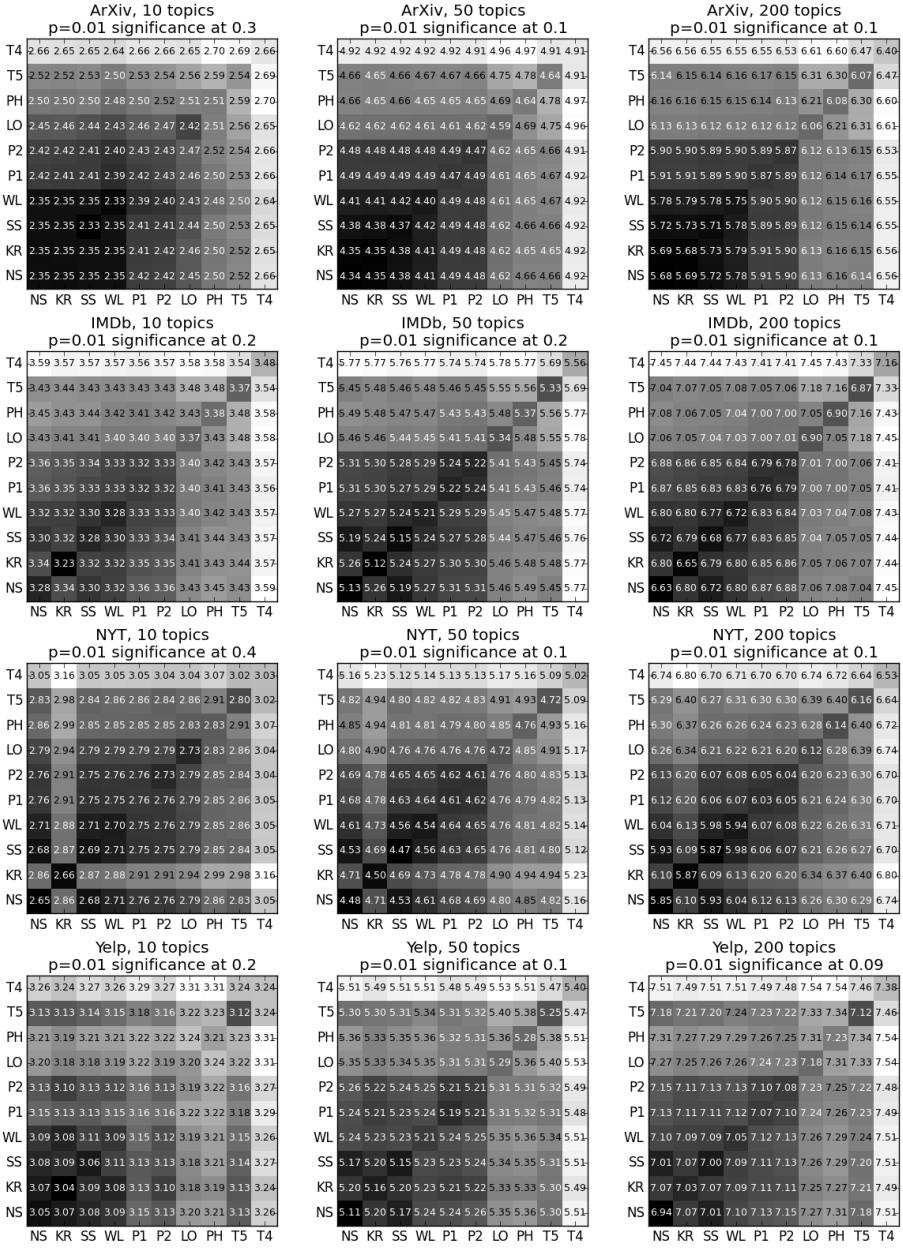


Figure 3.4: The variation of information between different treatments of corpora indicates that while light stemming may improve the comparative similarity of topic models, heavier stemmers produce less stable topic assignments. The minimum for statistical significance is computed as the maximum $p = 0.01$ value for any topic model as compared with itself (i.e. the 95% confidence interval on the diagonal).

There are two striking exceptions to this trend. The first is the Krovetz stemmer. The intra-treatment VOI of Krovetz topic models stays closer to that of the untreated corpus than the S-stemmer or the WordNet lemmatizer. However, the higher inter-treatment VOI between Krovetz and the unstemmed corpus suggests that the Krovetz stemmer produces small but significant changes in the optima of the topic model. For IMDb, NYT, and Yelp at 200 topics, and NYT again at 50 topics, the VOI between the untreated corpus and Krovetz-stemmed corpus is significantly greater than the VOI of the untreated corpus with itself. In contrast, the variation of information between untreated and S-stemmed corpora is only negligibly higher than the intra-treatment VOI of the S-stemmer. This is interesting given the reputation of Krovetz as a weak stemmer.

The second exception is the five-truncation stemmer. Though a very strong stemmer, its VOI is indistinguishable from the heavier Lovins and Paice-Husk stemmers on most corpora. When applied to Yelp with 200 topics, however, this baseline is more stable than either of these conventional rule-based stemmers, in both intra-treatment and inter-treatment VOI with the unstemmed corpus. This effect can be seen to a less significant extent in models with fewer topics over Yelp. This does not imply that five-truncation is a competitive stemmer, but rather illustrates that by this measure, strong stemmers perform worse than a naïve baseline on a corpus with short words and irregular text.

Influential Words

To identify word types that positively or negatively affect the quality of the model after stemming, we use our idf-probability and entropy metrics for each word type. The idf-probability metric strongly indicates that while conflation improves

probability of words on average, the improvement applied primarily to conflated words. Untreated words that are solitary members of their respective conflation conflation classes under a given morphological treatment (e.g. "marquess") often become less probable on average after stemming. Their inferred hyperparameters are larger and thus encourage less sparsity in stemmed topic models. As a result, the probability of rarer words in their own conflation classes decreases as that probability is more distributed across topics. This effect also increases the entropy of stemmed words from these single-word-type conflation classes.

We confirm several hypotheses from earlier in the paper using these methods. The conflation classes with the greatest weighted probability improvement for the truncation stemmers in ArXiv include huge conflation classes of words with the same prefix but wildly different roots. In effect, these classes have forced sparsity over topic distributions where it should not necessarily have been, degrading coherence. In the 50-topic NYT models, this effect is exemplified by the Porter stemmer. Applying this stemmer improves the likelihood of common words, like "street" ($TPscore = 5370$) and "mr" ($TPscore = 13945$), an outcome aligned with the rule-based stemmer's aim to cope well with common words. But for rarer words like "purgative" ($TPscore = -17.5$) and "pranks" ($TPscore = -15.4$), no such improvement is seen. These common words do not have extreme entropy values, which supports our hypothesis that while the likelihood of common words improves with Porter stemming, those words were already in the same topic and did not affect model coherence. While we cannot use the same $TPscore$ entropy measurement on the lemmatizer, which may modify affixes differently depending on context, we see the same effect: the most-improved words in terms of likelihood are common words in the vocabulary, and the words made less likely in the stemmed model are rare words and names.

Interesting results also arise from the five-truncation stemmer. Unlike prescriptive rule-based stemmers, the truncation stemmer does not produce more errors when typos arise; in fact, it can accommodate typos at the ends of words in a way that other stemmers cannot. While, once again, we observe that the word probabilities of truncated words are much improved for common words and slightly reduced for rare words, we discover that the best entropy improvements from untreated to stemmed words are elongated words and exclamations such as "eeeee" ($\Delta H_w(k) = -2.56$) and "haaaa" ($\Delta H_w(k) = -3.25$). At the opposite score extreme, several classes of words with many misspellings have increased entropy after stemming. This is potentially misleading: topic models are very good at distinguishing dialects, and systematic misspellings are likely to create differently-spelled but semantically similar topics in a many-topic model. Over one hundred words conflate to "defin" with five-truncation, including upwards of sixty misspellings of "definitely," which removes distinction between good and bad spellers that might be correlated with other features.

3.3.2 Stopwords

We evaluate the results of removing stopwords for topic modeling on two different corpora: a corpus of United States State of the Union (SOTU) addresses from 1790 to 2009 split into paragraphs, and a 1% sample of the New York Times Annotated corpus [137], spanning articles from 1987 to 2007 and split into 500-word segments to handle overly-long articles. For experiments relying on held-out data for the NYT corpus, we sampled approximately 5% of the articles to be used as a testing corpus. We treat the full article set as a reference corpus for word co-occurrence. The details of the size of each corpus are in Table 3.5. We use a

standard stoplist from MALLET for our experiments [100]. We compare models with no stopwords removed (control), stopwords removed before training (pre), and stopwords removed after training (post) all with the same effective corpus. Metrics are averaged over 10 models per treatment.

We train several types of topic models on New York Times (NYT) data. Our standard treatment defines a document as one full article, but we additionally train models on a segmented version of the corpus (NYT-S) where each article is broken into 100-word segments. In addition, we include models with unoptimized hyperparameters (NYT-U), set as $\sum_k \alpha_k = 5$ and $\beta = 0.01$.

Mutual Information

Table 3.6 presents the $MI(d, k)$ for models with all tokens included (All), only the stopwords, removed stopwords before and after training and compared to fully randomized tokens. We can see that the stopwords, as expected, contain less information than do non-stopwords. Interestingly, there is only a small difference between removing stopwords before or after the training of the model. For the smaller corpus, removing the stop-words before training seems reasonable and actually gives a better model. However, for the larger corpus, the distinction is

Corpus	Documents	Tokens
NYT	18820	10.33M
NYT-S	18820	6.50M
SOTU	19254	1.264M
SOTU-S	19254	681K

Table 3.5: Details of the New York Times (NYT) and State of the Union (SOTU) corpora used for topic modeling. We experiment with a fixed English stoplist of 524 words to remove stopwords (-S). We use the full SOTU corpus for training.

Corpus	All	Stopwords	Post	Pre	Random
NYT-1 10	1.146	0.960	1.237	1.293	0.008
NYT-1 50	1.687	1.208	1.936	1.989	0.045
NYT-1 200	2.130	1.426	2.486	2.502	0.180
NYT-1C 10	1.29	1.117	1.348	1.438	0.059
NYT-1C 50	2.102	1.672	2.225	**2.342**	0.333
NYT-1C 200	2.827	2.233	2.929	**3.068**	1.026
NYT-5 10	1.233	1.093	1.297	1.372	0.008
NYT-5 50	1.895	1.518	2.085	2.149	0.046
NYT-5 200	2.478	1.962	2.724	2.760	0.184
NYT-1U 10	1.025	0.899	1.061	1.064	0.008
NYT-1U 50	1.730	1.353	1.901	1.927	0.044
NYT-1U 200	2.483	1.928	2.756	2.793	0.182
SOTU 10	1.128	1.070	1.120	1.192	0.065
SOTU 50	1.625	1.378	1.666	**1.844**	0.316
SOTU 200	2.088	1.673	2.085	**2.347**	0.863

Table 3.6: Mutual information between documents and topics for different numbers of topics and both 1% and 5% NYT samples as well as the State of the Union corpus, evaluate over single trials. We test NYT with and without hyperparameter optimization (U), as well as split into document chunks of 100 tokens (C). Differences larger than 0.1 in MI between pre- and post training removal are marked.

no longer meaningful. In considering the results without hyperparameter optimization, leaving stopwords actually seem to improve the model slightly. This is consistent with the observation mentioned previously that hyperparameter optimization helps to concentrate background words such as stopwords in a single topic. Taken together, the pre-removal of stopwords does not seem to have a strong effect on inference of the final model.

In Figure 3.5, we examine topic-document mutual information as the number of stopwords removed increases, comparing removal before and after training the model. By removing stopwords before training, we obtain a slightly higher $MI(d, k)$ than removing stopwords after training, in support of hypothesis 1 in Section 3.1.2. However, this difference is relatively small compared to including more stopwords or changing the number of topics.

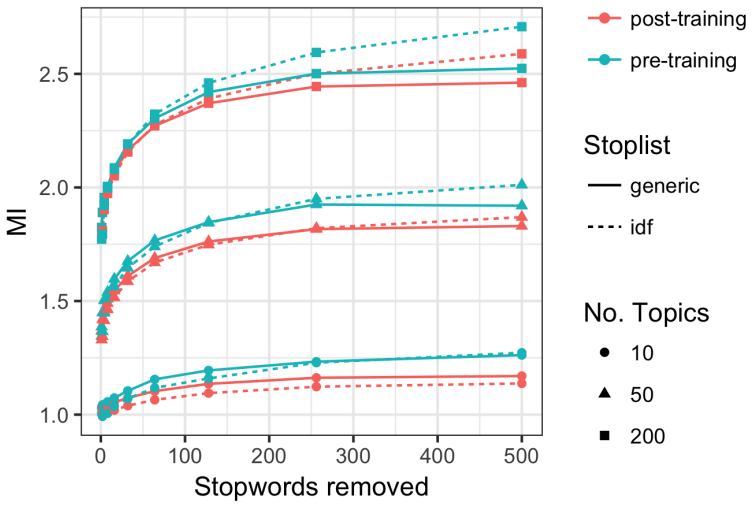


Figure 3.5: Comparison of mutual information $MI(d, k)$, with stopwords removed before and after training from the NYT corpus. Words were removed one by one for models in order of frequency, with one model trained per stopword removed. Removing stopwords before training leads to a slightly higher MI overall.

If we focus on terms besides stopwords in Figure 3.6, we can see that the effect of removing stopwords is relatively small. The most common stopwords produce some change in inference, but extending the stoplist has diminishing returns. This finding supports hypothesis 2 in Section 3.1.2.

Log Likelihood

In order to better evaluate the effect of stopword removal on improving model training, we compare the inferred log-likelihood of models trained on the small New York Times sample on a different sample from the same corpus, containing 5% of the corpus. We also consider the inferred likelihood of the training data.

As we can see in Figure 3.7, the results are less dramatic than one might expect. On the New York Times held out dataset, the effect of removing stopwords

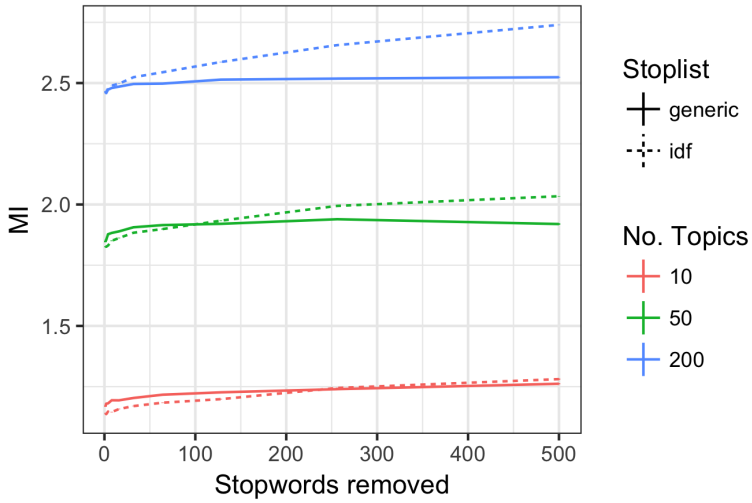


Figure 3.6: Mutual information of the non-stopwords in the NYT corpus as the number of stopwords removed before inference increases. The effect on non-stopword tokens is small.

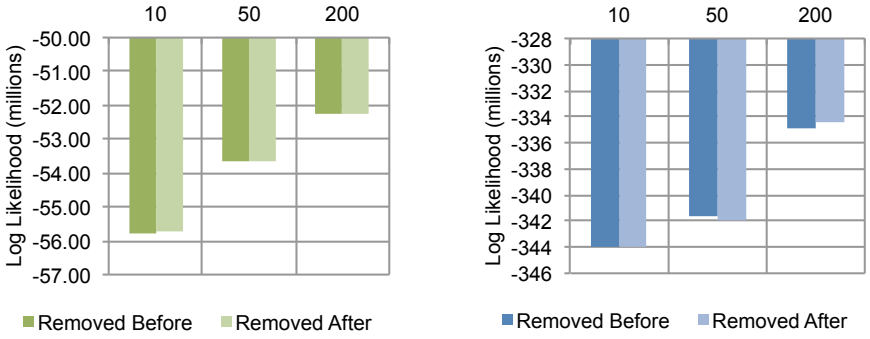


Figure 3.7: Log likelihood measures for training (left) and held out data (right) from New York Times models where stopwords were removed before vs. after training. Models with stopwords pre-removed are negligibly better on training data and not consistently better on held-out test data.

after the model is trained is nearly indistinguishable from pre-removing them. This supports hypothesis 2 in Section 3.1.2, that stopwords are not actually significantly affecting the model inference process for other terms.

The results demonstrate that, while including stopwords in the top keys

pre	1	num art museum work show artists works artist paintings exhibition gallery painting arts american collection
	2	num beloved paid family notice wife deaths husband late loving memorial funeral devoted service services
	3	life love world story sense young man makes good style full real beautiful dark turns
post	1	num art museum show artists work works exhibition gallery artist paintings arts painting american collection
	2	family president board passing love friend paid member notice jewish beloved chairman miss condolences deaths
	3	book life story man books young love written world characters character work writing james author

Table 3.7: Example topics from 50-topic New York Times models with stopwords removed before and after training. Post-removal topics look similar but lack some more common terms found with pre-removal.

certainly reduces apparent model coherence, the effect on the quality of the keys chosen with stopwords removed after training is competitive with the one with the model trained without stopwords. We see across word selection strategies that especially for a small topic count of 10, coherence of models where stopwords were removed after training sometimes outperforms models with pre-removal. Both significantly outperform topic summaries without stopword removal (control). This is in line with hypothesis 2 in Section 3.1.2, that much of the perceived coherence contribution of stopword removal is actually aesthetic, and does not effect the relationship of other terms with topics.

3.3.3 Coherence

We report the average NPMI scores for the New York Times and State of the Union data in Table 3.8. Though removing stopwords from the top keys improves model coherence over the control, the choice of whether the stopwords are removed before or after inference seems to have very little effect. This effect

Topics	Treatment	control	pre	post
10	NYT	0.0280	0.0874	0.0931
	NYT-S	0.0282	0.0850	0.0851
	NYT-U	0.0311	0.0863	0.0878
	SOTU	0.0248	0.0406	0.0402
50	NYT	0.0595	0.1271	0.1209
	NYT-S	0.0531	0.1257	0.1195
	NYT-U	0.0554	0.1233	0.1208
	SOTU	0.0438	0.0655	0.0612
200	NYT	0.0951	0.1352	0.1317
	NYT-S	0.0718	0.1317	0.1239
	NYT-U	0.1021	0.1352	0.1338
	SOTU	0.0542	0.0681	0.0637

Table 3.8: The average NPMI scores for New York Times and State of the Union data. Surprisingly, with 10 topics, post-removal of stopwords often produces better coherence.

is particularly pronounced with a small number of topics: with $K = 10$, the coherence of models in which stopwords were removed after training sometimes slightly outperforms models with pre-removal. This finding supports hypothesis 2 over hypothesis 1 in Section 3.1.2: though removal of stopwords before training improves coherence, *when* they are removed has no meaningful impact.

Classification with Key Terms

	control	pre	post
NYT	$47.1 \pm 0.3\%$	$69.4 \pm 0.2\%$	$69.9 \pm 0.2\%$
NYT-S	$47.1 \pm 0.2\%$	$54.0 \pm 0.2\%$	$53.3 \pm 0.1\%$
NYT-U	$62.6 \pm 0.3\%$	$69.8 \pm 0.2\%$	$69.8 \pm 0.2\%$
SOTU	$43.8 \pm 0.3\%$	$48.7 \pm 0.2\%$	$48.8 \pm 0.2\%$

Table 3.9: Classification results using top terms of 50-topic models on NYT and SOTU data. Removing stopwords is often equally effective before and after training.

We use the 15 most probable words from each 50-topic model on New York Times sample data to train a logistic regression classifier to recognize the most

prominent topic for each document. We use 10-fold cross validation to compute accuracy, which we report in Table 3.9. Unsurprisingly, removing stopwords at some stage improves the classification accuracy of key terms over not removing any stopwords at all. However, we note that removing terms before training is significantly better only for one of the four treatments (NYT-S) and is actually significantly worse than removing after for the standard NYT setting. This again supports hypothesis 2 in Section 3.1.2: removing the stopwords before training does not alter the distinctiveness of topics based on high-probability terms.

Examples of topics in Table 3.7 provide some depth to understanding these results. Topic 1 is nearly identical across the two treatments, while topic 3 uses terms clearly from reviews when stopwords are removed before that seem to be lost when stopwords are removed afterwards. Anecdotally, common content words appear not to be modeled as well when stopwords are present.

3.4 Discussion

The results above provide the following dominant conclusions:

1. Topic model inference often places words sharing morphological roots in the same topics, making morphological conflation such as stemming redundant and potentially damaging to the resulting model. Though topic summaries may look redundant, it is better to avoid any kind of conflation treatment unless documents appear to be too short or the vocabulary too dispersed to work otherwise. In those settings, a lighter conflation treatment, such as a lemmatizer or S-stemmer, is probably sufficient.

2. Aside from extremely frequent stopwords, removal of stopwords does little to impact the inference of topics on non-stopwords. To find stopwords to remove in advance, it should be sufficient to use a frequency-based approach, such as finding words make up more than 1% of the corpus or that are in at least 50% of documents in a collection.
3. For both stemming and stopword removal, if the resulting topics are difficult to interpret due to less pre-processing, the resulting vocabulary may be processed after inference instead. Post-processed topics should look much like topics learned after pre-processing, but without the risk of inadvertently discarding useful information prior to inference of a model.

Our results substantially simplify the work of practitioners. Many burdensome tasks turn out to have little effect, such as stemming and corpus-specific stoplists. As a result, the methodology of post-processing a corpus instead of pre-processing can allow practitioners the option to test out pre-processing options on one trained model to decide on a treatment best suited for their application. These evaluations also may provide a roadmap for generalization to other languages with richer morphological and syntactic structure. The results suggest that when possible, it is good for practitioners to pre-process more lightly to avoid discarding useful word information by mistake.

CHAPTER 4

UNDERSTANDING TEXT DUPLICATION

Latent semantic models such as LDA look for patterns of repetition. But when text is repeated exactly, statistical methods may be diverted from more meaningful semantic groups: verbatim repetition looks, to the algorithm, more topical than actual topics. If not accounted for, repeated text can change measures of fitness to overvalue fit on repeated texts. Repetition may also "leak" held out data that is duplicated in the training data, artificially inflating the apparent model fit. At best, duplication may cause us to overestimate the expressiveness and reliability of models. At worst, models skewed by text duplication may draw invalid or misleading conclusions from their data.

Text replication is a persistent and difficult problem in natural language corpora. In social media, partial duplication is ubiquitous due to threading, quotation, and cascades of sharing. This phenomenon is not new. Of the 20k posts in the 20 Newsgroups corpus [81], 1,151 (5.76%)are exact duplicates shared in multiple newsgroups, and 25% of the tokens in the remaining non-duplicate posts are made up of quotes from other newsgroup messages.[1]

However, detecting these duplicates can be quite challenging. In literary corpora, different versions of the same document may also conflict: text files for Hamlet may differ slightly due to publisher information, line numbers, editorial changes between Shakespeare's folios, and footnotes. Removing exactly identical duplicates of texts is possible through lexicographic matching, but for lexical near-duplicates and partial textual overlap, we may need more careful heuristics

[1]Computed using `scikit-learn`'s 20 Newsgroups API: `http://scikit-learn.org/stable/datasets/twenty_newsgroups.html`

to detect duplicates. Design of these heuristics forces researchers to make broad judgments about what text to remove and what to keep. Evaluating what level of duplication is "safe" can therefore not only reduce the risk of false conclusions but also save great amounts of work spent identifying and removing duplication.

In this work,[2] we investigate the effect of text duplication on LDA by generating semi-synthetic corpora that amplify the magnitude of text duplication in a variety of corpora. We examine both how models shift to over-represent repeated text and how that shift affects the model representation of documents without repetition. To account for the variety of types of duplication, we simulate not only exact duplication of whole documents, but also repetition of a text segment across many documents. Finding that LDA often sequesters duplicated text into single topics, we make recommendations as to how to address duplication depending on its content with respect to the content of the corpus. [3]

4.1 Previous Work

Text duplication and reuse is a well-established problem in textual corpora. The web is filled with pages of near-duplicate content [23, 98]. Journalistic reuse is also common practice, in particular with the dissemination of information from news agencies to newspapers [34, 148], and plagiarism is prevalent in student submissions [33]. However, past work has primarily focused on the *identification* of reuse instead of the *effects* that duplication has on semantic models. An exception to this is work by Cohen et al. [35], which proposes a modification of

[2]The work of this chapter reproduces text and results from an existing co-authored published paper on text duplication [146] that also includes LSA models.

[3]Code for our experiments can be found at `https://github.com/heraldicsandfox/semantic-text-duplication`.

LDA to combat the problem of duplicate text. Our work differs in that it attempts to thoroughly quantify in detail the impact of text duplication on LDA model instead of modifying the model to accommodate duplication.

The detection of text reuse relies on the ability to measure similarity between documents or passages. In general, these techniques measure the similarity of textual content, though other similarity metrics for reuse identification have been proposed [12]. These measures can fall into two general groups: global and local. Global techniques measure the similarities of entire texts. These techniques are especially used for near-duplicate detection. A common approach of this form is fingerprinting [95, 124]. This method involves transforming a document into a smaller representation, or fingerprint, to afford quick comparison of documents, e.g. by using frequencies of a selected subset of n-grams much smaller than the corpus n-gram vocabulary. These methods connect to locality-sensitive hashing in that they compactly represent documents while still representing document similarity in the fingerprint. Local techniques measure similarity at a finer granularity (e.g. paragraphs or sentences). In this setting, text reuse from one source may be mixed with original text or reuse from other sources. Local techniques often involve two steps: one aligning sequences within documents with some method [84, 148] and one scoring similarity of aligned sequences, e.g., based on cosine similarity of the bag-of-words vector. Both global and local techniques require choices of hyperparameters such as similarity threshold and n-gram size that affect what the technique considers duplicate text. Our work focuses on understanding what types of document deduplication are important to LDA models, such that practitioners can make better-informed choices about which deduplication methods to apply to their text collections.

4.2 Theorized Impact

The fundamental problem with r in a distributional semantic model is the over-representation of specific word co-occurrences to a model. To understand this, we consider the matrix factorization representation of these models as inspired by Poisson matrix factorization as described in Section 2.2.3. We consider a corpus with D documents and vocabulary size V over which we want to learn a K-dimensional representation of each document and vocabulary term. We can build an $D \times V$ matrix, Y to represent our corpus, where y_{di} is a function of the frequency of term i in document d. LDA performs a non-negative matrix factorization on a smoothed stochastic version of Y, producing row-stochastic matrices θ of dimension $D \times K$ and ϕ of dimension $K \times V$. Optimally, the difference between this stochastic version of Y and the reconstruction $\hat{Y} = \theta\phi$ is minimized according to their distributional KL-divergence.

Duplicated text should produce more rows of Y with the particular signature of word frequencies of the duplicated text. This implies that a low-rank matrix factorization should devote more representative power to modeling this textual signature in order to minimize model loss. When we artificially introduce duplication, we therefore expect to observe two principal effects:

- As text is repeated more, to optimize model fit on the data, one or more topics will converge to model the repeated text.
- Consequently, text that is not exactly or near-exactly repeated (or *singular* text) will be fit less effectively by the model.

These anticipated effects are based solely on the incentive of the model to overfit repeated text: topics and dimensions modeling solely the repeated text will

leave less representational power for the remaining text, and the combination of repeated and singular text should yield less coherent topics.

4.3 Evaluation Methods

We quantitatively examine several aspects of models with varying forms and degrees of duplication to determine the magnitude of the change produced by repeated text. It is important to note that our goal is simply to measure the difference between models, and not to make normative statements about the *quality* of topics. Indeed, many measures of topic quality such as word intrusion [27] and word co-occurrence [20, 111, 83] may improve as a result of degenerate, single-document topics: most documents are internally coherent, so a single document's word distribution may appear to be a sensible topic.

Model Fit The first aspect is model fit. As stated in Section 4.2, as a segment of text is repeated more, we anticipate that the fit over documents containing repeated text will improve, while the fit over documents not containing repeated text will worsen. In this work, we evaluate this through perplexity via log likelihood estimates from MALLET's built-in left-to-right estimation [166].

Concentration Secondly, we examine component (e.g. topic/dimension) concentration. Repetition of a document amplifies the co-occurrence between the terms contained in the document. As this signal grows stronger, we expect models to begin "memorizing" these words. We anticipate that affected models will develop a simpler latent representation for the repeated document, one concen-

trated over a small number of components. For example, if a model devotes topic k to a repeated document, then instances of that document should have a high proportion of topic k in their vector representation in θ. While loss considered the combined likelihood of the model parameters θ and ϕ, concentration focuses specifically on the document-component or document-topic patterns θ.

To measure concentration in LDA models, we examine the *entropy* of document vectors. Information entropy for a probability distribution over discrete outcomes represents the expected minimum length of string required to represent which outcome occurred within the given probability distribution:

$$E_d = \sum_k \theta_{dk} \log \theta_{dk}$$

where θ_{dk} is the probability of a token generated in document d having topic k. Under the hypothesized effects described in Section 4.2, as text is repeated more, the topical entropy of duplicated text should decrease, as all of the topical mass should be concentrated in the topics converging to the duplicate text's unigram language model. Further, if the model fit has worsened for nonduplicate text, the entropy of these remaining documents should increase as text repetition increases, as topics will less adequately fit to the behavior of the singular documents.

Expressivity The final aspect we wish to capture is the *expressivity* of topics. If one topic converges to the unigram language model of repeated documents, the resulting model has effectively lost one topic worth of expressive power by focusing on overly-specific themes. Someone looking to learn generalized semantic corpus patterns from a topic model will therefore have one fewer topic of interest available per topic fit to such repetition. With sufficient repetition, the frequency of terms in the repeated text may also overwhelm the most probable

terms in topics, which would reduce a practitioner's ability to interpret topics.

Using LDA models, we may measure expressivity via topic summaries, obtained as the top M most probable terms of a topic where M is a fixed parameter. We select the same number of terms M from the most probable tokens in a unigram language model of the repeated text. We report what proportion of the tokens obtained from the topic summaries are contained in the set of top terms of the unigram language model of the duplicated text.

4.4 Experimental Setup

The primary experimental work is in the creation of semi-synthetic datasets to simulate duplication behaviors. This requires creating de-duplicated base text collections and algorithms for introducing duplication.

Data We use two corpora: a sample of articles from the New York Times Annotated Corpus [137] and a collection of Reuters Spanish-language newswires (REUSL) [61]. We choose licensed news corpora because they provide well-curated text with repeated subjects but few exact document-level duplicates, though quotes and templated text may still cause text duplication. We use these collections to construct semi-synthetic corpora representing general duplication behaviors we see across a variety of settings. Text from these collections is lower-cased and filtered to only include tokens of three or more characters, with contractions or hyphenations as single tokens. New York Times article lengths average 494.5 words, while Reuters newswires average 201.5 words.

To ensure our experiments are the only cause of exact duplication of text in our corpora, we use strict methods of text deduplication. When two or more documents have more than 70% unigram overlap, we remove all but the longest document. In addition, we delete 7-grams that appear in more than 10 documents based upon existing thresholds for plagiarism detection [32]. To account for stopwords, we remove all terms appearing in more than 80% of documents. Finally, we remove documents with fewer than 7 tokens after processing. We perform this process on a random sample of 30,000 documents from each corpus to ensure we may obtain a sample of 25,000 curated documents for each of our two corpora. We also produce 10% samples of these corpora, containing 2,500 documents each, to test whether corpus size affects our results.

Text Duplication Treatments We use our deduplicated news corpora to construct datasets with artificial text duplication. We examine two different duplication scenarios: exact document duplication and template string duplication.

In *exact document duplication*, we randomly sample $p\%$ of the documents in the dataset and include c copies of each sampled document in our final corpus along with one copy of each remaining document, which we refer to as *singular* documents. To test extreme rates of repetition, as might be seen in cascades in a social network, we also perform *single document* tests for large c with only one repeated document. From these synthetically duplicative corpora, we can determine whether effects are triggered by the sheer volume of duplicated text or if they are influenced by the diversity of the copied documents.

In *template string duplication*, rather than duplicating the sampled $p\%$ of documents, we prepend a fixed string to each document in the $p\%$ sample, producing

what we refer to as *templated* documents or texts. As repeated text may be lexically similar or different from the non-repeated text of the corpus, we consider two different types of prepended string. The first is a randomly-sampled document from the deduplicated corpus that is not included in the training set (*Sampled Template*), to simulate repeated text that is lexically similar to the document content. The second is the first 100 words of the classic pseudo-Latin Lorem Ipsum filler text (*Lorem Template*), to simulate repeated text with little lexical overlap with the documents. Because we are investigating a bag-of-words model, we do not worry about grammatical errors in the nearly-duplicated text, so the segmentation of this repeated prefix should not be a concern.

Model Inference. LDA models are inferred using MALLET [100] with the number of topics increasing by factors of two, ranging from $K = 5$ to 320. Stepping away from the best practice of using an asymmetric prior α [165], this work uses fixed symmetric hyperparameters $\alpha = 50/K$ and $\beta = 0.01$ to ensure entropy measures are not skewed by topic size. To ensure topic expressivity can be measured from short topic summaries without post-processing, we remove stopwords using canonical stoplists of English and Spanish from MALLET.

4.5 Results

Because of the exponential combination of different experimental settings available, it would be unfeasible to examine all our metrics for all data. Instead, we focus our analysis on several representative experiments.

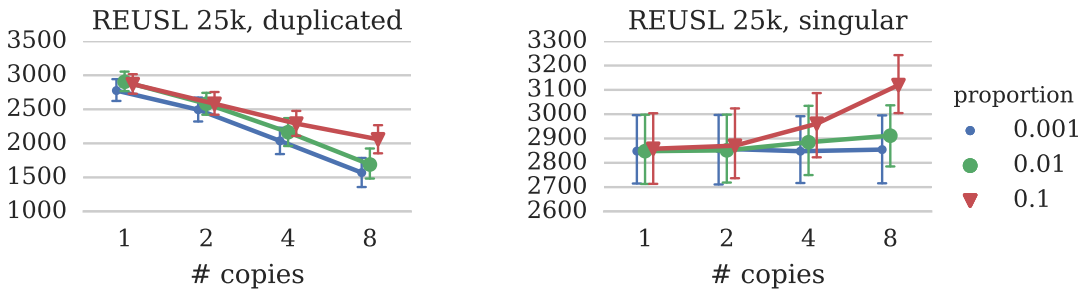


Figure 4.1: Training perplexity with LDA models of the REUSL 25k corpus with 80 topics. Perplexity decreases significantly for the duplicated documents as the rate of repetition increases, but the effect on singular documents is negligible so long as less than a proportion of 0.1 of the corpus repeated.

4.5.1 Loss

We begin with the case of exact document duplication. In Figure 4.1, the perplexity of LDA decreases substantially as documents are duplicated. This reduction is due to better fit to the duplicated documents. As fit improves in duplicated documents, however, we do not see a meaningfully worse fit for singular documents. The sheer volume of duplicated text does not by itself damage model fit, likely because the duplicated text can be easily modeled by a single topic.

Figure 4.2 shows that the fit for held-out test data is not significantly affected by increased repetition. There is a pattern within the data, in which repeating documents 4 times seems to produce better perplexity for singular documents than 2 or 8, significantly so for a small fraction of the corpus. A theory for this is that at the right level of repetition, LDA fits the repeated text cleanly to a single topic, resulting in improved predictive performance on held-out de-duplicated text. However, additional repetition further saturates the remaining topics and adds noise to the meaningful co-occurrence signal.

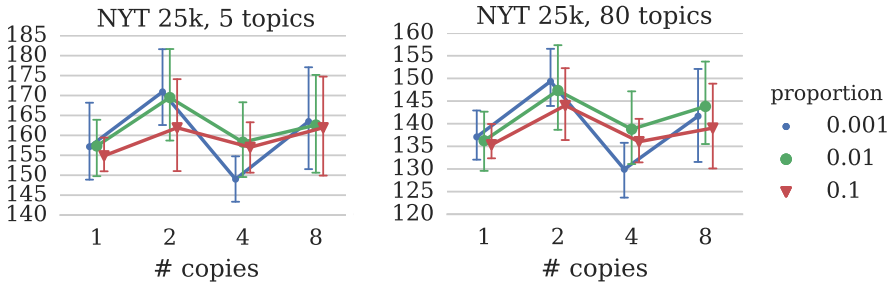


Figure 4.2: Held-out data perplexity (in thousands) for different the NYT 25k corpus with varying numbers of topics K. Increasing the proportion of repetition for exact duplicate documents does not increase test perplexity. With repeated corpus proportion $p = 0.001$, however, repeating documents exactly 4 times (but not 2 or 8 times) significantly improves perplexity, potentially because it induces a new topic to model it. Held-out data was de-duplicated with the training data.

4.5.2 Concentration

We expect the effect of duplication on entropy will be inversely correlated with its effect on model loss. As we increase the proportion of the corpus that is repeated, the model will devote more resources to duplicate text, leaving less modeling power for the remaining text. We therefore expect dispersion to increase with p for duplicate documents and decrease with p for singular documents. In Figure 4.4, the first effect clearly holds for LDA, but the second does not: there is a negligible change in entropy with the number of repetitions of documents.

We can examine the extreme effects of the change in component concentration for singular documents by looking at its behavior in the *single document* treatment. In Figure 4.5, we see that while entropy remains level for repetitions comprising smaller portions of the corpus, eventually the entropy drops for both repeated and singular documents. This may be because most topics describe the repeated document, leaving few to model the remaining singular documents.

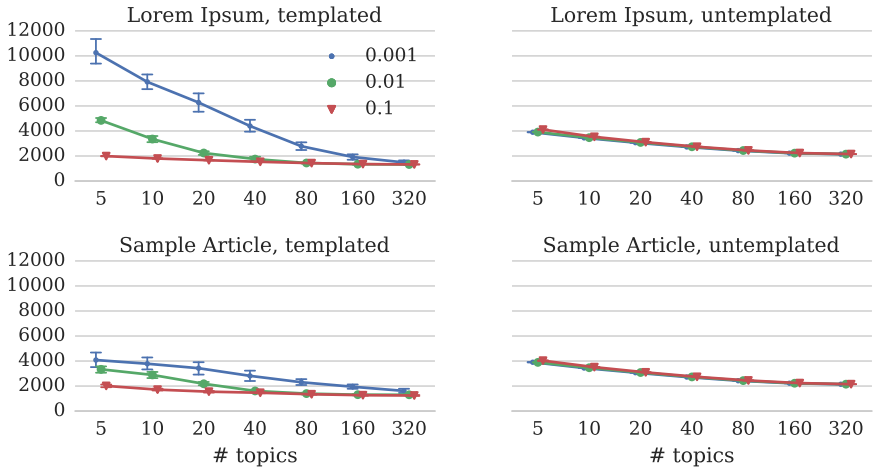


Figure 4.3: LDA training perplexity for REUSL 2.5k with different types of templated text repetition. The effect of duplication is prominent for small numbers of topics but diminishes with more topics to sufficiently model the missing text. With the fraction of the corpus that contains duplicates $p = 0.1$, the perplexity of template documents is below that of untemplated texts.

The effect is more subtle when templated text is repeated within documents. Figure 4.6 shows that with $K = 20$ LDA topics, if we apply the *Sample Template* to a small fraction of documents ($p = 0.001$), it produces a higher entropy than corpora with larger template inclusion proportion p. This is not surprising: though the template text and the original document are similar in style, with high probability they will still have different topics, which the model will have trouble fitting well without more observations. The *Lorem Template* has the reverse effect: the language is sufficiently disjoint from the content of the documents that few topics or even a single topic can model the repeated text fully, leading to low entropy. When the language model of duplicated text is disjoint from that of the text of interest, the template can be modeled by one or a few topics or components without significantly affecting other text.

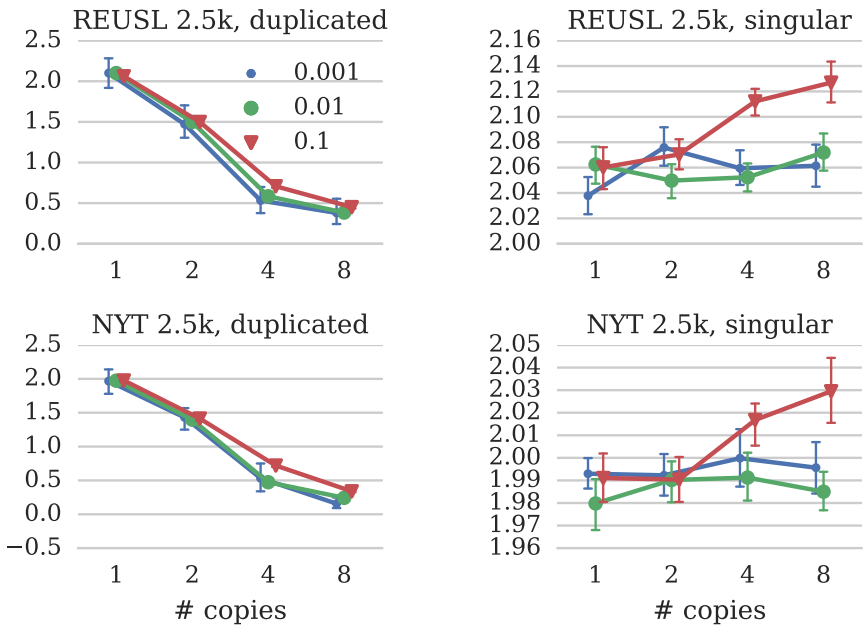


Figure 4.4: Entropy for LDA with 80 topics decreases for duplicated documents as the frequency of those documents increases, has little initial effect on the entropy of the singular documents.

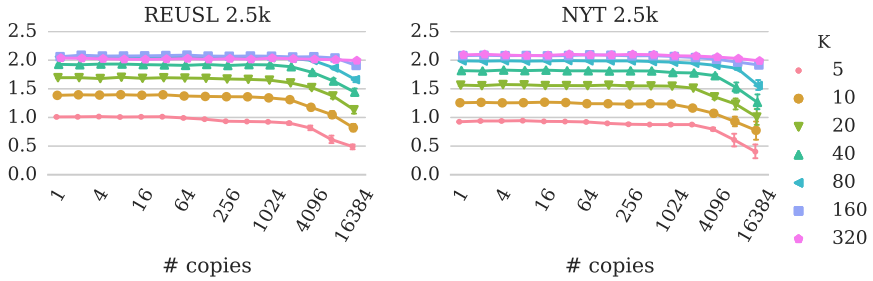


Figure 4.5: When a single document in the short corpora is repeated enough to comprise the majority of the corpus, the LDA entropy decreases over singular documents.

4.5.3 Expressivity

Quantitative analyses of model fit and topic uncertainty are helpful to analyze the effect of different settings on how topics distribute their representational power. However, these quantities do not indicate whether topics from corpora

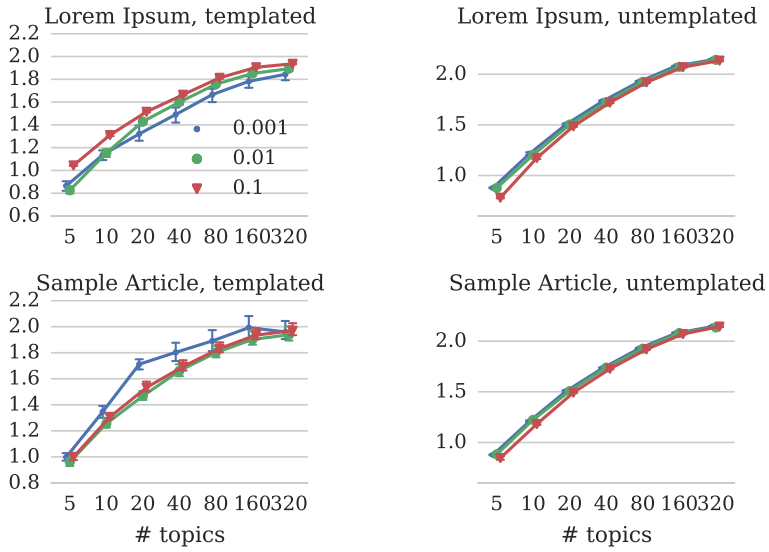


Figure 4.6: LDA entropy for the REUSL 25k corpus with *Sample Template* and *Lorem Template* treatments. With few topics, templated documents have lower entropy than untemplated documents, but with many topics, their entropy is higher. In the middle range of number of topics K for *Lorem Template*, higher proportions of sampled text p produce higher entropy, but for *Sample Template*, lower p produces higher entropy.

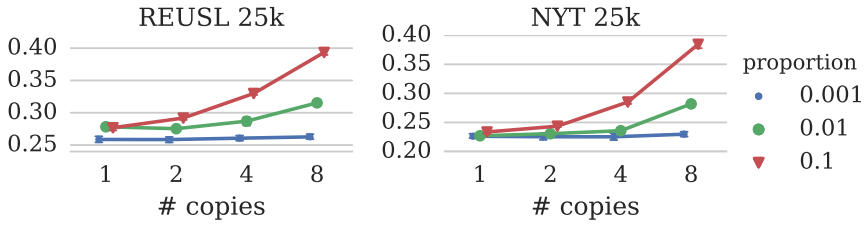


Figure 4.7: With 80-topic LDA models of our larger datasets, we see that increased repetition leads to significant increases in the amount of representation of repeated text in the top keys of topics.

with repeated documents are actually useful. Analysis of expressivity can help fill in some of the gaps in our explanations above as to how repetition affects individual topic representations according to their summaries.

In Figure 4.7, we see that for a moderate number of topics, increased repetition

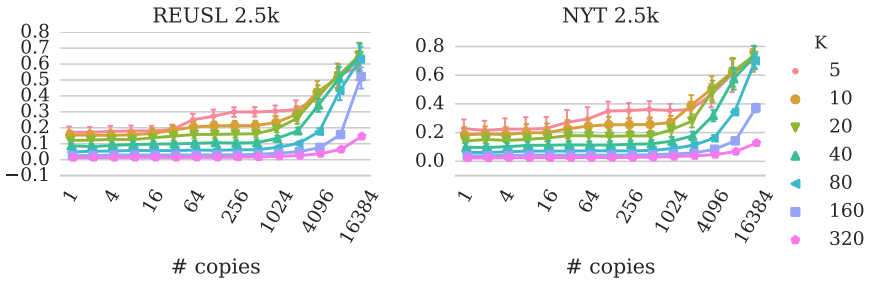


Figure 4.8: Top keys of LDA topics for only a single repeated document remain concentrated in only a few topics in models with $K > 5$, negligibly impacting the top keys of remaining topics.

of documents impacts a substantial portion of the top-ranked words, or most probable terms of topics. The saturation effect has some relation to the number of repeated documents. With a single document repeated, as in Figure 4.8, as the number of topics increases, the ratio of top-ranked words belonging to the unigram language model drops. With few topics, there is a clear "saturation point" where the duplicated , which remains level until half the short corpora are represented by the duplicate document. The pattern overwhelmingly shows that single texts are easily fit by single topics.

In the case of the *Lorem Template* input, where little textual overlap exists between the template and original text, a few topics quickly fill in the repeated text, producing a limited effect on most topics. In Table 4.1, the number of topics containing "lorem" and "ipsum" remains small as the number of topics grows.

Proportion	K = 5	10	20	40	80	160	320
0.001	0	0	0	0	0	0.2	0.8
0.01	0	0	0.2	0.56	1	1	1
0.1	1	1	1	1	1.78	2.3	3

Table 4.1: As the number of total topics K increases, the average number of topics fit to the *Lorem Template* duplicate text remains stable, only rising above 1 when in 10% of the corpus with a model of at least 80 topics.

Regardless of either the topic count or duplicate proportion, topics containing "lorem ipsum" are entirely broken Latin: the top probable terms of an example 320-topic model with $p = 0.1$ are *est justo donec iaculis sit ipsum quam lorem tristique sed amet eget pharetra curabitur fringilla non consequat mattis nec nascetur*, a list of words that can be found entirely in the Lorem Ipsum template text.

4.6 Conclusion

Duplicate text can substantially alter the dimensions learned by distributional semantic models. The effect of duplication depends on several factors: the number of distinct repeated strings, the similarity of repeated strings to the rest of the corpus, and the number of repetitions. We find that while LDA is certainly affected by this repetition, there are straightforward methods to alleviate the effect of duplication without exhaustively removing all duplicated documents. We provide the following specific conclusions:

LDA accommodates low rates of document duplication for many documents. With high rates of repetition, the algorithm is able to sequester repeated text into small numbers of topics if certain conditions hold. To handle this case, the model must have sufficiently many topics available relative to the number of repeated strings, and the language of the repeated text must be sufficiently distinct. If these conditions are met, repeated text should largely be isolated to a small number of topics. These can be identified by their similarity to specific documents, or automatically based on lower than expected inter-document variability within a topic [110] or distance from specific corpus-word or document-word

distributions [5]. We therefore suggest inferring a model first with a slightly larger number of topics K than strictly, then evaluating if there are any signs of repeated texts overwhelming several topics due to low coherence or corpus statistics. If no such indications occur, or if the duplication remains in one or two topics, then there is no need to modify the corpus or reinfer the model, as the duplicate-capturing topic may be ignored. For more widespread repetition, the duplicate-rendering topics can help select text to delete prior to re-inference.

Repeated text templates for LDA are sequestered by the model so long as they do not overlap heavily with topics of interest. In a topic model, it may be easy to identify the templated text based upon it appearing in one topic. However, if there is a concern that there is systematic use of text templates in documents that may be too close to the language model, the n-gram removal approach used in this work and inspired by Citron and Ginsparg [32] is a slow but straightforward way to ensure these strings are detected and deleted.

Part II

Local Privacy for Topic Models

CHAPTER 5

BACKGROUND

Sparse bag-of-words features are well-established as meaningful descriptors of unlabeled natural language text. While these features lack syntactic information necessary for coherent text generation, they are often sufficient for meaningful text classification and retrieval. Furthermore, these counts are the key to generating many popular semantic models, or numerical vector representations of words and documents where similar representations imply similar meaning or content. Models such as LDA [18] and LSA [38] operate on nothing but sparse bag-of-words features for the documents of a dataset. These methods are useful both for feature reduction before classifying and as a direct means of expecting large unlabeled text corpora for interesting patterns.

However, access to large-scale bag-of-words data on interesting corpora can be both difficult to obtain and potentially risky for its authors. It is possible to reconstruct text with bag-of-ngrams data [49]; with auxiliary information, document-level word counts might be sufficient to reconstruct meaningful portions of text corpora. Should this text also contain private information about individuals, whether they are authors or subjects, this reconstruction risks revealing that information through bag-of-words features. With the use of public ngram information, it may even be possible to probabilistically stitch unigrams together well enough to identify attributes of individual documents that are sensitive. This also restricts the possibility of releasing corpora of named documents with detailed word counts after 1923, the last year of publication for texts in the public domain at present. Corpora such as Google Ngrams [106], a release of per-year n-gram counts across millions of novels, provide insight into general

trends but cannot support specific hypothesis on subsets of texts.

The subsequent work presented in this thesis looks at methods for interacting with bag-of-words data in a privacy-preserving way, both for inferring Bayesian Poisson Matrix Factorization models (a family of models which includes LDA) and for direct data release. This is done through the application of *differential privacy* [41], a method of introducing probabilistic uncertainty in computational outputs in order to produce uncertainty about what the original data truly was. In all of this work, we focus on methods which directly perturb the input data counts themselves, instead of as part of the inference algorithm. This chapter introduces relevant background material for understanding differential privacy definitions, existing methods for private inference of related models, and the underlying Bayesian Poisson Matrix Factorization method.

5.1 An Introduction to Differential Privacy

Differential privacy [41] provides a theoretical bound on the privacy given by a randomized computation. The theory behind it builds on the understanding that any kind of data mining operation on nontrivial private data must reveal some amount of private information about the dataset if that data was to be useful. Differentially private mechanisms produce probabilistic outputs such that one may quantify the amount of privacy that is being "spent" by the algorithm. The bound specifically applies to stochastic algorithms, and limits the maximum difference in probability of obtaining any computational output between two "neighboring" datasets, or datasets that differ by at most one observation.

5.1.1 Formal Definitions of Differential Privacy

Multiple definitions differential privacy guarantees exist to quantify the privacy of algorithms. We begin with the classic formal definition of ϵ-differential privacy.

Definition 5.1 (Differential Privacy.)**.** *A mechanism $\mathcal{M}$ is ϵ-differentially private if, for two neighboring datasets D and D' that differ by at most a single observation and a set $\mathcal{S}$ of possible outputs in the range of $\mathcal{M}$,*

$$P[\mathcal{M}(D) \in \mathcal{S}] \leq e^{\epsilon} \cdot P[\mathcal{M}(D') \in \mathcal{S}], \tag{5.1}$$

where ϵ is the privacy budget *of the mechanism.*

As a budget, ϵ is meant to bound the maximum amount of information that is leaked by a computation or query when a differentially private mechanism is applied. A higher value of ϵ shares more private information and therefore gives a weaker privacy guarantee. When it is impossible to obtain a global differential privacy that fits Definition 5.1, one may also appeal to the weaker *(ϵ, δ)-differential privacy*, in which an additive term is also allowed in the upper bound of the likelihood of $P[\mathcal{M}(D) \in \mathcal{S}]$. Though recent work on differential privacy for topic models [119] applies this (ϵ, δ) definition, it is not used here.

Differential privacy provides multiple useful properties [42], which generalize to all mechanisms satisfying the property of Definition 5.1:

- *Post-processing*: the privacy guarantee of a differentially private algorithm cannot be worsened through post-processing of its output without auxiliary access to private information about the original data.
- *Composition*: the composition of different mechanisms is additive: for example, two ϵ-differentially private mechanisms would combine to produce

one 2ϵ-differentially private mechanism.

- *Group privacy*: if a mechanism is ϵ-differentially private for an observation in the dataset, it is at least $k\epsilon$-differentially private for a group of k observations taken together in the dataset.

Most existing work in high-dimensional differential privacy relies on a *central* model of privacy, in which a trusted curator is able to access all of the private data in order to make global decisions about how to add random perturbation. This allows analysis of *sensitivity* δ, or the maximum difference in a query's output between two neighboring datasets. In order to give a privacy guarantee ϵ, the random noise scales to the sensitivity δ: a larger sensitivity requires more noise. However, in many settings, having a trusted curator with access to this information itself is risky. In particular, one must guarantee the security of the server hosting the non-private data in such a way that satisfies individual data providers (e.g. end users of a web application or patients at hospitals). It is much safer to produce workflows where raw information that might be personally identifying is not directly accessible even by designers.

An alternative to the central model is the *local model* of privacy, in which noise is added in a decentralized way before aggregation, e.g., on individual's phones before data is sent back to a central server. This was popularized long before the formalization of differential privacy as the *randomized response* model of introducing privacy for survey data [169], in which random noise is added directly to data based upon information about the domain of possible values prior to aggregation on a potentially unsafe central server.

Definition 5.2 (Local Privacy.)**.** *A mechanism $\mathcal{M}$ is ϵ-locally differentially private if,*

for two observations y and y' and a set $\mathcal{S}$ of possible outputs in the range of $\mathcal{M}$,

$$P[\mathcal{M}(y) \in \mathcal{S}] \leq e^{\epsilon} \cdot P[\mathcal{M}(y') \in \mathcal{S}]. \tag{5.2}$$

The advantage here in a machine learning context is that the data itself is private, and may be used in many downstream applications ranging from broad aggregation queries to the learning of classifiers and unsupervised models. This advantage comes at the cost of the magnitude of noise. Because there is no way to evaluate the global sensitivity of the dataset, noise must be added in a conservative way for any possible sensitivity level. In the context of a matrix of bag-of-words features, the burstiness of words described in Section 2.3 interacts poorly with the local interpretation of the differential privacy inequality of Equation 5.2. Specifically, one must make sure that the distributions of possible outputs of some $\mathcal{M}(\cdot)$ are similar across not just the documents in the final dataset, but any possible document that could arise. This significantly increases the distributional density of nonzero entries in such bag-of-words data, and with an additive random noise mechanism, it will also introduce a number of negative entries. Even in this setting, there is some ambiguity about the privacy guarantee; unless one fixes a maximum number of tokens in a document.

For many applications in this dissertation, we instead apply *limited-precision local privacy*, a generalization of the definition of local privacy introduced in joint work with several co-authors [141]. In this model, the ϵ-differentially-private guarantee of a mechanism is assured only in cases where the difference between two observations in the data is no more than a user-selected N. This definition is analogous to the limited-precision definition of differential privacy introduced by Flood et al. for the central model [46]. In some settings, this may arise from

having a single variable of unknown domain. In the space of bag-of-words features, the similarity of treatment of counts of unique vocabulary words allows this to be re-expressed as a bound on the number of words which must be changed to move from one document's bag-of-words representation to another.

Definition 5.3 (Limited-Precision Local Privacy [141].)**.** *A mechanism $\mathcal{M}$ has (N, ϵ)-limited precision local privacy if, for two observations y and y' whose difference in a private value is no more than N and a set $\mathcal{S}$ of possible outputs in the range of $\mathcal{M}$,*

$$P[\mathcal{M}(y) \in \mathcal{S}] \leq e^{\epsilon} \cdot P[\mathcal{M}(y') \in \mathcal{S}]. \tag{5.3}$$

Formalized for two documents $\vec{y}$ and $\vec{y'}$, the bound on their difference would be $\sum_{i=1}^{V} |y_i - y'_i| \leq N$. This can also be thought of as expressing a new definition for what an observation is in the context of privacy. One could express a natural observation in text as an instance of a word, a sentence, or a full document. With the definition of an observation as a single token, by inverting the group theorem of differential privacy from earlier in this section, this limited precision locally private mechanism would also be an ϵ/K-differentially private mechanism for tokens. If, however, one formalized a text observation as a span of text containing N words, then a mechanism satisfying (N, ϵ)-limited precision local privacy would also satisfy true ϵ-differential privacy for that notion of an observation. The definition of limited-precision local privacy also resembles ideas of geoindistinguishability from geolocation applications [7], and can be seen as a special case of blowfish privacy [66] or profile-based local privacy [54].

5.1.2 Mechanisms of Differential Privacy

The most ubiquitous mechanism of differential privacy is the *Laplace mechanism*, a randomized response mechanism, which scales Laplace-distributed noise to each private numerical value in the data. However, numerous other mechanisms exist to provide differential privacy for different types of data. The family of additive randomized response mechanisms include not only the Laplace mechanism but also the *Gaussian mechanism* [42], which applies Gaussian noise; the *geometric mechanism* [55], which uses the two-sided geometric distribution to produce symmetric integer noise, and the *staircase mechanism* [52], which produces noise using a combination of the uniform and two-sided geometric distributions to produce a continuous analogue of the integer-valued geometric mechanism.

Other mechanisms transform focus on slightly more complex transformations of high-dimensional data. In the realm of text data, generation of a perturbed covariance matrix using the Johnson-Lindenstrauss transform permits creation of a differentially private word-word co-occurrence matrix [19]. For high-dimensional data, the *compressive mechanism* [90] gives differential privacy guarantees by first randomly projecting data to generate a reduced set of random observations, then adding Laplace noise before decompressing the data again to the full dataset size. The work in Chapter 6 describes the fundamental challenges of adding privacy to data in bag-of-words domain, and offers some initial strategies to mitigate these challenges through dimensionality reduction.

Two primary domains have dominated work in high-dimensional differentially private mechanisms: medicine [37, 57, 135, 138, 149, 170] and search query analysis [30, 70, 118, 152, 151, 153, 171]. Several pieces of work look more specifically at distributional semantic models, including topic models [119, **?**], neural

language models [101], and tag recommendation systems [179]. These model-specific works either directly incorporate differential privacy into the inference of the model, or adjust how noise is added to the data to account for specific demands of privacy in these domains. This connects into the work of Chapter 7, which also focuses on adding privacy to the specific BPF model introduce in Section 2.2.3. However, instead of adding private noise during inference, the model is modified to perform better inference by including information about the distribution of the specific private noise mechanism. The next section introduces the original private BPF model in slightly more detail.

5.2 Private BPF

Bayesian Poisson factorization (BPF) models include not only LDA but also models of networks [2], populations [125], and higher-order dynamic models [139, 140]. The broad range of models of counts that can be expressed as BPF models make BPF as a strong candidate to focus on for privacy, as results can generalize beyond the LDA model of text.

Chapter 7 of this dissertation builds on a locally private version of BPF, a method of introducing privacy designed in collaboration with several other researchers [141]. This model incorporates the geometric mechanism [55], an additive randomized response mechanism, to add symmetric integer-valued noise to each count in the observed data input Y before modeling. This is done in a limited-precision locally private way as specified in Definition 5.3, with a pre-specified privacy budget per count observation ϵ and maximum difference between the final count observations for individual observations N. Though

noise may be added to each observation before aggregation into a central matrix or tensor, as the noise function is fully specified by ϵ and N. Though it is possible to vary the choice of ϵ and N for different features (for instance, to specify a higher privacy budget for some words in a vocabulary or documents than for others), the subsequent descriptions assume a unified choice of positive values ϵ and N. The two-sided geometric distribution, the probability distribution used in the geometric mechanism, takes a single parameter, α, computed as

$$0 \leq \alpha = e^{-\epsilon/N} \leq 1. \tag{5.4}$$

It should be noted that there are infinitely many combinations of ϵ and N that have the same resulting α value due to their ratio. So, if one were to add noise for a given N and ϵ and later discovered the choice of N was too low, all is not lost: one could still provide a modified privacy guarantee for a new N' with a less conservative privacy budget $\epsilon' = N\epsilon/N'$.

This added noise fundamentally alters the distribution of the data in two chief ways. First, unlike in true Poisson data, negative observations will be produced when symmetric noise is added. Rounding these count values to be non-negative can help mitigate this, as suggested in the truncated geometric mechanism [55], but this will only round values up and thus will bias the effective noise distribution to be overly positive instead of symmetric. Additionally, while most Poisson factorization models tend to perform well on sparse count data, this random noise operation is inherently dense. Though the mode and expectation of the noise distribution is zero, the expected density of the resulting data is quite dense: even for a desired ratio $\epsilon/N = 2$ per observation, which would imply a relatively modest privacy budget even for $N = 1$, roughly 24% of the

sampled noise will be a nonzero integer. In networks or documents where 1-10% of the observed counts may be zero, this additive noise can dramatically multiply the number of observed nonzero entries. naïve approaches to inference with this change, such as simple direct application of an MCMC algorithm for BPF to the noisy data, learn latent parameters that incorrectly capture much denser correlative structure than existed in the true data.

To account for this, *locally private Bayesian Poisson factorization* (LPBPF) [141] modifies the traditional BPF inference algorithm to be aware not only that random noise has been added, but also about the specific distribution of noise that was added. This does not violate the privacy of the model: according to the post-processing theorem of differential privacy mentioned in Section 5.1.1, modeling private noise does not violate the privacy of the true data so long as it does not explicitly access non-private information about the true data in the process. The initial MCMC inference procedure for PBPF jointly infers latent parameters for the true data and for the additive noise, allowing the BPF parameters to help estimate which entries are more likely to have had noise added.

There are two principle challenges to performing inference for PBPF. The first is determining how to update or resample variables. Though additional analysis of this exact algorithm is left to Chapter 7, the broad observation is that inference relies on a way to directly condition the observed data $\tilde{y}_{\boldsymbol{i}}^{(\pm)}$ on the latent parameters of the true model, μ, instead of merely the sampled count observations from those parameters, $y_{\boldsymbol{i}} \sim \text{Poisson}(\mu)$. We use two alternate expressions of the two-sided geometric distribution to do this. The first directly expresses the noise as the difference of two Poisson variables, $g_{\boldsymbol{i}}^{(+)}$ and $g_{\boldsymbol{i}}^{(-)}$: $\tilde{y}_{\boldsymbol{i}}^{(\pm)} = y_{\boldsymbol{i}} + g_{\boldsymbol{i}}^{(+)} - g_{\boldsymbol{i}}^{(-)}$. The second leverages this to re-express a noisy data

observation $\tilde{y}_{\boldsymbol{i}}^{(\pm)}$ as the difference between two Poisson variables, one containing the true data and positive component of the noise $y_{\boldsymbol{i}} + g_{\boldsymbol{i}}^{(+)}$, and one containing the negative component of the noise $g_{\boldsymbol{i}}^{(-)}$. This distribution can be described using a Skellam distribution conditioned directly on the true model parameters and noise priors, while an additional Bessel-distributed auxiliary variable $m_{\boldsymbol{i}}$ describes the minimum of the two Poisson variables $y_{\boldsymbol{i}} + g_{\boldsymbol{i}}^{(+)}$ and $g_{\boldsymbol{i}}^{(-)}$ [175]. These expressions permit a closed-form MCMC inference algorithm.

The second challenge of inference is performance. The efficiency of existing BPF algorithms largely stems from the exploitation of the sparsity of the data: nonzero count observations may be ignored in sampling. MCMC sampling for LDA exploits the same advantage: rather than needing to consider probabilities of every word type in every document, Equation 2.1 only applies to tokens which actually appear in the observed documents. However, no zero count in the observed data $\tilde{y}_{\boldsymbol{i}}^{(\pm)}$ is guaranteed to correspond to a true zero in $y_{\boldsymbol{i}}$: inference in consideration of the random noise should consider every possible count, not merely those observed to be nonzero. Chapter 7 proposes and justifies a number of strategies to bring performance of this algorithm to a rate comparable to non-private BPF, as well as offering evaluations of this privatization method and new insights into how privacy can improve fit over nonprivate models.

CHAPTER 6

ADAPTIVE STRATEGIES FOR INTRODUCING PRIVATE NOISE TO TEXT FEATURES

Local privacy for bag-of-words models introduces random noise to features in each document before collecting documents together, a useful property for information security. But traditional methods for adding locally private noise overwhelm the true signal in the text data, removing the properties of sparsity and non-negativity often relied upon by distributional semantic models. Our objective in this chapter[1] is to describe properties of a released dataset of bag-of-words features with privacy. Broadly, the released data should nor allow reconstruction of meaningful portions of the original text, but should still provide sufficient information about term co-occurrence such that distributional semantic models inferred on these data are informative about the corpus.

To do this, we consider two modifications to existing local privacy mechanisms. First, we use a generalization of local privacy, called *limited-precision local privacy*, to define our text privacy constraints, which allows us to specify privacy at the level of text spans instead of documents. We argue that this is a more natural privacy guarantee for text, and offer some insights in how to parameterize effectively for this privacy definition. Second, to more accurately retain co-occurrence information, we offer a modified version of classic differential privacy with compression [178] where instead of randomly combining documents, we use a low-rank projection to combine features. We show that the combination of these approaches leads to released data that not only resembles the original data, but also maintains second-order word co-occurrence information, which

[1]The work of this chapter is part of a pending submission alongside coauthors Gregory Yauney, Aaron Schein, Steven Wu, Hanna Wallach, and David Mimno.

we measure through comparisons of topic models. Further, we explore the possibility of using models learned from a public reference corpus with the same language to produce more accurate compression of the same dimension.

6.1 Challenges of Privacy for Bag-of-Words Features

An increasingly popular approach to introducing privacy into distributed data applications is *local privacy*, which has recently been deployed by a number of commercial platforms [45, 156]. As defined in Section 5.1, local privacy offers very appealing privacy guarantees by avoiding the need for a single trusted curator of the data, but also presents many challenges for bag-of-words models. First, bag-of-words observations are difficult to privatize due to their high-dimensional features: the size of a vocabulary for a text dataset can be orders of magnitude larger than the number of words in the document and, according Zipf's law, continues to grow as the number of documents increases. Second, words are bursty: if a word shows up at all in a document, it is likely to appear several times. Since standard local privacy requires any two documents to be indistinguishable after perturbation, we would need to add significant amounts of random noise to any of the vocabulary features, overwhelming the signal in the data.

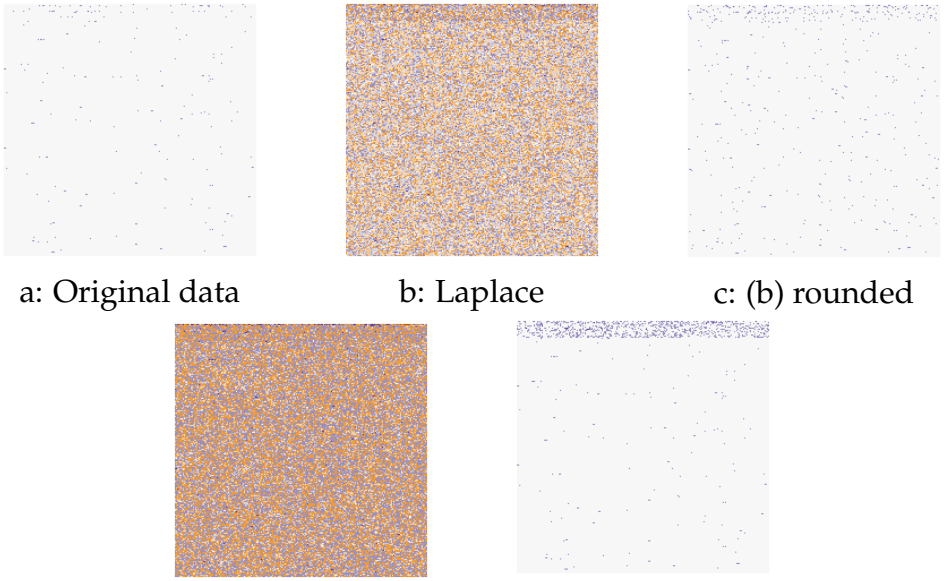
We present three existing privacy mechanisms. We demonstrate these fail on bag-of-words data because they lose either the sparsity or specificity needed to infer models from the word co-occurrence information.

Laplace mechanism. The simplest private mechanism is to add Laplace noise to each feature of the observation. The scale of the noise is proportional to the

ℓ_1-*sensitivity* of the features, defined as the largest difference summed across all features between any two documents in the corpus. Adding noise using this mechanism turns sparse, non-negative integer data into dense data containing many small negative values. As shown in Figure 6.1b, adding Laplace noise converts a 93.6% sparse matrix to a 0.3% sparse matrix. Even after rounding and clipping values, sparsity is still 53.4%. More troubling, the resulting text document was 50 times the size of the original, significantly affecting storage and algorithmic efficiency. Even after post-processing the data to ensure all values are rounded to the nearest non-negative integer as in Figure 6.1c, the data is much more dense, particularly in areas where features used to be very infrequent. This effect only grows with larger corpora and vocabularies and true data, where the burstiness of otherwise rare words produce large sensitivity values. When these sensitivities are used to generate noise, they can explode the size of the corpus and obscure most interesting co-occurrence patterns.

Geometric mechanism. A similar differentially private mechanism is the geometric mechanism [55], which adds integer-valued noise drawn from the two-sided geometric distribution, similar to a geometric distribution reflected around zero. The continuous analogue of this function is proven by [52] to be the optimal mechanism for general differential privacy under standard conditions.[2] But as shown in Figures 6.1d-e, this optimality does not solve the problem of producing viable private data for language modeling. The magnitude of the noise in both the original and rounded versions of the privatized data is larger, but concentrated in high-sensitivity features. While the global distribution of terms is similar to the original, the co-occurrence information between less frequent

[2]There is no universal optimal mechanism for differential privacy of arbitrary queries [22].



a: Original data b: Laplace c: (b) rounded

d: Geometric e: (d) rounded

Figure 6.1: Four views of privatized text collections using the Laplace and geometric mechanisms. Columns represent documents, rows represent vocabulary words in descending order by frequency. Positive values are blue, negative are orange. The Laplace and geometric mechanisms with a privacy budget $\epsilon = 10$ create dense and often negative data. Density is still high even after rounding to the nearest non-negative integer.

words is overwhelmed by the dense frequent words and random noise.

Sketch mechanism An alternate technique specifically targeted for sparse data release subject to privacy constrains is the data sketch [1, 91]. This family of methods adds noise to each document by applying a random projection, then estimating the original document based on the random projection matrix used. An advantage of this method is that the privacy computation is independent of other documents in the dataset, allowing for the post-hoc merging of many datasets as long as private noise was added using the same privacy parameters. As shown in Figure 6.2, however, on synthetic data, this method produces denser and noisier values than the original data. Computationally, the complexity of

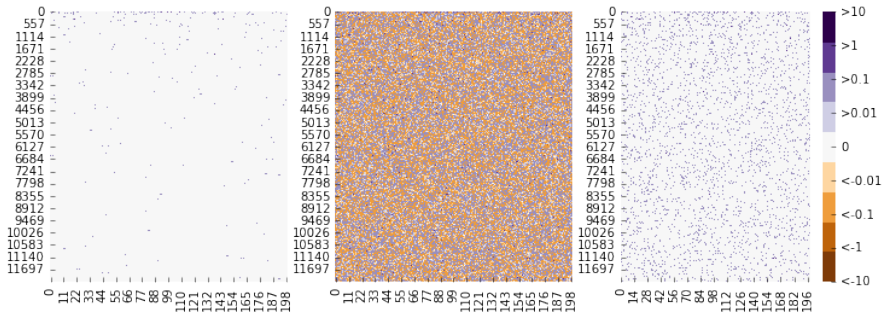


Figure 6.2: Data generated by a sketch algorithm using privacy parameter $\delta = 0.01$ (middle) is much denser than the original data (left), even after rounding to non-negative integers (right).

this algorithm scales inversely with the level of privacy desired: if we need less privacy, processing even a single document can be impractically expensive.

In addition to changing global properties, these transformations also lose semantic properties. To assess this, we train one topic model with 25 topics on each dataset and compare the output topics of the original data to each of the topics trained on privatized datasets. We consider a topic a reasonable match if over half of the top 20 most frequent terms intersect between two topics. Matching performs poorly: with the best-performing method for matching, the Laplace mechanism, three topics have more than 50% overlap in key terms with topics from the original data. For the other two privacy models, no topic exceeds 25% overlap (five shared key terms) with a topic from the model trained on the original data. This level of overlap does not imply quality: for example, one of these 25% overlap topics is represented by the incoherent set of 10 most frequent words, "chocolate sacramento hangar tour bicycle fun rocks river texas las."

In the case of training distributional semantic models using bag-of-ngrams data, a privacy-preserving data transformation should retain two things: the accuracy of frequency queries for non-unique words or ngrams on specified

subsets of the document corpus, and higher-order trends of word co-occurrence across the whole corpus. Both families of existing methods focus on retaining the first kind of data, limiting noise on more common features. Both randomization and attribute censorship tend to remove or muddy information about rarer events that might distinguish only small subsets of the documents, such as rarer co-occurrences or individual terms in a corpus.

For two large domains of private text mining mentioned in Section 5.1.2, medicine and search query analysis, this level of privacy is appropriate: identifying a record's identity may leak sensitive information about an individual. However, there are many cases where this level of anonymity is not required to protect what is sensitive about the underlying data, such as text under copyright or anonymous papers under submission to a conference. A more appropriate goal for these settings is to protect aspects of the contents of that text that are *unique* to that document, such that a sensitive phrase within the text cannot be reconstructed from frequency data even if the source of the text is known.

One objective in text reconstruction prevention, then, is to treat small passages as the object of privacy, not documents, through the application of limited-precision local privacy from Definition 5.3. Limited-precision differential privacy targets a different goal from typical differential privacy for these corpora. Instead of needing to ensure that a document's presence or absence is statistically difficult to determine, the objective of limited-precision privacy is reduced to the presence or absence of a particular *portion* of a document. This presupposes that passages of the document themselves represent something valuable, as opposed to the identity of the document. In typical differential privacy, it is necessary to protect against someone who has access to a partial database of potential rows

to ensure that the presence of a row in the database is not revealed. However, when preventing text reconstruction, we assume that if someone has enough information to uniquely identify that a passage was part of a document in a database, then she already has all of the interesting information the database had about that passage. In protecting intellectual property, this definition intuitively makes sense: anyone attempting to identify whether a particular sentence is or is not somewhere in the corpus already must know enough about the sentence to have accessed it already. This weaker privacy constraint is not appropriate for all cases: in a medical transcript merely recording the fact that a condition was discussed, related term can violate privacy as much as the condition name.

6.2 Applying Compression with Differential Privacy

We next propose a mechanism for operationalizing limited-precision local privacy. An easy way to retain information about large-scale correlation in the data in a privacy mechanism is through data compression. Simple random projections are known to be sufficient for some of these methods [178]. In our case, we add noise to a compressed representation of the data and then reverse the compression, so that noise will be distributed over multiple elements of the original data. We refer to this process of combining columns as *horizontal compression*, distinguished from the typical method of *vertical* compression that combine rows (e.g. documents) to make a synthetic database containing a smaller number of summary documents to analyze. This horizontal approach relies on the shared domain of bag-of-words features as counts of tokens within a single text collection.

Bag-of-words text data is often amenable to low-rank representations. Many

distributional semantic models, including LDA [8], latent semantic analysis/indexing (LSA/LSI) [38], and various neurally-inspired word embedding models [85, 107, 121] can be written as a low-rank matrix decomposition of a document-word matrix. Not only do these factorizations produce reconstructions close to the original data, the resulting bases for words and documents can be used to approximate similarity in ways that can generalize to contexts outside of the corpus. Intuitively, a low-rank basis learned from text in a language ℓ should transfer well to similar text in the same language ℓ. Using compression of these features can actually benefit machine learning models [162, 172].

In horizontal compression, we should be able to use this property to produce better compression for our noise data. In contrast to previous methods such as using data sketches, we compress each document to a smaller representative number of columns before applying a privacy mechanism, and re-expand the data afterwards. To do this, we choose groups of the original features to sum to a single feature, then add noise to that compressed version based on our original entry-wise privacy budget ϵ using the geometric mechanism, [55] under limited-precision local privacy guarantees. Finally, we reverse this public compression: using total corpus-wide frequencies of each feature with added two-sided geometric noise, we can randomly sample original features from a multinomial distribution with the normalized frequencies for the corresponding original features as the categorical sample prior and the differentially private count of the new compressed feature as the sample size.

We test several approaches to grouping features. The first is random, in which features are uniformly at random assigned to one of the compressed feature indices. Unlike sketches, this allows us to ensure that frequency information is

retained somewhere in our compressed data, as well as retaining where features were randomly projected to. The second is more deterministic: using global feature frequencies for each word in the whole corpus privatized with the geometric mechanism, we assign the features to compressed feature indices in a round-robin fashion in order of frequency, such that the most frequent features are all assigned to different latent dimensions. The third uses the observation that we can use learn a public basis for compressing our data. In our case, we use pretrained word embeddings of dimension 100[3] generated using GloVe [121] to cluster features together using K-means. Differential privacy is immune to post-processing, so using a rank-k compression for documents with number of features m spends k/m of the privacy budget of the original algorithm, plus a small additional $m\epsilon$ cost to determine global frequency information.

One concern caused by this approach is that the appropriateness of the public basis to the private data determines the quality of the reconstruction. A basis generated from a reference corpus of news articles might be a poor choice for a corpus of restaurant reviews, as it may devote relatively little of its modeling power to descriptions of specific cuisines. However, the same lack of fit can also work to the advantage of those interested in privatizing the corpus: the choice of reference corpus provides a way for data owners to specify which kinds of data should be rendered clearly by the resulting compression and which should be de-emphasized. Specific, unusual language in a collection might be the most important to privatize, as it may signal rare, personally identifying events.

Another concern is that the basis required to project the data in a useful way must be large or overly specific. We find this is true to some extent in our results, as bases trained on small datasets or with small dimension (for instance, below

[3]We found that embeddings of larger or smaller dimension made little difference in this step.

Compression Type	ϵ	N	Jaccard
uncompressed	0.5	10	0.108
random	0.5	1	0.201
frequency order	0.5	1	0.301
uncompressed	0.5	1	0.408
random	5	1	0.281
frequency order	5	1	0.310

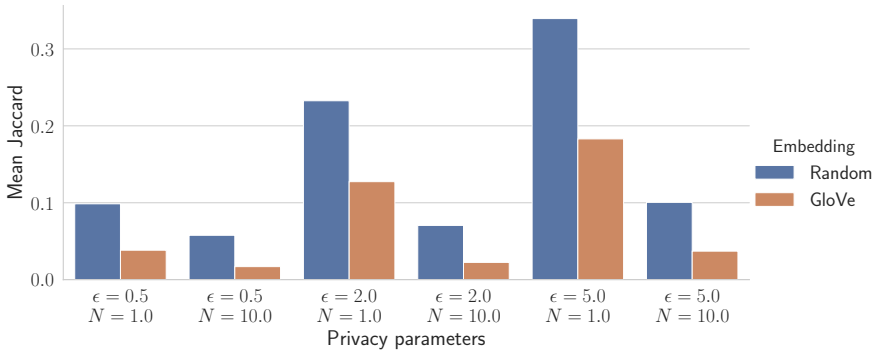
Table 6.1: For synthetic data, Jaccard similarity between the topics from nonprivate data and different private projections using the same total privacy budget. Listed epsilons are per-feature.

100) fail to produce meaningful data correlation. However, public bases can easily be extracted from available massive datasets such as Wikipedia, with a sufficient vocabulary to encapsulate many subjects. The GloVe embeddings we use for our projection [121] were pretrained general-purpose English embeddings learned from a 6 billion word corpus that combines of 2014 English Wikipedia[4] and the Gigaword corpus [120]. These texts should capably represent many subjects.

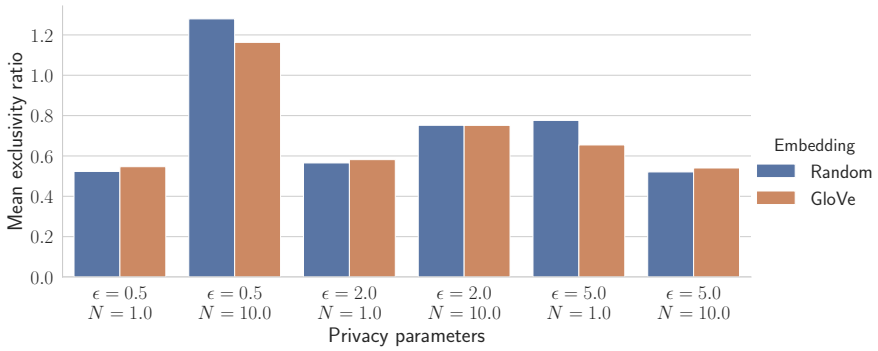
6.3 Experiments

Though we have provable differential privacy guarantees based on spending our privacy budget using horizontal compression, this result does not guarantee that the output data is clean enough to produce useful semantic models. We experimentally validate the efficacy of our model first using synthetic data generated from an LDA generative model [18].

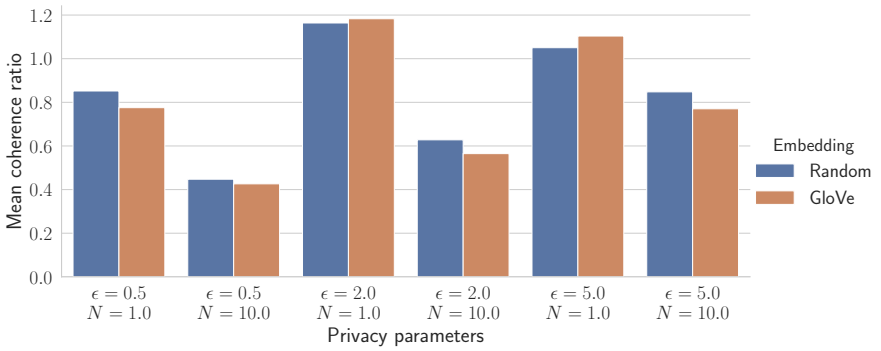
[4]From `https://dumps.wikimedia.org/`



a: Mean Jaccard similarity coefficient between all 50 pairs of private and nonprivate topics when ranked by most overlapping words in the top 20.



b: Mean ratio across all 50 topics of private exclusivity to nonprivate exclusivity of 0.2642.



c: Mean ratio across all 50 topics of private per-word coherence to nonprivate per-word coherence of -1.9167.

Figure 6.3: Metrics comparing topics trained with limited-precision local privacy to topics trained in the absence of privacy. 1000 latent dimensions were used for all experiments. For GloVe, metrics were evaluated using an embedding dimension of 100.

6.3.1 Synthetic data generation

We use for our data generation the generative model underlying latent Dirichlet allocation (LDA) [18]. In this model, we first generate K topics, or probability distributions over a vocabulary of V words, as categorical distributions whose priors are sampled from a sparse symmetric Dirichlet distribution of dimension V and hyperparameter $\beta = 0.01$. Each of the D documents in our corpus has a corresponding categorical distribution over the K topics with a prior drawn from a sparse symmetric Dirichlet distribution with hyperparameter of $\alpha = \frac{5}{K}$. To generate a document of length N, for each of the N words, one samples a topic from the document's document-topic distribution, then a word from the corresponding topic's topic-word distribution. The advantage of synthetically generating LDA data here is that we can actually know the original document-topic and topic-word distributions that generated our data. To run our experiments, we vary our generative process to generate 5 topics over a corpus of 100 documents, 100 vocabulary, and length-100 documents.

6.3.2 Real data

We also perform experiments using a dataset of U.S. consumer complaints about financial products and services released by the CFPB [24]. This was pared down to a subset of 9,528 documents from Illinois, each with 100-500 words of text and a common set of stopwords removed as defined by the MALLET stoplist.

Embedding	ϵ	N	Match	Jaccard	Top 20 words in private topic	Corresponding nonprivate topic
Random	1	5	5^{th}	0.5385	**them number phone calling call called calls company are** send **day work** debt **times stop about** contact said again now	**call phone number calls them calling are called company times day work stop** cell message when but **about** after received
Random	1	5	10^{th}	0.4815	**account report reporting information** when **all** date until **has** company **bureaus removed** collection **reported** opened **off** accounts **agencies being been**	**report reporting reported bureaus all information account** late **off** dispute **removed** are **agencies being has** negative reports correct still **been**
Random	1	5	25^{th}	0.3333	**them** told **never but because any did** when about **will** also **even** asked their had still **which** sent company **due**	bankruptcy **but because** filed chapter **which** are **them even will never due** proof **did any** advised discharged though loan included
GloVe	1	5	5^{th}	0.2903	**been has** also told **them but** called **company** account there had you **now since** call **out time** who their when	**been has time** times over **them but now** months get **since** years again still **out** last several issue are **company**
GloVe	1	5	10^{th}	0.2903	**them are all** you because call **but there** get she **what one any** time please more than need still her	**are** their these can **there what** when other **them** even **but all** how **one** only **any** same does just people
GloVe	1	5	25^{th}	0.2903	**equifax has breach** mine which **security** account reports **their company** also reporting inaccurate investigate any disputed rights please called over	information **equifax their security** personal data **breach** freeze **has** usaa are fraud should social **company** myself affected financial monitoring place
Random	$\frac{1}{2}$	10	5^{th}	0.0819	pleading left **interest** harassed **amount** confirmation **month** missing filed owed opinion letter whom tell thieves car handled provide conduct authorized	balance **interest amount** paid statement pay full payment charges off due charged account which charge made bmo **month** billing statements
Random	$\frac{1}{2}$	10	10^{th}	0.0819	able located job including month she applied **what** seem dispute apart **did** money days well times walmart regarding **been** during	had would were **did** about told time after there **what** when informed which contacted could **been** should them never being
Random	$\frac{1}{2}$	10	25^{th}	0.0526	dont stating bill proof local previous opening fargo life call **get** each consumer decision fees pursuing when thirty **payments** proper	pay help work **get** make can month time **payments** payment them paying but job had could will because financial now
GloVe	$\frac{1}{2}$	10	5^{th}	0.0256	issuers alerted infraction identification color scrambling community damaged clerks ltd judgments carefully **paid** succeeded grief automobile these assisting recent conclusion	balance interest amount **paid** statement pay full payment charges off due charged account which charge made bmo month billing statements
GloVe	$\frac{1}{2}$	10	10^{th}	0.0256	incident randomly lawyer treated mae yearly thee begged tcf restored before **hard** homeless reimburse discrepancies york residential inadvertently signup documenting	report inquiries inquiry transunion removed **hard** were remove companies would contacted requested did these bureau inquires are reporting union all
GloVe	$\frac{1}{2}$	10	25^{th}	0.0256	minor unlawful qualifying practicing held although breached word trans incurred institutions payday **bank** preparing appalled anxiety contribute intend wage wrong	**bank** america third branch had boa fifth tcf about which made did but then contacted fraud banks after only what

Table 6.2: Comparison of private and nonprivate topics. Pairs of topics were matched according to highest Jaccard similarity coefficients. Words in bold are shared between both topics

6.3.3 Evaluations

To evaluate the closeness of the private data to the original, we use a few different metrics to compare to models learned on nonprivate data. First, we compare the top 20 terms of private topics with nonprivate topics to find the Jaccard similarity between the closest corresponding topics. Second, we study the exclusivity of

Embedding	ϵ	N	Percentage increase (mean $\pm$ std)
Random	$\frac{1}{2}$	1	12.39 ± 8.44
Random	$\frac{1}{2}$	10	131.45 ± 87.80
Random	2	1	1.73 ± 1.21
Random	2	10	32.42 ± 21.81
Random	5	1	0.084 ± 0.072
Random	5	10	12.38 ± 8.42
GloVe	$\frac{1}{2}$	1	12.49 ± 8.41
GloVe	$\frac{1}{2}$	10	131.92 ± 88.13
GloVe	2	1	1.77 ± 1.21
GloVe	2	10	32.57 ± 21.87
GloVe	5	1	0.087 ± 0.071
GloVe	5	10	12.50 ± 8.40

Table 6.3: Mean percentage increase in words per document after adding noise. With strong privacy parameters, the length of documents increases substantially, but still retains sparsity.

each topic, measured as how unique the terms in that topic are to the topic as compared to other topics in the same model [16]. Finally, in the real data, we consider topic coherence [111] to measure how well the top words of a topic actually adhere to a shared subject in meaning space.

6.3.4 Results

In the synthetic data in Table 6.1, we see that if we compare based on per-entry noise added, the uncompressed model appears to do better. However, the total privacy budget used in synthetic experiments tests a latent dimension of rank 10 compared to the original dimension of rank 100. A fair comparison by total privacy budget is actually between a factor of 10 difference of ϵ/N. Comparing the uncompressed model with $\epsilon = 0.5, N = 1$ and the random and frequency projections of $\epsilon = 5, N = 1$, we can see these projections do close to as well

as the unprojected data. However, if we compare the stronger privacy level $\epsilon = 0.5, N = 10$ from the uncompressed data with the privacy level $\epsilon = 0.5, N = 1$ compressed, we see that both random and frequency order actually outperform the uncompressed model in terms of topic Jaccard similarity. Using the specific frequency ordering of features instead of random feature projections, even with differential privacy of the counts, seems to improve quality.

An additional positive feature of the compressed method is the retained sparsity of the data. Where true random noise on raw features multiplied the length of documents by 50 times even with modest privacy in Section 6.1, we see in Table 6.3 that documents stay close to the same average length.

In considering evaluations of our real data in Figure 6.3, we can see that random projections actually often fare better than use of a pre-trained embedding. The intuition behind this relates to what we observe in the synthetic data: there is a benefit to avoiding grouping together multiple frequent features into one. As random methods are more likely to evenly distribute frequent original features among the compressed features, these produce better reconstructions of the original data. This is also reflected in Table 6.2, where topics look more similar to those in the nonprivate data when learned using random compression. We therefore learn that, while auxiliary information about term frequency is useful, models capturing term similarity may be subtler to apply to compression without attention to the effects of these frequencies. However, we observe from experiments with a much lower rank of data that the model retains comparable coherence scores for topics even with a small privacy budget ϵ.

6.4 Discussion

Local privacy for text is an important but difficult goal. In practice, researchers often find that access to text collections is the single most important limiting factor for text-as-data research, rather than any algorithmic considerations. Having a greater ability to assure collection owners and communities of specific privacy guarantees would go a long way towards opening up more sensitive — and interesting — collections. But the specific nature of text data, with its high dimensionality and sparsity, make it difficult to apply standard results from differential privacy that have been successful in other domains. In this work we provide a promising new alternative, by using public compression mechanisms to help condense features. We show that while in light privacy settings, these may seem ineffective, with stronger privacy goals this method can help conserve privacy budget while also forcing a more sophisticated method of interpretation.

CHAPTER 7

EFFICIENT INFERENCE OF PRIVATE BPMF IN THE TOPIC SETTING

Locally private Bayesian Poisson factorization (LPBPF) [141], as introduced initially in Section 5.2, gives a generative model for Poisson-generated count data that has been privatized through the addition of two-sided geometric noise [55]. This work provides a locally private inference strategy for not only LDA, but also a variety of other low-rank generative models of count data in the Bayesian Poisson Factorization (BPF) family of models. However, to replace non-private models in use by social scientists, private models must be easy to infer, close to the nonprivate model, and interpretable. While the initial proposed MCMC algorithm for LPBPF inference can produce comparable models for modest privacy guarantees, it runs orders of magnitude more slowly than the non-private algorithm and require substantially more memory.

This work[1] presents both alternative methods for bringing private model inference close to the same speed as non-private BPF inference and evaluations demonstrating the comparative quality of the resulting models. This work also highlights a surprising results of LPBPF model inference on real-world text data: introducing a small amount of private noise may improve model inference. Using a variety of evaluations from Section 2.3, we interpret why this might occur and what it suggests about Poisson factorization models as a whole.

[1]The work of this chapter reproduces text and results from existing published workshop papers [141, 142, 145].

7.1 An Introduction to LPBPF Inference

The full LPBPF model combines the generative processes for standard Bayesian Poisson factorization and the geometric mechanism. For each entry in the observed matrix or tensor of counts, indexed by a position subscript $\boldsymbol{i}$, the true data count $y_{\boldsymbol{i}}$ is unobserved. Instead, we observe a noised version of it, $\tilde{y}_{\boldsymbol{i}}^{(\pm)} = \tau_{\boldsymbol{i}} + y_{\boldsymbol{i}}$, where $\tau_{\boldsymbol{i}} \sim \text{2SGeom}\,(\alpha_{\boldsymbol{i}})$ is noise drawn from a two-sided geometric distribution scaled by a privacy parameter $\alpha_{\boldsymbol{i}} \in [0, 1]$ given in Equation 5.4 to satisfy a user-specified limited-precision local privacy guarantee (Definition 5.3). MCMC inference proceeds by treating the true data $y_{\boldsymbol{i}}$ itself as a latent variable, resampling it and variables describing the generated noise from derived conditional distributions with respect to all other latent quantities.

To produce a viable closed-form inference algorithm in MCMC, our initial algorithm relied on three formulations of the generative process of limited-precision local privacy. These formulations use different variable distributions and interdependencies to express the same joint posterior given the data. By using these formulations, we can take advantage of useful statistical properties of Poisson distributions (e.g., that the sum of Poisson distributions is also a Poisson distribution) to express an iterative MCMC inference algorithm with variable and parameter updates drawn from closed-form samplers.

Aside from the original formulation of two-sided geometric noise as its own distribution, $\tau_{\boldsymbol{i}}$ may also be written as the difference of two Poisson variables $g_{\boldsymbol{i}}^{(+)}$ and $g_{\boldsymbol{i}}^{(-)}$, generated with Gamma-distributed priors $\lambda_{\boldsymbol{i}}^{(+)}$ and $\lambda_{\boldsymbol{i}}^{(-)}$, respectively. We define auxiliary variables $\tilde{y}_{\boldsymbol{i}}^{(+)} = y_{\boldsymbol{i}} + g_{\boldsymbol{i}}^{(+)}$ and $\tilde{y}_{\boldsymbol{i}}^{(\pm)} = y_{\boldsymbol{i}} + g_{\boldsymbol{i}}^{(+)} - g_{\boldsymbol{i}}^{(-)}$ to represent a two-step process of adding the two "sides" of the noise function

as Poisson variables. We introduce another auxiliary variable, $m_{\boldsymbol{i}}$, to describe the minimum of $\tilde{y}_{\boldsymbol{i}}^{(+)}$ and $g_{\boldsymbol{i}}^{(-)}$. The sign of observed variable $\tilde{y}_{\boldsymbol{i}}^{(\pm)}$ is sufficient to determine which variable $m_{\boldsymbol{i}}$ represents: if $\tilde{y}_{\boldsymbol{i}}^{(\pm)} \leq 0$, then $m_{\boldsymbol{i}} = \tilde{y}_{\boldsymbol{i}}^{(+)}$, and if $\tilde{y}_{\boldsymbol{i}}^{(\pm)} \geq 0$, then $m_{\boldsymbol{i}} = g_{\boldsymbol{i}}^{(-)}$. If $\tilde{y}_{\boldsymbol{i}}^{(\pm)} = 0 = g_{\boldsymbol{i}}^{(-)}$, both these equations are satisfied.

The key feature of this variable scheme is that it permits a statistical formulation of the generative process that joins together quantities drawn from the Poisson parameters of interest (e.g., the latent topics) and the noise distribution:

$$\left(\tilde{\boldsymbol{y}}_{\boldsymbol{i}}^{(+)} \mid - \right) \sim \text{Multinom}\left(\tilde{y}_{\boldsymbol{i}}^{(+)}, (\lambda_{\boldsymbol{i}}^{(+)}, \mu_{\boldsymbol{i}1}, \ldots, \mu_{\boldsymbol{i}K})\right). \tag{7.1}$$

Here, $\tilde{\boldsymbol{y}}_{\boldsymbol{i}}^{(+)} = (g_{\boldsymbol{i}}^{(+)}, y_{\boldsymbol{i}1}, \ldots, y_{\boldsymbol{i}K})$ is a vector of latent sub-counts that sum to $\tilde{y}_{\boldsymbol{i}}^{(+)}$. The first of these sub-counts, $g_{\boldsymbol{i}}^{(+)}$, represents the positive Poisson noise added to the sensitive data $y_{\boldsymbol{i}} = \sum_{k=1}^{K} y_{\boldsymbol{i}k}$, while the latter sub-counts correspond to each latent component in non-private Poisson factorization.

To infer LPBPF, one may first infer the sums of Poisson distributions of the underlying data, then thin them using multinomial and binomial draws. In these equations, delta functions $\mathbb{1}(\cdot)$ describe mathematical constraints that must be met deterministically according to the definitions of auxiliary variables. Some distributions have their own implicit delta function: for example, a multinomial distribution Multinom $(()\, \vec{x}; n, \vec{p})$, which describes the probability of drawing a vector $\vec{x}$ of counts given n draws from a categorical distribution with probabilities $\vec{p}$ of each outcome, implicitly constrains the sum of $\vec{x}$ to be equal to n.

Note that we sometimes write the probability parameters in the multinomial and binomials as unnormalized vectors instead of proper probability distributions. Also, according to the original definition of BPF in Section 2.2.3, $\mu_{\boldsymbol{i}} = \sum_k \prod_d \theta_{i_d,k}^{(d)}$, where i_d is the index into the dth dimension of index vector $\boldsymbol{i}$.

This gives the following equation as the full generative process of LPBPF:

$$
\begin{aligned}
P\left(g_{\boldsymbol{i}}^{(+)}, g_{\boldsymbol{i}}^{(-)}, (y_{\boldsymbol{i}k})_{k=1}^{K}, y_{\boldsymbol{i}}, \tilde{y}_{\boldsymbol{i}}^{(+)}, \tilde{y}_{\boldsymbol{i}}^{(\pm)}, m_{\boldsymbol{i}}\right) = \\
\text{Poisson}\left(g_{\boldsymbol{i}}^{(-)}; \lambda_{\boldsymbol{i}}^{(-)}\right) \text{Poisson}\left(\tilde{y}_{\boldsymbol{i}}^{(+)}; \lambda_{\boldsymbol{i}}^{(+)} + \mu_{\boldsymbol{i}}\right) \\
\text{Binom}\left((y_{\boldsymbol{i}}, g_{\boldsymbol{i}}^{(+)}); \tilde{y}_{\boldsymbol{i}}^{(+)}, (\mu_{\boldsymbol{i}}, \lambda_{\boldsymbol{i}}^{(+)})\right) \text{Mult}\left((y_{\boldsymbol{i}k})_{k=1}^{K}; y_{\boldsymbol{i}}, \left(\prod_{d} \theta_{i_d,k}^{(d)}\right)_{k=1}^{K}\right) \\
\mathbb{1}\left(\tilde{y}_{\boldsymbol{i}}^{(\pm)} = \tilde{y}_{\boldsymbol{i}}^{(+)} - g_{\boldsymbol{i}}^{(-)}\right) \mathbb{1}\left(m_{\boldsymbol{i}} = \min\{\tilde{y}_{\boldsymbol{i}}^{(+)}, g_{\boldsymbol{i}}^{(-)}\}\right).
\end{aligned} \tag{7.2}
$$

Another way to describe this process flips the order of generation from the actual process used in creation of the private data. Here, we describe first generating $\tilde{y}_{\boldsymbol{i}}^{(\pm)}$ and minimum $m_{\boldsymbol{i}}$ as Skellam and Bessel random variables conditioned on what would have been the priors of $\tilde{y}_{\boldsymbol{i}}^{(+)}$ and $g_{\boldsymbol{i}}^{(-)}$. We then compute $\tilde{y}_{\boldsymbol{i}}^{(+)}$ and $g_{\boldsymbol{i}}^{(-)}$ via their deterministic relationship and, finally, thin $\tilde{y}_{\boldsymbol{i}}^{(+)}$ into the vector of counts per latent component $\tilde{\boldsymbol{y}}_{\boldsymbol{i}}^{(+)}$ using binomial and multinomial draws:

$$
\begin{aligned}
P\left(g_{\boldsymbol{i}}^{(+)}, g_{\boldsymbol{i}}^{(-)}, (y_{\boldsymbol{i}k})_{k=1}^{K}, y_{\boldsymbol{i}}, \tilde{y}_{\boldsymbol{i}}^{(+)}, \tilde{y}_{\boldsymbol{i}}^{(\pm)}, m_{\boldsymbol{i}}\right) = \\
\text{Skel}\left(\tilde{y}_{\boldsymbol{i}}^{(\pm)}; \lambda_{\boldsymbol{i}}^{(+)} + \mu_{\boldsymbol{i}}, \lambda_{\boldsymbol{i}}^{(-)}\right) \text{Bes}\left(m_{\boldsymbol{i}}; |\tilde{y}_{\boldsymbol{i}}^{(\pm)}|, 2\sqrt{\lambda_{\boldsymbol{i}}^{(-)}(\lambda_{\boldsymbol{i}}^{(+)} + \mu_{\boldsymbol{i}})}\right) \\
\mathbb{1}\left(\tilde{y}_{\boldsymbol{i}}^{(+)} = m_{\boldsymbol{i}}\right)^{\mathbb{1}(\tilde{y}_{\boldsymbol{i}}^{(\pm)} \leq 0)} \mathbb{1}\left(g_{\boldsymbol{i}}^{(-)} = m_{\boldsymbol{i}}\right)^{\mathbb{1}(\tilde{y}_{\boldsymbol{i}}^{(\pm)} > 0)} \mathbb{1}\left(\tilde{y}_{\boldsymbol{i}}^{(\pm)} = \tilde{y}_{\boldsymbol{i}}^{(+)} - g_{\boldsymbol{i}}^{(-)}\right) \\
\text{Binom}\left((y_{\boldsymbol{i}}, g_{\boldsymbol{i}}^{(+)}); \tilde{y}_{\boldsymbol{i}}^{(+)}, (\mu_{\boldsymbol{i}}, \lambda_{\boldsymbol{i}}^{(+)})\right) \text{Mult}\left((y_{\boldsymbol{i}k})_{k=1}^{K}; y_{\boldsymbol{i}}, \left(\prod_{d} \theta_{i_d,k}^{(d)}\right)_{k=1}^{K}\right).
\end{aligned} \tag{7.3}
$$

This second factorization enables direct closed-form inference: unlike in Equation 7.2, $\tilde{y}_{\boldsymbol{i}}^{(\pm)}$ can be connected directly to the parameters of interest for the latent model, $\mu_{\boldsymbol{i}k}$, without necessarily having a good direct estimate of $y_{\boldsymbol{i}}$. The full MCMC algorithm is presented in Algorithms 7.1-7.3 below.

Algorithm 7.1: Initialization for LPBPF [141].

function INITIALIZATION(a, b, α)
  $\phi, \theta \sim$ Gamma (a, b);
  for all indices $\boldsymbol{i}$ in $\tilde{\boldsymbol{Y}}^{(\pm)}$ **do**
    $\lambda_{\boldsymbol{i}}^{(+)}, \lambda_{\boldsymbol{i}}^{(-)} \sim \text{Exp}\left(\frac{\alpha}{1-\alpha}\right)$;
  end for
  for $d = 1$ to M **do**
    $\hat{\theta^{(d)}} = 0$; ▷ Dimensions should be specified for each latent parameter matrix θ.
  end for
end function

Algorithm 7.2: MCMC updates for privacy variables in LPBPF.

function PRIVATEINFERENCE($\tilde{\boldsymbol{Y}}^{(\pm)}$ α, a, b)
  for all indices $\boldsymbol{i}$ in $\tilde{\boldsymbol{Y}}^{(\pm)}$ **do**
    $m_{\boldsymbol{i}} \sim \text{Bessel}\left(|\tilde{y}_{\boldsymbol{i}}^{(\pm)}|, 2\sqrt{(\lambda_{\boldsymbol{i}}^{(+)} + \mu_{\boldsymbol{i}})\lambda_{\boldsymbol{i}}^{(-)}}\right)$; ▷ Sample $m_{\boldsymbol{i}}$
    if $m_{\boldsymbol{i}} < 0$ **then** ▷ Compute $\tilde{y}_{\boldsymbol{i}}^{(+)}$ and $g_{\boldsymbol{i}}^{(-)}$
      $\tilde{y}_{\boldsymbol{i}}^{(+)} = m_{\boldsymbol{i}}$;
      $g_{\boldsymbol{i}}^{(-)} = \tilde{y}_{\boldsymbol{i}}^{(+)} - \tilde{y}_{\boldsymbol{i}}^{(\pm)}$;
    else
      $g_{\boldsymbol{i}}^{(-)} = m_{\boldsymbol{i}}$;
      $\tilde{y}_{\boldsymbol{i}}^{(+)} = \tilde{y}_{\boldsymbol{i}}^{(\pm)} + g_{\boldsymbol{i}}^{(-)}$;
    end if

    $y_{\boldsymbol{i}} \sim \text{Binom}\left(\tilde{y}_{\boldsymbol{i}}^{(+)}, \frac{\mu_{\boldsymbol{i}}}{\mu_{\boldsymbol{i}} + \lambda_{\boldsymbol{i}}^{(+)}}\right)$; ▷ Sample $y_{\boldsymbol{i}}$, compute $g_{\boldsymbol{i}}^{(+)}$
    $g_{\boldsymbol{i}}^{(+)} = \tilde{y}_{\boldsymbol{i}}^{(+)} - y_{\boldsymbol{i}}$;

    $\lambda_{\boldsymbol{i}}^{(+)} \sim \text{Gamma}\left(g_{\boldsymbol{i}}^{(+)} + 1, \alpha\right)$; ▷ Sample $\lambda_{\boldsymbol{i}}^{(+)}, \lambda_{\boldsymbol{i}}^{(-)}$
    $\lambda_{\boldsymbol{i}}^{(-)} \sim \text{Gamma}\left(g_{\boldsymbol{i}}^{(-)} + 1, \alpha\right)$;
  end for
end function

Algorithm 7.3: Full LPBPF algorithm.

function LPBPFINFERENCE($\tilde{\boldsymbol{Y}}^{(\pm)}$, α, a, b, B, S)
  INITIALIZATION(a, b, α)
  for $i = 1$ to $B + S$ **do**
    PRIVATEINFERENCE($\tilde{\boldsymbol{Y}}^{(\pm)}$, α, a, b); ▷ From Algorithm 7.2
    BPFINFERENCE($y, \mu, \theta^{(d)}$); ▷ This is standard non-private inference using y to estimate μ and each θ.
    if $i > B$ and $i \,\%\, 100 == 0$ **then**
      for $d = 1$ to M **do**
        $\hat{\theta^{(d)}} = \hat{\theta^{(d)}} + \theta^{(d)}$;
      end for
    end if
  end for
  for $d = 1$ to M **do**
    $\hat{\theta^{(d)}} = \hat{\theta^{(d)}} / (\frac{S}{100})$;
  end for
  return $\hat{\theta^{(d)}}$; ▷ Return all latent parameters.
end function

7.2 Improving MCMC Performance

The MCMC inference procedure described above is guaranteed to eventually converge to true samples of the posterior. However, this method has significant issues scaling to larger data matrices. There are three primary reasons for this, but each can be addressed through modifications to the standard MCMC algorithm.

Scheduling parameter updates (SCHEDULE). First, though our method's sampling procedure for the random noise variables converges to samples from the posterior, it does not guarantee stability among those samples. From a privacy

perspective, this is ideal: the goal is not to reverse the addition of noise to individual entries, but to statistically account for this noise in the inference of our more general model parameters. However, the high variance among samples of the estimated true count data $y_{\boldsymbol{i}}$ is sufficiently high in practice that it may interfere with inference of the Poisson factorization model parameters.

To allow these $y_{\boldsymbol{i}}$ parameters to converge, we offer an alternative schedule of resampling the parameters $\lambda_n^{(s)}$ and latent variables $g_n^{(s)}$ at a much slower rate than the Poisson latent variables and parameters. We choose r to be the number of iterations of BPF inference per single iteration of inference of the privacy variables. Strong results arose from $r = 100$, or updating the privacy variables once per 100 iterations of Poisson factorization parameter updates produced. As the inferred noise variables are also the most expensive to resample, we find that this greatly reduces the running time. This strategy also retains the theoretical guarantee that the MCMC algorithm will converge.

Approximating Bessel samples with iterated conditional modes (ICM**).** As mentioned earlier in this section, the sampling procedure to infer the added two-sided geometric noise is slow. This is due in particular to the expense of sampling m from the Bessel distribution. Though it is possible to sample from a Bessel distribution efficiently without computing Bessel functions or ratios [39], it still requires an iterative process that is considerably more time-consuming than that of the other distributions involved in inference.

An alternate approach, motivated by the method of iterated conditional models (ICM) [13], is simply to estimate the value of m as the mode of that Bessel. At each sampling step, we set m as the mode of the Bessel according to its

Algorithm 7.4: LPBPF with the SCHEDULE approach for resampling private variables less frequently via PRIVATEINFERENCE.

function LPBPFSCHEDULEINFERENCE($\tilde{\boldsymbol{Y}}^{(\pm)}$ α, a, b, B, S, r)
  INITIALIZATION(a, b, α)
  for $i = 1$ to $B + S$ **do**
    if $i \,\%\, r == 0$ **then** ▷ SCHEDULE change to reduce frequency
      PRIVATEINFERENCE($\tilde{\boldsymbol{Y}}^{(\pm)}$, α, a, b);
    end if
    BPFINFERENCE($y, \mu, \theta^{(d)}$);
    if $i > B$ and $i \,\%\, 100 == 0$ **then**
      for $d = 1$ to M **do**
        $\theta^{\hat{(d)}} = \theta^{\hat{(d)}} + \theta^{(d)}$;
      end for
    end if
  end for
  for $d = 1$ to M **do**
    $\theta^{\hat{(d)}} = \theta^{\hat{(d)}} / (\frac{S}{100})$;
  end for
  return $\theta^{\hat{(d)}}$;
end function

formulation in [39], with the parameters $\nu_n = |\tilde{y}_n^{(\pm)}|$ and $a_n = 2\sqrt{(\lambda_n^{(+)} + \mu_n)\lambda_n^{(-)}}$ conditioned on the observed data and other sampled parameters and variables:

$$m_n := \left\lfloor \frac{\sqrt{4\left(\lambda_1 + \mu_n\right)\lambda_n^{(-)} + \left(\tilde{y}_n^{(\pm)}\right)^2} - \left|\tilde{y}_n^{(\pm)}\right|}{2} \right\rfloor. \tag{7.4}$$

In practice, the mode is very close to the mean of the Bessel distribution. In fact, we can prove that the mode of a Bessel distribution is always an integer neighbor of the mean, i.e., either the floor or ceiling of the Bessel distribution:

Proposition 7.1. *(Proven in Appendix A.) The absolute difference between the mean and mode of the Bessel distribution is bounded by 1:*

$$\left|\mathbb{E}_{Bessel(m;\nu,a))}[m] - \text{mode}\left(Bessel\left(m; a, \nu\right)\right)\right| \leq 1. \tag{7.5}$$

This results in only a small negative bias in the expectation of m.

Hybrid noise inference (HYBRID**).** A final problem in sampling comes from the density of the noise parameter space. Typically, inference of Poisson factorization models benefits from sparse observation matrices; updates to parameters can be computed by iterating through only the nonzero entries in the data. However, in our method, we cannot assume that any observed zero (i.e., a count that is zero with private noise added) is actually a zero. As a result, we must store and compute samples densely for the noise proportional to the total size of the observed data. For data sets with large dimensions, inference over the dense noisy data observation matrix can be slow and space-intensive.

Some of this slowness may be remedied through simple parallelization. Each count in the data matrix has its private noise generated independently of the other counts in the matrix, so resampling of these variables may occur in any order or simultaneously without additional coordination. However, that scales only roughly linearly with the number of processors available. An option to somewhat reduce this load is to reduce noise inference only to entries whose observed value is above a certain threshold. Motivated by the concept of "heavy hitters" in networks, this strategy aims to focus the resources of reconstruction on the entries large enough that, even with noise added, we have some reliable

notion of their magnitude. In many applications, large magnitude entries are the ones that are most interesting to model correctly.

For entries less than or equal to a given threshold t, we perform inference naïvely. In Algorithm 7.2, this would manifest as a modification to line 2 to only include indices where $\tilde{y}_{\boldsymbol{i}}^{(\pm)}$ is greater than a designated threshold. In our experiments below, we set the threshold at $t = 0$, such that any count observed to be 0 or less after noise is treated as a "true" data observation by our model. We only infer $\lambda_n^{(s)}$ and $g_n^{(s)}$ for positive observed counts, with the possibility that inference may lead to estimates of those true counts as zero. As the choice of which noise variables to infer is conditioned only the observed data, we can save memory by initializing only noise variables for sufficiently large $\tilde{y}_{\boldsymbol{i}}^{(\pm)}$.

Figure 7.1 shows the relative performance improvements of each of these methods. We evaluate the mean relative error of the estimated matrix of Poisson priors with respect to the true observed data using 25 synthetic social network data matrices, each generated synthetically using the generative process for a Bayesian mixed-membership stochastic block model for overlapping community detection [11, 60, 177]. The model data are generated with 100 agents and 5 latent communities with a Gamma shape and rate prior of $a = 0.1$ and $b = 1$, respectively. We evaluate our inference code with parallelization disabled to obtain a clearer sense of the timing. We find that applying the SCHEDULE approach converge at both a clock time and iteration rate indistinguishable from the naïve model performing no inference of noise. Though the ICM and HYBRID strategies sometimes take longer to converge after reaching a local optimum, the convergence behavior is not affected by composing with the SCHEDULE approach. We therefore suggest that default implementations use the SCHEDULE approach,

Algorithm 7.5: MCMC for private variables including both the HYBRID (lines 3-6) and ICM (line 7) approximations.

1: **function** APPROXPRIVATEINFERENCE($\tilde{\boldsymbol{Y}}^{(\pm)}$ α, a, b, t)
2: **for all** indices $\boldsymbol{i}$ in $\tilde{\boldsymbol{Y}}^{(\pm)}$ **do**
3: **if** $\tilde{y}_{\boldsymbol{i}}^{(\pm)} < t$ **then** ▷ HYBRID approximation uses naïve inference for observed entries $\tilde{y}_{\boldsymbol{i}}^{(\pm)}$ less than a threshold t
4: $y_{\boldsymbol{i}} = \max(\tilde{y}_{\boldsymbol{i}}^{(\pm)}, 0)$;
5: **continue**;
6: **end if**
7: $m_{\boldsymbol{i}} = \left\lfloor \frac{\sqrt{4(\lambda_{\boldsymbol{i}}^{(+)}+\mu_{\boldsymbol{i}})\lambda_{\boldsymbol{i}}^{(-)}+(\tilde{y}_{\boldsymbol{i}}^{(\pm)})^2}-|\tilde{y}_{\boldsymbol{i}}^{(\pm)}|}{2} \right\rfloor$; ▷ ICM uses the mode of the Bessel instead of sampling from the Bessel distribution
8: **if** $m_{\boldsymbol{i}} < 0$ **then**
9: $\tilde{y}_{\boldsymbol{i}}^{(+)} = m_{\boldsymbol{i}}$;
10: $g_{\boldsymbol{i}}^{(-)} = \tilde{y}_{\boldsymbol{i}}^{(+)} - \tilde{y}_{\boldsymbol{i}}^{(\pm)}$;
11: **else**
12: $g_{\boldsymbol{i}}^{(-)} = m_{\boldsymbol{i}}$;
13: $\tilde{y}_{\boldsymbol{i}}^{(+)} = \tilde{y}_{\boldsymbol{i}}^{(\pm)} + g_{\boldsymbol{i}}^{(-)}$;
14: **end if**
15:
16: $y_{\boldsymbol{i}} \sim \text{Binom}\left(\tilde{y}_{\boldsymbol{i}}^{(+)}, \frac{\mu_{\boldsymbol{i}}}{\mu_{\boldsymbol{i}}+\lambda_{\boldsymbol{i}}^{(+)}}\right)$;
17: $g_{\boldsymbol{i}}^{(+)} = \tilde{y}_{\boldsymbol{i}}^{(+)} - y_{\boldsymbol{i}}$;
18:
19: $\lambda_{\boldsymbol{i}}^{(+)} \sim \text{Gamma}\left(g_{\boldsymbol{i}}^{(+)} + 1, \alpha\right)$;
20: $\lambda_{\boldsymbol{i}}^{(-)} \sim \text{Gamma}\left(g_{\boldsymbol{i}}^{(-)} + 1, \alpha\right)$;
21: **end for**
22: **end function**

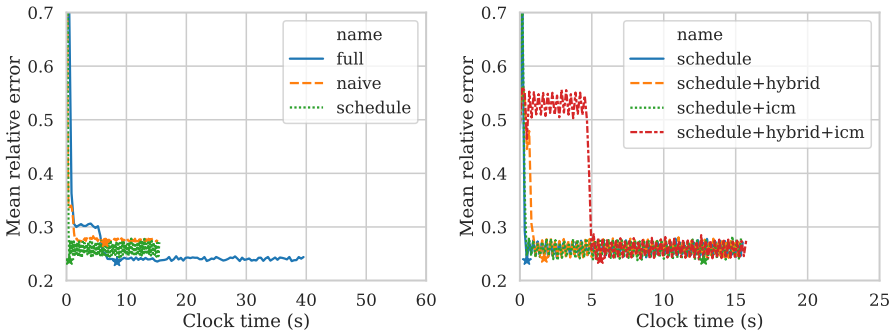


Figure 7.1: Elapsed time in seconds of wall clock time versus relative error of the Poisson parameters during inference, demonstrating the performance improvements offered by the provided approaches. Samples are plotted every 50 iterations. This leads to apparent oscillation on schedule-based algorithms where the privacy parameters update only every 100 iterations.

with the possibility of applying ICM for large input matrices or HYBRID for very sparse input data if further performance improvements are necessary.

We can consider a variety of evaluation metrics to validate these methods as producing models of similar quality to the full inference approach. Classically, when considering how well a statistical is fit to held-out data, we would test using held-out likelihood or similar, as described in Section 2.3. In this case, we can even consider using the model fit on the noise-free version of the training data. However, training log likelihood and perplexity suffer a significant problem in evaluating a private method that targets the restoration of sparsity to a model. Because these metrics are computed via summed *log* probabilities $\log p(w_{di}|\cdot)$ of individual tokens, which produce large negative numbers for smaller numbers, a small fraction of unlikely words in a document can significantly detract in the resulting evaluation. Models that fail to learn useful coherent latent structure, e.g. by assigning nontrivial probability to a large portion of the vocabulary, will not suffer these kinds of penalties even if the likelihoods are somewhat worse for common tokens. In a setting where we wish to incentivize privacy,

this is particularly unsettling, as the most consequential parts of the likelihood evaluation will be rarer tokens that should be private.

We therefore evaluate fit instead with *mean absolute error* (MAE) of the Poisson priors for the model with respect to the true data counts. This places more emphasis on recovery of higher-order counts, and therefore better measures how well we recover the "big picture" latent structure of the true data. For example, if the true data $\boldsymbol{Y}$ for an LDA model has dimensions $D \times V$, we would use the following formulation to average over all entries of $\boldsymbol{Y}$:

$$MAE = \frac{\sum_{d=1}^{D} \sum w = 1^{V} |y_{dv} - \mu_{dv}|}{D \cdot V}. \tag{7.6}$$

Because the full LDA model is hard to test using the original method on large data, we initially evaluate our approximations on a smaller BPF model that instead infers a mixed-members stochastic block model for Enron email data [78]. For a network of V members, the data entry y_{ij} represents the number of messages sent from network member i to network member j. Supposing that C communities help describe these message behaviors, we wish to factorize this matrix of parameters $\mu = \theta_0 \theta_1 \theta_0^T$, where θ_0 is a $V \times C$ matrix of community memberships, and θ_1 is a $C \times C$ matrix of inter-community interactions. Figures 7.2 and 7.3 demonstrate the described effect: where the former strongly penalizes the private models for failing to fit rare occurrences of interactions in the true data, MAE places more emphasis on recovering a *sparse* structure to fit the data, which is a requirement for interpretable models.

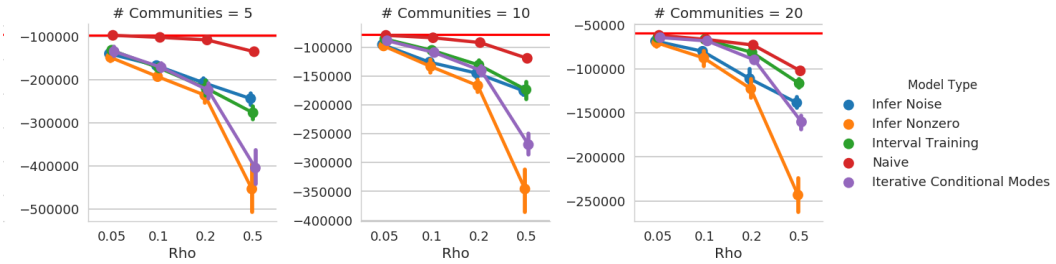


Figure 7.2: Log likelihood of mixed-membership stochastic block models inferred on Enron community structure with methods of improvement. In this case, we see that the log likelihood for the naïve model is quite high, due to the emphasis on representing rare events. The red horizontal line represents the value across 10 models inferred directly on nonprivate data.

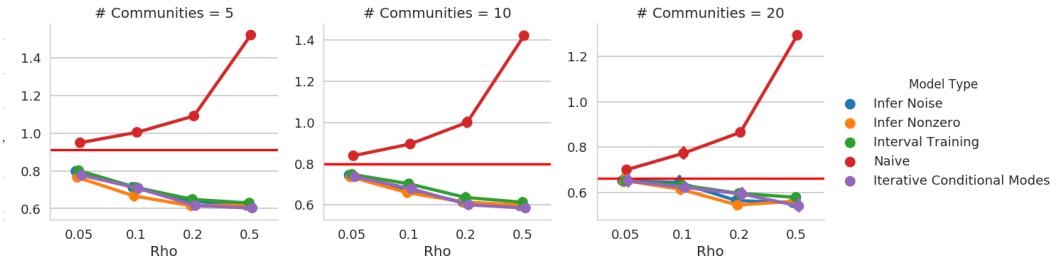
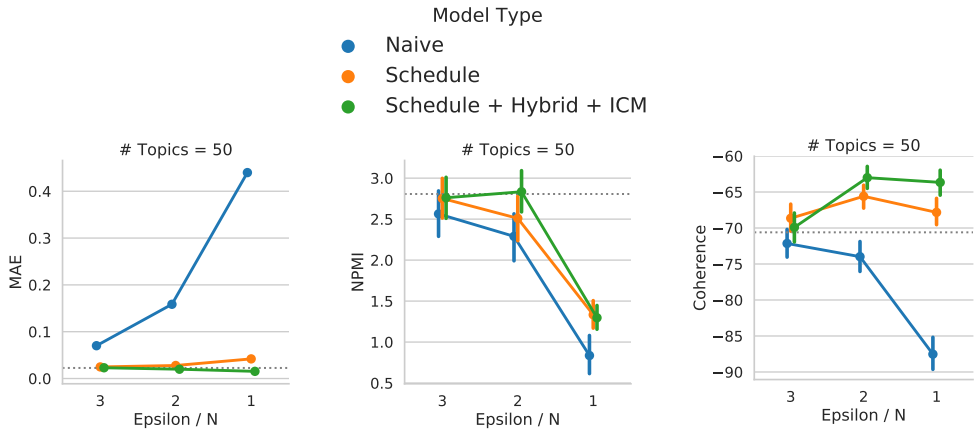


Figure 7.3: MAE of the same mixed-membership stochastic block models inferred on Enron community structure in Figure 7.2. In this case, error for the naïve method is high, reflecting too high a probability of rare tokens due to the added geometric noise.

7.3 Private Noise As Regularizer

Using our faster inference with the above approaches, we can evaluate performance on a full corpus of Enron email text [78]. To test, we infer models with 50 topics using naïve inference, the SCHEDULE approach above, and a mix of all three MCMC algorithm performance improvements. We consider a variety of standard evaluations as described in Section 2.3 of this thesis. We show results of these evaluations in Figure 7.4. Across our metrics, we see that our fast approaches produces results that not only improve over naïve inference, but also

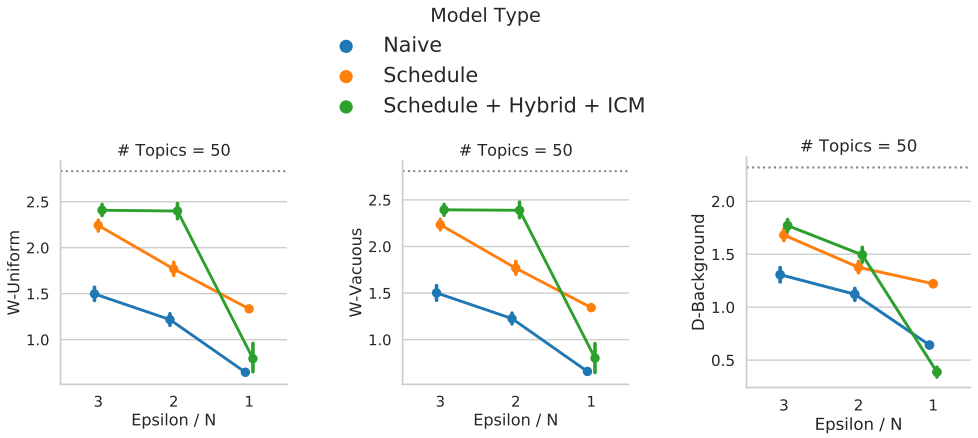


a: MAE b: NPMI c: Topic coherence

Figure 7.4: Evaluation metrics for topics learned with private inference using Enron emails as a dataset. The x axis is presented as the ratio of privacy budget ϵ to limited-precision bound N, with the magnitude of private noise increasing as ϵ/N decreases. Our inference method not only helps to recover a high quality sparse model, it also can improve over the nonprivate model inferred without noise (represented in grey dashes).

produce close to non-private results at $\epsilon/N = 2$. We can further observe in Figure 7.5 that this arises from LPBPF's stronger ability to resist inferring "junk" topic archetypes given by AlSumait et al. [5] and discussed in Section 2.3.

A significant surprise of this work is the fact that topic coherence appears to improve over nonprivate inference as private noise is added. The primary hypothesis for why this might occur relates to the observation that in traditional BPF, only Poisson processes generate words in text collections. In contrast, LPBPF allows for the explanation of some of the more surprising tokens in the true data to have their statistical generation explained by random noise. This allows the model to fit more parsimoniously to words with stronger co-occurrence signals, producing topics with higher coherence than even typical non-private models. The result suggests that some benefit may come from inferring BPF-style models using this or a related method with an additional generative process for



a: Uniform topic b: Vacuous topic c: Background topic

Figure 7.5: KL-divergence from the three junk topic distributions introduced by AlSumait et al. [5] to help understand how our inference method improves over naïve inference.

independent noise to produce more interpretable models.

7.4 Initialization with Variational Inference

While the performance improvements in the previous section may speed up MCMC inference, they do not remove the slow bottleneck of sampling from the Bessel distribution. Coordinate-ascent variational inference (CAVI) offers several advantages over MCMC for performance, as outlined in Section 2.2.2. It replaces expensive sampling procedures for latent variables with analytic updates of parameters describing those variables' distributions while also giving a clear objective, the ELBO, to track model convergence. CAVI methods therefore often run in far fewer iterations and more quickly than MCMC algorithms. However, these methods also rely on derivations of appropriate variational distributions to closely approximate the true distribution. This section outlines the derivational process used to infer a new CAVI algorithm for LPBPF inference.

Of particular interest is the variational distribution for $m_{\boldsymbol{i}}$. In the formulation of the generative process given in Equation 7.3, $m_{\boldsymbol{i}}$ is sampled from a Bessel distribution. The Bessel distributionBessel (ν, a) has two parameters, ν and a. In the case of our algorithm, for each index of $m_{\boldsymbol{i}}$, the corresponding Bessel distribution from which it is sampled has a known order value defined by the observed data $\tilde{y}_{\boldsymbol{i}}^{(\pm)}$. When ν is considered a constant, the Bessel distribution reduces to a single-parameter distribution that is in the exponential family.

Proposition 7.2. *The Bessel distribution Bessel $(k; \nu, a)$ for a fixed value of ν is of the exponential family, with sufficient statistic $T(k) = 2k + \nu$, natural parameter $\eta(a) = \log(\frac{a}{2})$, base measure $h(k) = \frac{1}{k!\Gamma(k+\nu+1)}$, and log partition $A(a) = \log I_\nu(a)$.*

Propositions 7.2 and is proven in Appendix B. This allows us to infer this fixed-ν Bessel as the optimal choice of candidate distribution for $m_{\boldsymbol{i}}$:

$$Q(m_{\boldsymbol{i}}) = \text{Bessel}\left(m_{\boldsymbol{i}};\ |\tilde{y}_{\boldsymbol{i}}^{(\pm)}|, 2\sqrt{\mathbb{G}_Q\left[\lambda_{\boldsymbol{i}}^{(-)}\right]\mathbb{G}_Q\left[\lambda_{\boldsymbol{i}}^{(+)} + \mu_{\boldsymbol{i}}\right]}\right). \tag{7.7}$$

Unfortunately, this derivation alone is not sufficient to construct a closed-form inference algorithm for VI. In particular, the derivation of the optimal variational distribution for $\tilde{\boldsymbol{y}}_{\boldsymbol{i}}^{(+)}$, or the individual component counts summed to $\tilde{y}_{\boldsymbol{i}}^{(+)}$, is found to be a multinomial with number of draws:

$$Q(\tilde{\boldsymbol{y}}_{\boldsymbol{i}}^{(+)}) = \text{Multinom}\left(\tilde{\boldsymbol{y}}_{\boldsymbol{i}}^{(+)};\ \mathbb{E}_Q\left[\tilde{y}_{\boldsymbol{i}}^{(+)}\right], \left(\mathbb{G}_Q\left[\lambda_{\boldsymbol{i}}^{(+)}\right], \mathbb{G}_Q\left[\mu_{\boldsymbol{i}1}\right], \ldots, \mathbb{G}_Q\left[\mu_{\boldsymbol{i}K}\right]\right)\right). \tag{7.8}$$

The optimal choice of Q-distributions in Equations 7.7 and 7.8 are incompatible: the expectation of $m_{\boldsymbol{i}}$ is not necessarily an integer, but the number of draws

in a multinomial distribution, computed deterministically from $\mathbb{E}_Q[m_{\boldsymbol{i}}]$ and integer $\tilde{y}_{\boldsymbol{i}}^{(+)}$ must be an integer. To resolve this, we reuse Proposition 7.1 to select the mode of the Bessel as a suitable, close alternative to the mean, and replace $\mathbb{E}_Q\left[\tilde{y}_{\boldsymbol{i}}^{(+)}\right]$ with mode$(\tilde{y}_{\boldsymbol{i}}^{(+)})$. Though it is not necessary for the Q-distributions to be optimal for the algorithm to converge, it will converge much more efficiently with choices of Q-distribution well-matched to the true model.

In order to test this result, we generated synthetic count data using the LDA structure of Poisson factorization model with D documents, V unique terms in a document vocabulary, and K latent topics. We first used a Gamma prior to generate two matrices of latent parameters, θ of dimension $D \times K$ and ϕ of dimension $K \times V$. We then compute the product of these, $\theta\phi = \boldsymbol{Y}$, as the Poisson prior of our data generation process. Finally, we add two-sided geometric noise scaled to $\epsilon/N = 1$, a ratio that applies when the privacy budget ϵ and the maximum allowed difference between documents N are equal (e.g., a privacy budget of 2 to privatize 2-word spans). We then test our inference procedure to see how closely it estimates the true parameters of the original model. We find that our model successfully converges to within a reasonable estimate of the true model parameters given the data, as shown in a small example in Figure 7.6 in a random block structure has been imposed on the parameter matrices.

Using a larger example of a synthetic 1000-by-1000 matrix of count observations, we test the compare this algorithm with MCMC. In a single-threaded version of the MCMC implementation with the SCHEDULE method, each iteration of inference takes approximately 0.8 seconds. Using 4000 iterations of burnin and 1000 to collect samples with 32 cores, we are able to take the combined inference time down to 5 minutes (using 160 minutes of observed CPU time). In contrast,

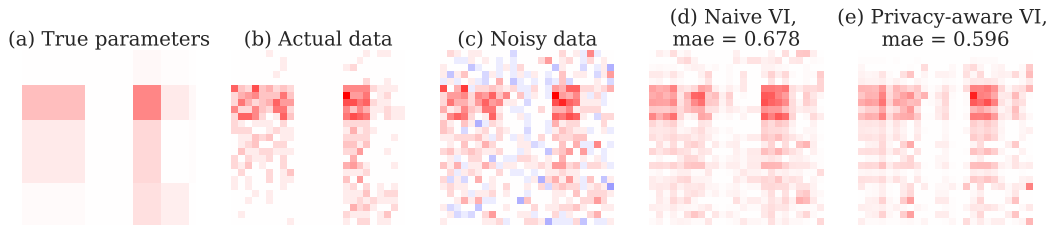


Figure 7.6: Demonstration of the results of the VI inference process for a 25-word, 25-document synthetic dataset with 5 latent topics and Gamma prior parameters of shape 0.5 and rate 1. For sparse, structured data with dense regions, true data parameters (a) are recovered better by our inference procedure (e) than by naïve variational inference (d), even though the noisy data (c) is much denser than the true data (b).

Method	Iterations	MAE
Naïve MCMC	5000 burnin + 1000	0.396
Private MCMC	3000 burnin + 1000	0.208
Private MCMC	2000 burnin + 1000	0.224
Naïve VI	42	0.492
Naïve VI	100	0.465
Private VI	42 (converged)	0.396
Private MCMC, VI init	1000 burnin + 1000	0.223
Private MCMC, VI init	2000 burnin + 1000	0.208

Table 7.1: MAE over estimates of the Poisson model parameters inferred over the same 1000-by-1000 matrix with 50 latent components and noise scaled to $\epsilon/N = 1$. MCMC results used the algorithm from [141] were averaged over 10 samples taken 100 iterations apart starting immediately after the last burnin iteration.

our variational inference model takes approximately 2.8 seconds per iteration, and requires only 41-42 iterations to converge, a combined 1 minute of inference (16 minutes observed CPU time). Depending on parallelization, this method offers a 5-10x speedup over the fast inference method. This improvement should increase significantly as the size of the observed data increases. The MAE result of these synthetic experiments are shown in Table 7.1.

To demonstrate that this model can learn interesting patterns on larger scale

Nonprivate	1	000 year billion dollars budget million fiscal tax program expenditures
	2	people education jobs american life workers work better million good
40 iterations,	3	new children energy environment america help schools school technology science
MAE = 0.0897	4	health care work security insurance people make social welfare americans
Naïve VI	1	year billion dollars federal budget programs fiscal government million program
	2	work people americans children make help let america need know
52 iterations,	3	congress act energy states national program legislation economic america law
MAE = 0.0157	4	health care new congress year insurance medical americans legislation government
Private VI	1	000 year million fiscal billion dollars years budget increase expenditures
	2	know work children welfare people year reform new believe america
100 iterations,	3	government congress energy federal national act states new policy program
MAE = 0.0157	4	health care government federal public security insurance congress new make

Table 7.2: Sample topics from 20-topic models inferred using our VI inference methods with privacy set to $\epsilon/N = 5$. MAE measured against the nonprivate data is lower for our method when we overestimate privacy than in a nonprivate model.

data, we also consider topic output for a topic model. We display topic model results in Table 7.2 using a dataset of paragraphs from all United States State of the Union addresses through 2009, with standard English stopwords removed. We find that our model does produce results worse than naïve MCMC inference. However, the inferred VI model is an effective initialization for MCMC, potentially saving time by reducing expensive sampling in MCMC inference. Further, these models appear to experience regularization benefits, as they demonstrate lower MAE through inference than even nonprivate inference.

7.5 Discussion

This chapter demonstrates a number of different strategies for improving the speed of inference of locally private Bayesian Poisson factorization (LPBPF). Using both modifications to the original MCMC algorithm and a new VI algorithm, we are able to find ways to push inference to take little more time than private inference. Furthermore, we are able to draw new insights on how privacy-aware

inference in LPBPF affects the inferred models, including their tendency towards producing more coherent topics in the LDA setting than nonprivate models.

Though this work is exciting, it still relies on a simple additive noise mechanism through two-sided geometric noise. A weakness of this work is that it fails to address some of the fundamental problems of local privacy as described in Chapter 6 for text data, namely the curse of dimensionality, that make it challenging to navigate the tradeoff of privacy and model effectiveness. Using the approaches of compression from Figure 6 as additional matrices in a more complex factorization scheme could provide a possible approach to adding noise to a smaller set of observed coordinates. Additional work in progress also looks at how to use models like this and other private language models to generate synthetic documents. This could be helpful in industry settings where true documents are unavailable for testing machine learning models, but an approximation of the true results would be useful to recover in testing.

CHAPTER 8

CONCLUSIONS

In this dissertation, I present a number of pieces of work that evaluate how LDA models and their private counterpart, LPBPF, respond to changes in data processing. This includes work on understanding of stemming [144], stopword removal [143], text duplication [146], and the introduction of differential privacy [145]. Though these projects all center around data processing for one core model, a key contribution of this work is methods of building understanding that can generalize to latent variable models with collocations and more complex assumptions of statistical structure behind text generation.

The evaluation metrics from Part I provide a roadmap for practitioners to answer practical questions about the effects of pre-processing on their own data. However, the description of these methodologies is not a replacement for an integrated tool that helps implement these evaluations. In my A exam proposal, I discussed a user interface that implements a post-processing approach. This work would tap into a rich literature of topic model visualization [4, 25, 31, 51, 158] and interactive topic models [71, 96] that supplies intuitions into how practitioners use topic models in their work. After several independent projects with undergraduate researchers and discussion with collaborators outside of computer science, I am excited to restart this work with new insights into what tools and workflows should be best supported. Ideally, this work would take inspiration from tools like wordvectors.org, which provide online methods of fast evaluation for newly inferred word embeddings in order to benchmark the success of a particular pre-processing and training approach.

A limitation of the work presented here is that it often relies on standard

benchmark datasets, such as 20 Newsgroups [131], Enron [78], New York Times articles [137], Reuters newswires, [86], and so on. These datasets are often represented as the analogues of MNIST, CIFAR-10, and ImageNet in computer vision, as baseline datasets for strong comparative evaluation on basic tasks. Like these datasets, canonical English test collections often still have processing and labeling issues; beyond this, however, these datasets reflect specific conditions of language generation with biases we often avoid discussing. For example, a significant portion of 20 Newsgroups' talk.politics.guns discussion from the collected period is about the concurrent Waco siege of 1993, a feature that becomes prominent and unsettling with reading. Enron emails contain detailed personal messages, and the Reuters Spanish Language dataset contains articles in English. Taken together, even benchmark text collections for bag-of-words methods can be problematic sources from which to generalize, and it is for this reason that I emphasize the methodology, not the recommendations, as the most impactful part of this work. Reproducing these results with non-Indo-European languages with varied morphologies, across new forms of social media and old texts with messy OCR, will be the true test of these conclusions. The only effective way to do this validation is in consultation with experts in these types of data and communication from social science, journalism, literature, and any other areas that focus on large text collections. Again, this stresses the need for interactive tools so experts outside computer science may perform these tests.

The privacy work of this paper is purposefully vague about some of the consequences of a limited-precision local approach to privacy. In differential privacy as a whole, the question of how to choose an appropriate privacy budget, ϵ, shares subjectivity with the choice of pre-processing treatments in natural language processing: experts have opinions about good and bad choices, but

there is no justification for a consensus on how to choose ϵ. To move private models like LPBPF into the mainstream requires addressing these questions empirically for the text domain for not only bag-of-words data, but also n-gram data and more complex language models. Inference for LPBPF in particular should be tested with strong case studies of private text data to monitor what levels of privacy concretely obscure facts one might want to recover. My hope is that datasets like the CFPB Consumer Reports [24] will provide an opportunity to perform semi-synthetic experiments to better validate meaningful privacy thresholds and their consequences for model inference.

Meaningful privacy work also requires addressing actual problems for which there is demand for privacy-preserving machine learning. For instance, a current collaboration focuses on ways to generate synthetic syntactic text data that preserves language co-occurrence patterns, such that researchers looking to apply machine learning methods on private user text data may use this synthetic data to test their code and validate broad results. Recent work from OpenAI on language generation produces a hopeful sign for the future of language generation models, but effective deployment of such models will require guarantees of not reproducing training data in generation, which privacy can provide.

Finally, a criticism of some of this work is its focus on a single model, LDA. In fact, as a first year PhD student attending my first Neural Information Processing Systems conference, a senior researcher on hearing that I worked with LDA advised me against doing so. His argument was that LDA was no longer a popular model in the community (never mind its broad use in industry and social science, nor its thousands of citations annually). I see the importance of this argument: it is critical to study new, powerful methods of machine learning

in order to reach new levels of potential understanding of both models and human behavior. However, my belief is that the pursuit of this understanding should not be done at the cost of abandoning methods that are powerful but not fully understood. The chief lesson I have taken from these projects is that we have much work left to do to ensure unsupervised statistical models like LDA and LPBPF are efficient, interpretable, and effective at answering human questions about the text we produce. It is my hope that we can combine the creation of new machine learning methods, whether statistical, neural, or otherwise, with rigorous theoretical and empirical work on our existing tools to build our understanding of what it is our machines learn.

BIBLIOGRAPHY

[1] C. Aggarwal and P. Yu. On Privacy-Preservation of Text and Sparse Binary Data with Sketches. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, Proceedings, pages 57–67. Society for Industrial and Applied Mathematics, 2007. DOI: 10.1137/1.9781611972771.6.

[2] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9(Sep):1981–2014, 2008.

[3] Nikolaos Aletras and Mark Stevenson. Evaluating topic coherence using distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Long Papers*, pages 13–22. Association for Computational Linguistics, 2013.

[4] Eric Alexander, Joe Kohlmann, Robin Valenza, Michael Witmore, and Michael Gleicher. Serendip: Topic model-driven visual exploration of text corpora. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 173–182. IEEE, 2014.

[5] Loulwah AlSumait, Daniel Barbará, James Gentle, and Carlotta Domeniconi. Topic significance ranking of LDA generative models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 67–82. Springer, 2009.

[6] Anima Anandkumar, Dean P Foster, Daniel J Hsu, Sham M Kakade, and Yi-Kai Liu. A spectral algorithm for latent Dirichlet allocation. In *Advances in Neural Information Processing Systems*, pages 917–925, 2012.

[7] Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, pages 901–914. ACM, 2013.

[8] Sanjeev Arora, Rong Ge, Yonatan Halpern, David Mimno, Ankur Moitra, David Sontag, Yichen Wu, and Michael Zhu. A practical algorithm for topic modeling with provable guarantees. In *Proceedings of the 30th International Conference on Machine Learning*, pages 280–288, 2013.

[9] Sanjeev Arora, Rong Ge, and Ankur Moitra. Learning topic models–going beyond svd. In *Proceedings of the 53rd IEEE Annual Symposium on Foundations of Computer Science*, pages 1–10. IEEE, 2012.

[10] Rajkumar Arun, Venkatasubramaniyan Suresh, CE Veni Madhavan, and MN Narasimha Murthy. On finding the natural number of topics with latent Dirichlet allocation: Some observations. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 391–402. Springer, 2010.

[11] Brian Ball, Brian Karrer, and Mark E. J. Newman. Efficient and principled method for detecting communities in networks. *Physical Review E*, 84(3):036103, 2011.

[12] Daniel Bär, Torsten Zesch, and Irnya Gurevych. Text reuse detection using a composition of text similarity measures. In *Proceedings of the 24th International Conference on Computational Linguistics*, volume 1, pages 167–184, 2012.

[13] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, 48:259–302, 1986.

[14] Narayan L Bhamidipati and Sankar K Pal. Stemming via distribution-based word segregation for classification and retrieval. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 37(2):350–360, 2007.

[15] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, 2009. Available at: http://www.nltk.org/book/.

[16] Jonathan Bischof and Edoardo M Airoldi. Summarizing topical content with word frequency and exclusivity. In *Proceedings of the 29th International Conference on Machine Learning*, pages 201–208, 2012.

[17] David M Blei and John D Lafferty. Dynamic topic models. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 113–120. ACM, 2006.

[18] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003.

[19] Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. The Johnson-Lindenstrauss transform itself preserves differential privacy. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium*, pages 410–419. IEEE, 2012.

[20] Gerlof Bouma. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, pages 31–40, 2009.

[21] Jordan Boyd-Graber, David Mimno, David Newman, Edoardo M Airoldi, David Blei, and Elena A Erosheva. Care and feeding of topic models: Problems, diagnostics, and improvements. *Handbook of mixed membership models and their applications*, pages 3–34, 2014.

[22] Hai Brenner and Kobbi Nissim. Impossibility of differentially private universally optimal mechanisms. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 71–80. IEEE, 2010.

[23] Andrei Z Broder, Steven C Glassman, Mark S Manasse, and Geoffrey Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8-13):1157–1166, 1997.

[24] CFPB. Consumer complaint database. *ConsumerFinance.gov*, 2018.

[25] Allison June-Barlow Chaney and David M Blei. Visualizing topic models. In *Proceeedings of the Sixth International AAAI Conference on Weblogs and Social Media*, 2012.

[26] Edward Y Chang, Hongjie Bai, and Kaihua Zhu. Parallel algorithms for mining large-scale rich-media data. In *Proceedings of the 17th ACM International Conference on Multimedia*, pages 917–918. ACM, 2009.

[27] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L. Boyd-Graber, and David M. Blei. Reading tea leaves: How humans interpret topic models. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, pages 288–296. Curran Associates, Inc., 2009.

[28] Matt Chaput. Stemming library, 2010. Available at: https://bitbucket.org/mchaput/stemming.

[29] Chaitanya Chemudugunta, Padhraic Smyth, and Mark Steyvers. Modeling general and specific aspects of documents with a probabilistic topic model. In *Advances in Neural Information Processing Systems*, pages 241–248, 2007.

[30] Xiang Cheng, Sen Su, Shengzhi Xu, Peng Tang, and Zhengyi Li. Differen-

tially private maximal frequent sequence mining. *Computers & Security*, 55(Supplement C):175–192, 2015.

[31] Jason Chuang, Christopher D Manning, and Jeffrey Heer. Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the Internation Working Conference on Advanced Visual Interfaces*, pages 74–77. ACM, 2012.

[32] Daniel T Citron and Paul Ginsparg. Patterns of text reuse in a scientific corpus. *Proceedings of the National Academy of Sciences*, 112(1):25–30, 2015.

[33] Paul Clough et al. Old and new challenges in automatic plagiarism detection. In *National Plagiarism Advisory Service*, 2003. http://ir.shef.ac.uk/cloughie/index.html.

[34] Paul Clough, Robert Gaizauskas, Scott SL Piao, and Yorick Wilks. Meter: Measuring text reuse. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 152–159, 2002.

[35] Raphael Cohen, Iddo Aviram, Michael Elhadad, and Noémie Elhadad. Redundancy-aware topic modeling for patient record notes. *PLOS ONE*, 9(2):e87555, 2014.

[36] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167. ACM, 2008.

[37] Fida Kamal Dankar and Khaled El Emam. The application of differential privacy to health data. In *Proceedings of the 2012 Joint EDBT/ICDT Workshops*, EDBT-ICDT '12, pages 158–166, New York, NY, USA, 2012. ACM.

[38] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[39] Luc Devroye et al. Simulating bessel random variables. *Statistics & Probability Letters*, 57(3):249–257, 2002.

[40] Mark Dredze, Hanna M. Wallach, Danny Puller, and Fernando Pereira. Generating summary keywords for emails using topics. In *Proceedings of*

the 13th International Conference on Intelligent User Interfaces, IUI '08, pages 199–206. ACM, 2008.

[41] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Theory of Cryptography Conference*, volume 3876, pages 265–284, 2006.

[42] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[43] Jacob Eisenstein, Amr Ahmed, and Eric P Xing. Sparse additive generative models of text. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1041–1048. Omnipress, 2011.

[44] Michael Elhadad, Mehi Adler, Yoav Goldberg, and Rafi Cohen. Topic models for morphologically rich languages. In *Abstract presented at Workshop on Machine Translation and Morphologically-Rich Languages*. Israel Science Foundation, 2011.

[45] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. RAPPOR: randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pages 1054–1067, 2014.

[46] Mark D Flood, Jonathan Katz, Stephen J Ong, and Adam Smith. Cryptography and the economics of supervisory information: Balancing transparency and confidentiality. 2013.

[47] Antske Fokkens, Marieke Van Erp, Marten Postma, Ted Pedersen, Piek Vossen, and Nuno Freire. Offspring from reproduction problems: What replication failure teaches us. In *Proceedings of the 51st ACL*, pages 1691–1701, 2013.

[48] Ryan J Gallagher, Kyle Reing, David Kale, and Greg Ver Steeg. Anchored correlation explanation: Topic modeling with minimal domain knowledge. *Transactions of the Association for Computational Linguistics*, 5:529–542, 2017.

[49] Matthias Gallé and Matías Tealdi. Reconstructing textual documents from n-grams. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 329–338. ACM, 2015.

[50] Kuzman Ganchev and Mark Dredze. Small statistical models by random feature mixing. In *Proceedings of the ACL 2008 HLT Workshop on Mobile Language Processing*, pages 19–20, 2008.

[51] Matthew J Gardner, Joshua Lutes, Jeff Lund, Josh Hansen, Dan Walker, Eric Ringger, and Kevin Seppi. The topic browser: An interactive tool for browsing topic models. In *NIPS Workshop on Challenges of Data Visualization*, volume 2. Whistler Canada, 2010.

[52] Quan Geng and Pramod Viswanath. The optimal mechanism in differential privacy. In *2014 IEEE International Symposium on Information Theory*, pages 2371–2375. IEEE, 2014.

[53] Sean Gerrish and David M Blei. How they vote: Issue-adjusted models of legislative behavior. In *Advances in Neural Information Processing Systems*, pages 2753–2761, 2012.

[54] Joseph Geumlek and Kamalika Chaudhuri. Profile-based privacy for locally private computations. *arXiv preprint arXiv:1903.09084*, 2019.

[55] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. Universally utility-maximizing privacy mechanisms. *SIAM Journal on Computing*, 41(6):1673–1693, 2012.

[56] Ryan J Giordano, Tamara Broderick, and Michael I Jordan. Linear response methods for accurate covariance estimates from mean field variational bayes. In *Advances in Neural Information Processing Systems*, pages 1441–1449, 2015.

[57] Aris Gkoulalas-Divanis, Grigorios Loukides, and Jimeng Sun. Publishing data from electronic health records while preserving privacy: A survey of algorithms. *Journal of Biomedical Informatics*, 50(Supplement C):4–19, 2014.

[58] Andrew Goldstone and Ted Underwood. What can topic models of pmla teach us about the history of literary scholarship. *Journal of Digital Humanities*, 2(1):39–48, 2012.

[59] Prem Gopalan, Jake M Hofman, and David M Blei. Scalable recommendation with Poisson factorization. *arXiv preprint arXiv:1311.1704*, 2013.

[60] Prem K. Gopalan and David M. Blei. Efficient discovery of overlapping

communities in massive networks. *Proceedings of the National Academy of Sciences*, 110(36):14534–14539, 2013.

[61] Gustavo Graff, David amd Gallegos. Spanish news text. *Linguistic Data Consortium*, DVD: LDC95T9, 1995.

[62] Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *PNAS*, 101(suppl. 1):5228–5235, 2004.

[63] Thomas L Griffiths, Mark Steyvers, David M Blei, and Joshua B Tenenbaum. Integrating topics and syntax. In *Advances in Neural Information Processing Systems*, pages 537–544, 2005.

[64] Pu Han, Si Shen, Dongbo Wang, and Yanyun Liu. The influence of word normalization in english document clustering. In *Proceedings of the IEEE International Conference on Computer Science and Automation Engineering*, volume 2, pages 116–120. IEEE, 2012.

[65] Donna Harman. How effective is suffixing? *Journal of the American Society for Information Science*, 42(1):7–15, 1991.

[66] Xi He, Ashwin Machanavajjhala, and Bolin Ding. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, pages 1447–1458. ACM, 2014.

[67] Qirong Ho, James Cipar, Henggang Cui, Seunghak Lee, Jin Kyu Kim, Phillip B Gibbons, Garth A Gibson, Greg Ganger, and Eric P Xing. More effective distributed ml via a stale synchronous parallel parameter server. In *Advances in Neural Information Processing Systems*, pages 1223–1231, 2013.

[68] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[69] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57. ACM, 1999.

[70] Y. Hong, J. Vaidya, H. Lu, P. Karras, and S. Goel. Collaborative search log sanitization: Toward differential privacy and boosted utility. *IEEE Transactions on Dependable and Secure Computing*, 12(5):504–518, 2015.

[71] Yuening Hu, Jordan Boyd-Graber, Brianna Satinoff, and Alison Smith. Interactive topic modeling. *Machine Learning*, 95(3):423–469, 2014.

[72] Tomoharu Iwata, Takeshi Yamada, and Naonori Ueda. Probabilistic latent semantic visualization: topic model for visualizing documents. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 363–371. ACM, 2008.

[73] Carina Jacobi, Wouter van Atteveldt, and Kasper Welbers. Quantitative analysis of large amounts of journalistic texts using topic modelling. *Digital Journalism*, 4(1):89–106, 2016.

[74] Jagadeesh Jagarlamudi, Hal Daumé III, and Raghavendra Udupa. Incorporating lexical priors into topic models. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 204–213. Association for Computational Linguistics, 2012.

[75] Anjali Ganesh Jivani. A comparative study of stemming algorithms. *International Journal of Computer Technology and Applications*, 2(6):1930–1938, 2011.

[76] Matthew L Jockers and David Mimno. Significant themes in 19th-century literature. *Poetics*, 41(6):750–769, 2013.

[77] Sowmya Kamath, Atif Ahmed, and Mani Shankar. A composite classification model for web services based on semantic & syntactic information integration. In *Advance Computing Conference, 2015 IEEE International*, pages 1169–1173. IEEE, 2015.

[78] Bryan Klimt and Yiming Yang. The Enron corpus: A new dataset for email classification research. In *European Conference on Machine Learning*, pages 217–226. Springer, 2004.

[79] Robert Krovetz. Viewing morphology as an inference process. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 191–202. ACM, 1993.

[80] Andrea Lancichinetti, M Irmak Sirer, Jane X Wang, Daniel Acuna, Konrad Körding, and Luís A Nunes Amaral. High-reproducibility and high-accuracy method for automated topic classification. *Physical Review X*, 5(1):011007, 2015.

[81] Ken Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339, 1995.

[82] John Langford, Lihong Li, and Alex Strehl. Vowpal wabbit online learning project, 2007.

[83] Jey Han Lau, David Newman, and Timothy Baldwin. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 530–539. Association for Computational Linguistics, 2014.

[84] John Lee. A computational model of text reuse in ancient literary texts. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 472–479, 2007.

[85] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems*, pages 2177–2185, 2014.

[86] David Lewis. Reuters-21578 text categorization test collection. *Distribution 1.0, AT&T Labs-Research*, 1997.

[87] Aaron Q Li, Amr Ahmed, Sujith Ravi, and Alexander J Smola. Reducing the sampling complexity of topic models. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 891–900. ACM, 2014.

[88] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pages 583–598, 2014.

[89] Wei Li and Andrew McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 577–584. ACM, 2006.

[90] Yang D Li, Zhenjie Zhang, Marianne Winslett, and Yin Yang. Compressive mechanism: utilizing sparse representation in differential privacy. In *Proceedings of the 10th ACM Workshop on Privacy in the Electronic Society*, pages 177–182. ACM, 2011.

[91] Kun Liu, H. Kargupta, and J. Ryan. Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):92–106, 2006.

[92] Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 366–376. Association for Computational Linguistics, 2010.

[93] Siaw Ling Lo, David Cornforth, and Raymond Chiong. Effects of training datasets on both the extreme learning machine and support vector machine for target audience identification on twitter. In *Proceedings of ELM-2014 Volume 1*, pages 417–434. Springer, 2015.

[94] Julie B Lovins. Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11:22–31, 1968.

[95] Leena Lulu, Boumediene Belkhouche, and Saad Harous. Overview of fingerprinting methods for local text reuse detection. In *2016 12th International Conference on Innovations in Information Technology*, pages 1–6. IEEE, 2016.

[96] Jeffrey Lund, Connor Cook, Kevin Seppi, and Jordan Boyd-Graber. Tandem anchoring: A multiword anchor approach for interactive topic modeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 896–905, 2017.

[97] Rasmus E Madsen, David Kauchak, and Charles Elkan. Modeling word burstiness using the Dirichlet distribution. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 545–552. ACM, 2005.

[98] Gurmeet Singh Manku, Arvind Jain, and Anish Das Sarma. Detecting near-duplicates for web crawling. In *Proceedings of the 16th International Conference on the World Wide Web*, pages 141–150. ACM, 2007.

[99] Chandler May, Ryan Cotterell, and Benjamin Van Durme. Analysis of morphology in topic modeling. *arXiv preprint arXiv:1608.03995*, 2016.

[100] Andrew K. McCallum. MALLET: a machine learning for language toolkit. Available at: http://mallet.cs.umass.edu, 2002.

[101] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learn-

ing differentially private language models. In *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[102] Rishabh Mehrotra, Scott Sanner, Wray Buntine, and Lexing Xie. Improving LDA topic models for microblogs via tweet pooling and automatic labeling. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 889–892. ACM, 2013.

[103] Marina Meilă. Comparing clusterings—an information based distance. *Journal of multivariate analysis*, 98(5):873–895, 2007.

[104] José Ramon Méndez, Eva Lorenzo Iglesias, Florentino Fdez-Riverola, Fernando Díaz, and Juan M Corchado. Tokenising, stemming and stopword removal on anti-spam filtering domain. In *Conference of the Spanish Association for Artificial Intelligence*, pages 449–458. Springer, 2005.

[105] Yishu Miao, Edward Grefenstette, and Phil Blunsom. Discovering discrete latent topics with neural variational inference. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2410–2419, 2017.

[106] Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K Gray, Joseph P Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, et al. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182, 2011.

[107] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[108] David Mimno. Reconstructing Pompeian households. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, pages 506–513. AUAI Press, 2011.

[109] David Mimno. Computational historiography: Data mining in a century of classics journals. *Journal on Computing and Cultural Heritage*, 5(1):3, 2012.

[110] David Mimno and David Blei. Bayesian checking for topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 227–237. Association for Computational Linguistics, 2011.

[111] David Mimno, Hanna Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. Optimizing semantic coherence in topic models. In

Proceedings of the Conference on Empirical Methods in Natural Language Processing, pages 262–272. Association for Computational Linguistics, 2011-07.

[112] Yuhong Nan, Min Yang, Zhemin Yang, Shunfan Zhou, Guofei Gu, and XiaoFeng Wang. UIpicker: User-input privacy identification in mobile applications. In *Proceedings of the 24th USENIX Security Symposium*, pages 993–1008, 2015.

[113] David Newman, Arthur Asuncion, Padhraic Smyth, and Max Welling. Distributed algorithms for topic models. *Journal of Machine Learning Research*, 10(Aug):1801–1828, 2009.

[114] David J Newman and Sharon Block. Probabilistic topic decomposition of an eighteenth-century american newspaper. *Journal of the American Society for Information Science and Technology*, 57(6):753–767, 2006.

[115] Brendan O'Connor, Michel Krieger, and David Ahn. Tweetmotif: Exploratory search and topic summarization for Twitter. In *Fourth International AAAI Conference on Weblogs and Social Media*, 2010.

[116] Chris D Paice. Another stemmer. *ACM SIGIR Forum*, 24(3):56–61, 1990.

[117] Leysia Palen and Paul Dourish. Unpacking privacy for a networked world. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 129–136. ACM, 2003.

[118] HweeHwa Pang, Xuhua Ding, and Xiaokui Xiao. Embellishing text search queries to protect user privacy. *Proc. VLDB Endow.*, 3(1-2):598–607, 2010.

[119] Mijung Park, James R Foulds, Kamalika Chaudhuri, and Max Welling. Private topic modeling. In *Proceedings of the NeurIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[120] Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword corpus. *Linguistic Data Consortium*, 2011.

[121] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[122] Martin F Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[123] Martin F Porter. Snowball: A language for stemming algorithms, 2001. Available at: http://www.snowball.tartarus.org/texts/introduction.html.

[124] Martin Potthast and Benno Stein. New issues in near-duplicate detection. In *Data Analysis, Machine Learning and Applications*, pages 601–609. Springer, 2008.

[125] Jonathan K Pritchard, Matthew Stephens, and Peter Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155(2):945–959, 2000.

[126] Minghui Qiu, Feida Zhu, and Jing Jiang. It is not just what we say, but how we say them: LDA-based behavior-topic model. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 794–802. SIAM, 2013.

[127] Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D Manning. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 248–256. Association for Computational Linguistics, 2009.

[128] Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial Intelligence and Statistics*, pages 814–822, 2014.

[129] SP Ruba Rani, B Ramesh, M Anusha, and JGR Sathiaseelan. Evaluation of stemming techniques for text classification. *International Journal of Computer Science and Mobile Computing*, 4(3):165–171, 2015.

[130] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, 2010. ELRA. `http://is.muni.cz/publication/884893/en`.

[131] Jason Rennie. 20 newsgroups data set, 2008. Available at: http://qwone.com/ jason/20Newsgroups/.

[132] Martin Reynaert. Non-interactive OCR post-correction for giga-scale digitization projects. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 617–630. Springer, 2008.

[133] Lisa Rhody. Topic modeling and figurative language. *Journal of Digital Humanities*, 2(1):19–35, 2012.

[134] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 487–494. AUAI Press, 2004.

[135] M. K. Ross, Wei Wei, and L. Ohno-Machado. "big data" and the electronic health record. *Yearb Med Inform*, 9(1):97–104, 2014.

[136] Tim Salimans, Diederik P Kingma, Max Welling, et al. Markov chain monte carlo and variational inference: Bridging the gap. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1218–1226, 2015.

[137] Evan Sandhaus. The New York Times annotated corpus. *Linguistic Data Consortium*, DVD: LDC2009T19, 2008.

[138] Simone Scardapane, Rosa Altilio, Valentina Ciccarelli, Aurelio Uncini, and Massimo Panella. Privacy-preserving data mining for distributed medical scenarios. In *Multidisciplinary Approaches to Neural Computing*, Smart Innovation, Systems and Technologies, pages 119–128. Springer, Cham, 2018. DOI: 10.1007/978-3-319-56904-8_12.

[139] Aaron Schein, John Paisley, David M Blei, and Hanna Wallach. Bayesian Poisson tensor factorization for inferring multilateral relations from sparse dyadic event counts. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1045–1054. ACM, 2015.

[140] Aaron Schein, Hanna Wallach, and Mingyuan Zhou. Poisson-gamma dynamical systems. In *Advances in Neural Information Processing Systems*, pages 5005–5013, 2016.

[141] Aaron Schein, Zhiwei Steven Wu, Alexandra Schofield, Mingyuan Zhou, and Hanna Wallach. Locally private bayesian inference for count models. *arXiv preprint arXiv:1803.08471*, 2018.

[142] Aaron Schein, Zhiwei Steven Wu, Alexandra Schofield, Mingyuan Zhou, and Hanna Wallach. Locally private bayesian inference for count models. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[143] Alexandra Schofield, Måns Magnusson, and David Mimno. Pulling out the stops: Rethinking stopword removal for topic models. *Proceedings of*

the 17th Conference of the European Chapter of the Association for Computational Linguistics, page 432, 2017.

[144] Alexandra Schofield and David Mimno. Comparing apples to apple: The effects of stemmers on topic models. *Transactions of the Association for Computational Linguistics*, 4:287–300, 2016.

[145] Alexandra Schofield, Aaron Schein, Zhiwei Steven Wu, and Hanna Wallach. A variational inference approach for locally private inference of Poisson factorization models. In *Proceedings of the NeurIPS Workshop on Privacy Preserving Machine Learning (PPML)*, 2018.

[146] Alexandra Schofield, Laure Thompson, and David Mimno. Quantifying the effects of text duplication on semantic models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2737–2747, 2017.

[147] Catarina Silva and Bernardete Ribeiro. The importance of stop word removal on recall values in text categorization. In *Proceedings of the International Joint Conference on Neural Networks*, volume 3, pages 1661–1666. IEEE, 2003.

[148] David A Smith, Ryan Cordell, and Elizabeth Maddock Dillon. Infectious texts: Modeling text reuse in nineteenth-century newspapers. In *Proceedings of the IEEE International Conference on Big Data*, pages 86–94, 2013.

[149] Amber Stubbs and Özlem Uzuner. De-identification of medical records through annotation. In *Handbook of Linguistic Annotation*, pages 1433–1459. Springer, 2017.

[150] Chuan Su. Machine learning for reducing the effort of conducting systematic reviews in SE. *Bachelor Thesis*, 2015.

[151] W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li. Verifiable privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking. *IEEE Transactions on Parallel and Distributed Systems*, 25(11):3025–3035, 2014.

[152] Wenhai Sun, Bing Wang, Ning Cao, Ming Li, Wenjing Lou, Y. Thomas Hou, and Hui Li. Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ASIA CCS '13, pages 71–82, New York, NY, USA, 2013. ACM.

[153] Y. Tang and L. Liu. Privacy-preserving multi-keyword search in information networks. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):2424–2437, 2015.

[154] Yee W Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Sharing clusters among related groups: Hierarchical Dirichlet processes. In *Advances in Neural Information Processing Systems*, pages 1385–1392, 2005.

[155] Yee W Teh, David Newman, and Max Welling. A collapsed variational bayesian inference algorithm for latent Dirichlet allocation. In *Advances in Neural Information Processing Systems*, pages 1353–1360, 2007.

[156] Abhradeep Guha Thakurta, Andrew H Vyrros, Umesh S Vaishampayan, Gaurav Kapoor, Julien Freudiger, Vivek Rangarajan Sridhar, and Doug Davidson. Learning new words, 2017.

[157] Laure Thompson and David Mimno. Authorless topic models: Biasing models away from known structure. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3903–3914, 2018.

[158] Helga Thorvaldsdóttir, James T Robinson, and Jill P Mesirov. Integrative genomics viewer (IGV): High-performance genomics data visualization and exploration. *Briefings in Bioinformatics*, 14(2):178–192, 2013.

[159] Xiang Tong and David A Evans. A statistical approach to automatic OCR error correction in context. In *Fourth Workshop on Very Large Corpora*, 1996.

[160] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003.

[161] Chih-Hsiung Tu. The relationship between social presence and online privacy. *The internet and higher education*, 5(4):293–318, 2002.

[162] Jakob Uszkoreit and Thorsten Brants. Distributed word clustering for large scale class-based language modeling in machine translation. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 755–762, 2008.

[163] Roli Varma. Making computer science minority-friendly. *Communications of the ACM*, 49(2):129–134, 2006.

[164] Hanna M Wallach. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 977–984. ACM, 2006.

[165] Hanna M. Wallach, David M. Mimno, and Andrew McCallum. Rethinking LDA: Why priors matter. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, pages 1973–1981. Curran Associates, Inc., 2009.

[166] Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. Evaluation methods for topic models. In *Proceedings of the 26th International Conference on Machine Learning*, pages 1105–1112. ACM, 2009.

[167] Li Wan, Leo Zhu, and Rob Fergus. A hybrid neural network-latent topic model. In *Artificial Intelligence and Statistics*, pages 1287–1294, 2012.

[168] Yi Wang, Hongjie Bai, Matt Stanton, Wen-Yen Chen, and Edward Y Chang. PLDA: Parallel latent Dirichlet allocation for large-scale applications. In *International Conference on Algorithmic Applications in Management*, pages 301–314. Springer, 2009.

[169] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[170] Y. Xiao, J. Gardner, and L. Xiong. DPCube: Releasing differentially private data cubes for health information. In *2012 IEEE 28th International Conference on Data Engineering*, pages 1305–1308, 2012.

[171] S. Xu, S. Su, X. Cheng, Z. Li, and L. Xiong. Differentially private frequent sequence mining via sampling-based candidate pruning. In *2015 IEEE 31st International Conference on Data Engineering*, pages 1035–1046, 2015.

[172] Z Xu, Minmin Chen, K Weinberger, and Fei Sha. An alternative text representation to TF-IDF and bag-of-words. In *Proceedings of 21st ACM Conf. of Information and Knowledge Management (CIKM)*, 2012.

[173] Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. A biterm topic

model for short texts. In *Proceedings of the 22nd International Conference on the World Wide Web*, pages 1445–1456. ACM, 2013.

[174] Jinhui Yuan, Fei Gao, Qirong Ho, Wei Dai, Jinliang Wei, Xun Zheng, Eric Po Xing, Tie-Yan Liu, and Wei-Ying Ma. LightLDA: Big topic models on modest computer clusters. In *Proceedings of the 24th International Conference on the World Wide Web*, pages 1351–1361. International World Wide Web Conferences Steering Committee, 2015.

[175] Lin Yuan and John D Kalbfleisch. On the bessel distribution and related problems. *Annals of the Institute of Statistical Mathematics*, 52(3):438–447, 2000.

[176] Dongxu Zhang, Tianyi Luo, and Dong Wang. Learning from LDA using deep neural networks. In *Natural Language Understanding and Intelligent Applications*, pages 657–664. Springer, 2016.

[177] Mingyuan Zhou, Yulai Cong, and Bo Chen. The Poisson gamma belief network. In *Advances in Neural Information Processing Systems 28*, pages 3043–3051, 2015.

[178] Shuheng Zhou, Katrina Ligett, and Larry Wasserman. Differential privacy with compression. In *Information Theory, 2009. ISIT 2009. IEEE International Symposium on*, pages 2718–2722. IEEE, 2009.

[179] Tianqing Zhu, Gang Li, Wanlei Zhou, Ping Xiong, and Cao Yuan. Privacy-preserving topic model for tagging recommender systems. *Knowledge and Information Systems*, 46(1):33–58, 2016.

APPENDIX A

THE BESSEL MODE AND MEAN

In this chapter, we introduce the proof of an interesting fact used in performance improvements in Chapter 7: the mode and the mean of a Bessel distribution have a distance bounded by 1.

The intuitive meaning of this is that the mode of the Bessel distribution is guaranteed to be one of the two integers closest to the mean of the distribution. Given one of these two integers will always be the best integer approximation of this number, we can also say that there cannot exist an integer approximation of the mean of the Bessel that is strictly between the mode and the mean of the Bessel distribution.

Proof. A Bessel distribution takes two arguments, which we refer to as its order, ν and coordinate, a. The distribution is defined as:

$$\begin{aligned} &p(x = n \mid x \sim \text{Bessel}\,(\nu, a)) \\ &\qquad = \frac{1}{I_\nu(a) n! \Gamma(n + \nu + 1)} \left(\frac{a}{2}\right)^{2n+\nu}, \end{aligned} \tag{A.1}$$

where $I_\nu(a)$ is a modified Bessel function of the first kind. The arithmetic mean of the distribution is

$$\mathbb{E}_{\text{Bessel}(m;\nu,a))}[m] = \frac{a}{2} R_\nu(a),$$

with $R_\nu(a)$ referring to the ratio of two Bessel functions:

$$\frac{I_{\nu+1}(a)}{I_\nu(a)}.$$

The Bessel distribution has one or two neighboring integer modes. The mode can be computed directly from the parameters of the distribution without Bessel functions:

$$\text{mode}(\text{Bessel}\,(\nu, a)) = \left\lfloor \frac{\sqrt{a^2+\nu^2} - \nu}{2} \right\rfloor.$$

Unlike the mean, which can take arbitrary non-negative real values, the mode is guaranteed by the floor function to be a non-negative integer.

We use the following bound on the mean of a Bessel ratio from [39]:

$$\frac{a}{\nu + 1 + \sqrt{a^2 + (\nu+1)^2}} \leq R_\nu(a) \leq \frac{a}{\nu + \sqrt{a^2+\nu^2}}. \tag{A.2}$$

We multiply through by $\frac{a}{2}$ to bound the mean of the Bessel distribution:

$$\frac{a^2}{2(\nu + 1 + \sqrt{a^2 + (\nu+1)^2})} \leq \mathbb{E}_{\text{Bessel}(m;\nu,a))}[m] \leq \frac{a^2}{2(\nu + \sqrt{a^2+\nu^2})}. \tag{A.3}$$

We can rewrite these bounds using a difference-of-squares:

$$\begin{aligned}
\frac{a^2}{2(\nu + \sqrt{a^2+\nu^2})} &= \frac{a^2(\sqrt{a^2+\nu^2} - \nu)}{2(\nu + \sqrt{a^2+\nu^2})(\sqrt{a^2+\nu^2} - \nu)} \\
&= \frac{a^2(\sqrt{a^2+\nu^2} - \nu)}{2a^2 + \nu^2 - \nu^2} \\
&= \frac{\sqrt{a^2+\nu^2} - \nu}{2}.
\end{aligned}$$

This upper bound coincides with the unrounded formulation of the mode. Because the mode is the floor of this quantity, we know it is less than or equal to this upper bound with a difference of less than 1.

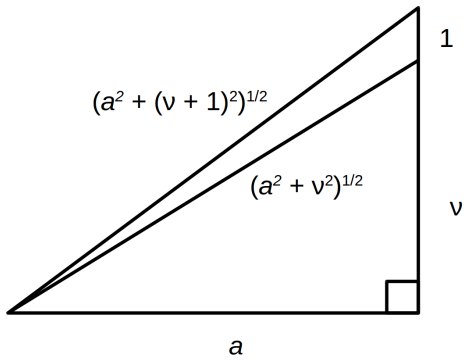
We can convert the lower bound in the same way:

$$\frac{a^2}{2((\nu+1)+\sqrt{a^2+(\nu^2+1)})} = \frac{\sqrt{a^2+(\nu+1)^2}-(\nu+1)}{2}.$$

We are interested in bounding the difference between the upper and lower bounds:

$$\frac{\sqrt{a^2+\nu^2}-\nu}{2} - \frac{\sqrt{a^2+(\nu+1)^2}-(\nu+1)}{2} = \frac{\sqrt{a^2+\nu^2}+1-\sqrt{a^2+(\nu+1)^2}}{2}. \tag{A.4}$$

Knowing that ν is positive, we can state that $\sqrt{a^2+\nu^2} < \sqrt{a^2+(\nu+1)^2}$, or $\sqrt{a^2+(\nu+1)^2} - \sqrt{a^2+\nu^2} > 0$. Based on the upper slice of the triangle in the figure below, the triangle inequality also gives us another bound, that $\sqrt{a^2+\nu^2}+1 > \sqrt{a^2+(\nu+1)^2}$, or $1 > \sqrt{a^2+(\nu+1)^2} - \sqrt{a^2+\nu^2}$.



Together, these imply that

$$0 \leq \sqrt{a^2+\nu^2}+1-\sqrt{a^2+(\nu+1)^2} \leq 1.$$

Substituting this in to our computation of the distance between the upper and lower bounds of the mean, we find that

$$\frac{\sqrt{a^2+\nu^2}+1-\sqrt{a^2+(\nu+1)^2}}{2} < \frac{1}{2}, \tag{A.5}$$

or that the inferred upper and lower bounds for the mean of the Bessel distribution produce an interval no larger than $\frac{1}{2}$. The upper bound of this $\frac{1}{2}$ interval is the same as the upper bound of the length-1 open interval of the mode of the Bessel. In the most extreme case when the mode is at the bottom end of its range and the mean is at the top, we have

$$\mathbb{E}_{\text{Bessel}(m;\nu,a))}[m] - \text{mode}(\text{Bessel}\,(\nu, a)) < 1.$$

In the opposite case, we have

$$\text{mode}(\text{Bessel}\,(\nu, a)) - \mathbb{E}_{\text{Bessel}(m;\nu,a))}[m] \leq \frac{1}{2} < 1.$$

□

APPENDIX B

THE BESSEL AS A CANDIDATE Q-DISTRIBUTION FOR CAVI INFERENCE

In this chapter, we show that that with the ν parameter fixed, the Bessel distribution can be defined as an exponential family function. This reproduces a proof from the appendices of the first version of "Locally Private Bayesian Inference for Count Models" [141], on which I was not an author. While I do not claim credit for this proof, I walk through it in somewhat more detail here.

Proof. The Bessel distribution [175] is a distribution over natural numbers:

$$f(k; a, \nu) = \frac{\left(\frac{a}{2}\right)^{2k+\nu}}{k!\,\Gamma(k+\nu+1)I_\nu(a)}. \tag{B.1}$$

The name Bessel refers to $I_\nu(a)$, a modified Bessel function of the first kind:

$$I_\nu(a) = \sum_{n=0}^{\infty} \frac{\left(\frac{a}{2}\right)^{2n+\nu}}{n!\,\Gamma(n+\nu+1)}. \tag{B.2}$$

To qualify as an exponential family distribution, the probability mass function in Equation B.1 must factor into the following exponentiated structure:

$$pmf(x; \vec{\theta}) = h(x)e^{\eta(\theta)\dot{T}(x)-A(\theta)}, \tag{B.3}$$

where x is the sampled value and θ are the distribution parameters. It is important that the natural parameters η depend only on the parameters θ, not the data, x. Similarly, the sufficient statistics, $T(x)$, reflect only the data and not the parameters. The function is normalized by a base measure $h(x)$ and a log partition $A(\theta)$ to produce a valid PMF.

To put the Bessel function with fixed ν into an exponential family form, we need to factorize into terms depending only on a and terms depending only on k. As we are treating ν as an integer constant instead of a parameter, we will rewrite the PMF as f_ν and replace the Gamma function with a factorial:

$$f_\nu(k;a) = \frac{\left(\frac{a}{2}\right)^{2k+\nu}}{k!(k+\nu)!I_\nu(a)}. \tag{B.4}$$

We then separate out the terms into those depending on a, k, or both:

$$f_\nu(k;a) = (k!(k+\nu)!)^{-1} \cdot (I_\nu(a))^{-1} \cdot \frac{a}{2}^{2k+\nu}. \tag{B.5}$$

We can take $h(k) = (k!(k+\nu)!)^{-1}$ to be the distribution's base measure, and exponentiate the remaining terms:

$$f_\nu(k;a) = h(k)e^{(2k+\nu)(\log(a/2))-\log(I_\nu(a)))}. \tag{B.6}$$

From here, the other three functions fall out, giving the full form:

$$h(k) = (k!(k+\nu)!)^{-1} \tag{B.7}$$

$$T(k) = 2k+\nu \tag{B.8}$$

$$\eta(a) = \log\left(\frac{a}{2}\right) \tag{B.9}$$

$$A(a) = \log I_\nu(a). \tag{B.10}$$

This shows that with fixed parameter ν, the PMF $f_\nu(k;a)$ of the Bessel distribution is exponential family. □