1. Research Statement

Son Nguyen web: https://sonpeter.github.io/ email: sonnguyenkstn@gmail.com

Research Statement

Introduction & Research experience

I am Son Van Nguyen - a Research Resident in Machine Learning Group at VinAI Research, which is a fundamental research institution based in Ha Noi, Vietnam. First of all, I would like to briefly introduce my research interests. Being deeply interested in how to build intelligent systems with the ability to perceive, reason, and make decisions like humans, I have oriented my central research towards developing impactful, reliable and interpretable machine learning algorithms, especially for the tasks of generalizing and adapting to real-world scenarios.

I currently approach this ultimate goal through solving important problems at the intersection of probabilistic modeling and deep learning. Probabilistic modeling could address the main issues of current machine learning systems including interpretability, generalization, principles of uncertainty calibration and robustness. Deep learning, on the other hand, address the shortcomings of the probabilistic approaches by providing universal approximation, representative extraction, as well as facilitating flexible generative modeling and fast approximate Bayesian inference. So, I aim to combine the complementary advantages of these two fields into problems of modeling, inference, and learning. More specifically, I have explored and devised **efficient and scalable probabilistic inference** methods applied in several domains including **Bayesian deep learning, deep generative models, hierarchical Bayesian models, large-scale online and continual learning**. I would like to illustrate my research using the following projects.

Probabilistic inference for Bayesian deep learning

In my latest research, I have conducted a project on approximate inference in Bayesian neural networks (BNNs), **in which I aimed to address the core question of how to construct an expressive approximation for the true posterior while maintaining computational efficiency and scalability**. Inspired by the Bayesian interpretation of Dropout regularization in deep learning models, I proposed a novel structured approximation named **Variational Structured Dropout (VSD)**. The main idea is that if a correlated noise was adopted instead of a factorized one in the Dropout procedure, we would enrich the expressiveness of Dropout posterior via the Bayesian interpretation. Specifically, VSD employs an orthogonal transformation to learn a structured representation on the variational Gaussian noise. As a result, VSD allows statistical dependencies in the approximate posterior and thereby goes beyond the mean-field family. VSD exhibits several significant advantages in terms of both flexibility and computational efficiency for approximate Bayesian inference. Especially, our method has successfully addressed the theoretical pathologies from the non-Bayesian perspective of previous Variational Dropout frameworks, therefore it could be seen as a general-purpose approach for Bayesian inference and in particular for BNNs. This work was published at NeurIPS 2021.

Probabilistic inference for deep generative models

Before the above project, I had an opportunity to conduct a timely research regarding probabilistic inference in autoencoders-based generative models. **Our goal in this research was to facilitate flexibly learning structured priors associated with the generative process of those models, thereby to alleviate the inherent issue of regularizer misspecification in the literature**. Particularly, we focused on a state-of-the-art model called deterministic Relational Regularized Autoencoder (DRAE), which imposes a relational regularization on the structural difference between the prior and the aggregated posterior. Basically, DRAE has provided a plausible solution for our aforementioned goal by utilizing an Optimal Transport discrepancy named sliced fused Gromov Wasserstein (SFGW) as the relational regularization. However, we found that the SFGW could not fully exploit the benefits of relational regularization due to its *uniform* slicing drawbacks. So, we proposed a new class of slicing technique for probability measures by imposing *spherical-based* distributions over projecting directions. The most prominent advantage of this proposal is that it could create a good balance in exploring the distinctiveness and also the informativeness of projections. This work was published at ICLR 2021.

Probabilistic inference for hierarchical Bayesian models on streams

My very first research project was on hierarchical Bayesian models such as Latent Dirichlet Allocation, Dirichlet Process Mixtures. **The core question posed was how to adopt these models to analyze massive datasets, including those arriving in a stream**. I had approached this problem via elegant tools of online Bayesian inference, of which online stochastic optimization and recursive Bayesian updating are the two most active research directions. While methods based on the first approach are potentially prone to inadequate local optima, I investigated the latter and spotted a research gap. Specifically, previous methods based on recursive Bayesian updating applied in conjugate exponential models could suffer from the phenomenon of overconfident posterior, which poorly represents the uncertainty of underlying data distribution. I proposed a novel online Bayesian inference framework with two effective mechanisms to handle that limitation. Concretely, (1) we adopted non-conjugate models which enable using flexible priors with well-calibrated variance to explicitly model temporal changes of the global parameter, and (2) we modeled uncertainties over the global parameter at each minibatch by perturbing it with stochastic Bernoulli noise that is similar to the Dropout technique in deep learning. Theoretically, I proved that our Dropout procedure induces a data-dependent regularization, which allows each parameter component to have its own search space to capture geometric properties, especially highly discriminative characteristics (noisy, sparse data, concept drift), of the data features. This work was published at IEEE BigData 2019.

Ongoing projects & Future plans

I am currently continuing to dig deeper into my latest research on Bayesian deep learning (BDL) on both theoretical and practical fronts, and thereby I aim to address as comprehensively as possible inherent shortcomings of traditional deep learning models on three aspects of **(*anytime*) uncertainty calibration, robustness, and adaptability**.

(**i**)– Firstly, I aim to study more thoroughly the functional behaviors of BNNs, with the ultimate purpose of better understanding the posterior predictive distribution. For deep neural networks, especially overparameterized ones, representation on parameter space may not really reflect the expected behaviors on functional space. This is because of the presence of large regions of degenerate directions which have not been determined by the data. Theoretically understanding the functional representation of deep learning models including discriminativeness and compactness would facilitate us to thoroughly characterize several critical issues such as miscalibrated prediction, catastrophic forgetting, etc. One potential approach I have been thinking of is to develop more effective and scalable functional-space variational inference methods for BNNs. This would require efficient techniques to cope with intractability, implicit parameterization, functional prior choice, etc. So, it would be intriguing but challenging for future works.

(**ii**)– I would also like to promote BDL by combining complementary advantages of kernel-based Bayesian principles (Gaussian Processes, Variational Stein methods, etc.) and deep learning models. For instance, I want to use neural networks to embed compelling inductive biases into Gaussian Processes/kernel methods and vice versa, deploying Gaussian Processes as a building block to facilitate uncertainty estimates and robustness in deep learning models. More generally speaking, I aim to bridge Gaussian Processes and BNNs to bring practical and desirable effects through training techniques of approximate inference, optimization, etc. Furthermore, exploiting the connection between BNNs and kernel learning, via guaranteed frameworks such as Neural Tangent Kernel or linearization approximation, would also allow us to characterize open issues regarding functional behavior of BNNs including loss landscape geometry, training dynamics, high-dimensional optimization on distributional space, etc. In this way, a new and more efficient optimization algorithm for mean-field BNNs is promising.

(**iii**)–Regarding applications, I am aiming for utilizing the reliability and flexibility of BDL techniques in non-stationary scenarios such as active learning, continual learning, domain adaptation (transfer learning), out-of-distribution detection/generalization, and so on. By way of example, I have been conducting a timely research on task-free continual learning, in which I proposed an amortized hierarchical Bayesian inference technique to tackle the catastrophic forgetting of deep learning models when trained with a sequence of tasks without task-specific identification.

All in all, I have a broad interest in impactful but equally challenging problems at the intersection of probabilistic approach and deep learning, with the leverage of powerful principles of Bayesian inference, kernel methods.

2. Curriculum Vitae

SON VAN NGUYEN

VinAI Research, Ha Noi, Viet Nam.
Homepage: sonpeter.github.io
(+84) 965 277 261 ⋄ sonnguyenkstn@gmail.com

EDUCATION

Ha Noi University of Science and Technology (HUST) — Ha Noi, Viet Nam

- Master of Data Science, *Master of Research degree* — Oct 2019 - Apr 2021
  Thesis title: "Improving Bayesian inference in deep neural networks with Variational Structured Dropout"
  CPA: 3.84/4.00, Thesis: 4.00/4.00
- Bachelor of Information Technology, *Program of Talented Engineers* — Aug 2014 - Jun 2019
  Thesis title: "An effective Bayesian approach for discovering hidden semantics from data streams"
  CPA: 3.50/4.00 (rank 2/21 in the talented class), Thesis: 4.00/4.00 (Best Thesis Award)

Phan Boi Chau High School for the Gifted Students, *Specialized Math Class* — Nghe An, Viet Nam
Aug 2011 - Jun 2014

RESEARCH INTEREST

My research currently focuses on methods at the intersection of probabilistic modeling and deep learning, from which I aim to combine the complementary advantages of these two fields into modeling, inference, and learning. I am particularly excited about efficient and scalable probabilistic inference methods applied in complex settings of several domains such as Bayesian deep learning, deep generative models, hierarchical Bayesian models, and large-scale online/continual learning.

EXPERIENCES

VinAI Research (www.vinai.io) — Ha Noi, Viet Nam
AI Research Resident — Jul 2020-present

- Main research topics: Bayesian Deep Learning, Deep Generative Models
- Advisor: Dr. Nhat Ho (Assistant Professor at UT, Austin)
- Knowledge gained: Advances in Bayesian Deep Learning (gradient-based MCMC, Variational Inference with dependence structure, principles of uncertainty estimation, applications in continual/active learning); Deep Generative Models (VAEs, GANs, Normalizing Flows, applications of Optimal Transport)

Applied Rotation Program — Sep 2021-Dec 2021

- Participate in Smart City project involving computer vision tasks such as face detection, face re-identification.

Data Science Laboratory (ds.soict.hust.edu.vn) — Ha Noi, Viet Nam
Research Assistant — Aug 2018 - Jul 2020

- Main research topics: Probabilistic Graphical Model, Bayesian inference
- Advisor: Dr. Khoat Than (Associate Professor at HUST)
- Knowledge gained: Foundations of Machine Learning, Deep Learning and Optimization; Bayesian inference (MCMC, scalable variational approximation, applications in hierarchical Bayesian models and online learning); Topic models (Latent Dirichlet Allocation)

Teaching Assistant — Jan 2020 - Jun 2020

- Machine Learning and Data Mining course

Viettel Network Technology R&D Center, *Department of Data Science* — Ha Noi, Viet Nam
Internship — Jun 2018 - Jun 2019

- Projects: analyze the consumer behavior in telecommunication of millions of users, develop recommendation algorithms for promotions

SUBMISSIONS

1. **Son Nguyen**, Khai Nguyen, Nhat Ho, "Amortized Bayesian Continual Learning", *To be submitted 2022*

PUBLICATIONS

1. Ha Nguyen*, Hoang Pham*, **Son Nguyen***, Linh Ngo, Khoat Than, "Adaptive Infinite Dropout for Noisy and Sparse Data Streams", *Machine Learning journal, 2022*

2. **Son Nguyen**, Duong Nguyen, Khai Nguyen, Khoat Than, Hung Bui*, Nhat Ho*, "Structured Dropout Variational Inference for Bayesian Neural Networks", *Advances in Neural Information Processing Systems (NeurIPS) 2021*

3. Khai Nguyen, **Son Nguyen**, Nhat Ho, Tung Pham, Hung Bui, "Improving Relational Regularized Autoencoders with Spherical Sliced Fused Gromov Wasserstein", *International Conference on Learning Representations (ICLR) 2021*

4. **Son Nguyen**, Tung Nguyen, Linh Ngo, Khoat Than, "Infinite Dropout for training Bayesian models from data streams", *IEEE International Conference on Big Data (Big Data) 2019*

TECHNICAL TALKS

1. Recent Advances in Deep Learning Uncertainty, *Data Science Lab - HUST* — Nov, 2021

2. Structured Dropout Variational Inference for Bayesian Neural Networks, *VinAI NeurIPS Workshop* — Nov, 2021

3. Uncertainty in Deep Learning and the case for Bayesian Deep Learning, *VinAI Research*, slide here — Jun, 2021

4. Optimal Transport for Generative Modelling, *VinAI Research*, slide here — Oct, 2020

PROFESSIONAL SERVICES

Thesis mentor for undergraduate students

- Ha Nguyen, Hoang Pham: Project "Online Bayesian inference methods for noisy and sparse data streams"
- Hoang Phan, Anh Phan: Project "Reducing catastrophic forgetting in neural networks via Gaussian mixture approximation" (accepted to PAKDD 2022, a rank-A conference with acceptance rate of 121/627≈19.30%)

AWARDS AND RECOGNITIONS

1. Vingroup Innovation Foundation (VINIF) Research Scholarship — 2019, 2020

2. Best Thesis Award, Best Presentation Award for undergraduate student — 2019

3. Third Prize in the Scientific Research Student Conference, HUST — 2019

4. Scholarship for students with excellent academic records, HUST — 2015, 2017

5. Vietnam Mathematical Olympiad for university students (VMS) — 2015, 2016
(First Prize in Calculus, Second Prize in Algebra)

6. Scholarship of the National Program for the Development of Mathematics, Vietnam Institute for Advanced Study in Mathematics (VIASM) — 2014, 2015

7. Second prize in Vietnam Mathematical Olympiad (VMO) for high school students — 2014

EDUCATIONAL CONTRIBUTIONS

1. Book: Olympic mathematical topics for gifted students, 2 volumes, *Vietnam National University Press, Ha Noi.* Nguyen Dinh Thanh Cong, Nguyen Van Huong, Nguyen Duy Hung, Tran Tri Kien, **Nguyen Van Son**, Le Nhat, Tran Bao Trung Jul 2017

2. Book: Topics on combinatorics and complex numbers, *Vietnam National University Press, Ha Noi.* Tran Tri Kien, **Nguyen Van Son**, Le Nhat Jul 2016

3. Member of GSTT Group (a non-profit educational organization), lead refresher courses and consolidate the knowledge for high school students Oct 2014 - Oct 2015

SPECIALIZED AND LANGUAGE SKILLS

Programming skills:

- Proficient: Python (PyTorch, numpy, pandas, scikit-learn)
- Familiar: C, JAVA, LATEX

Languages:

- Vietnamese: Native
- English: IELTS 6.5 overall

REFERENCES

Dr. Nhat Ho
Assistant Professor, Department of Statistics and Data Science
The University of Texas at Austin, USA
Email: minhnhat@utexas.edu

Dr. Hung Bui
Founding Director of VinAI Research, Vietnam
Email: v.hungbh1@vinai.io

Dr. Khoat Than
Associate Professor, Head of Data Science Lab
Ha Noi University of Science and Technology, Vietnam
Email: khoattq@soict.hust.edu.vn

Dr. Dinh Phung
Professor, Department of Data Science and AI
Monash University, Australia
Email: dinh.phung@monash.edu

3. Copies of Certificates

SOCIALIST REPUBLIC OF VIETNAM

Independence - Freedom - Happiness

Hanoi University of Science and Technology

PRESIDENT

has conferred the Degree of

MASTER OF SCIENCE

IN COMPUTER SCIENCE

Upon Mr. *Nguyen Van Son*

Date of Birth: *07 July 1996*

Degree Classification: *Very good*

Reference number: TH2021/0216

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập - Tự do - Hạnh phúc

HIỆU TRƯỞNG

Trường Đại học Bách khoa Hà Nội

cấp

BẰNG THẠC SĨ KHOA HỌC

KHOA HỌC MÁY TÍNH

Cho Ông *Nguyễn Văn Sơn*

Ngày sinh: *07/07/1996*

Hạng tốt nghiệp: *Giỏi*

Hà Nội, ngày 26 tháng 07 năm 2021

HIỆU TRƯỞNG

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

PGS.TS. Huỳnh Quyết Thắng

Số hiệu: M 002805

Số vào sổ cấp bằng: TH2021/0216

SOCIALIST REPUBLIC OF VIETNAM

Hanoi University of Science and Technology

PRESIDENT

has conferred

THE DEGREE OF ENGINEER

IN INFORMATION TECHNOLOGY

(Talent Program)

Upon Mr. *Nguyen Van Son*

Date of Birth: *07 July 1996*

Year of Graduation: *2019*

Degree Classification: *Very good*

Mode of Study: *Full-time*

Hanoi, 08 August 2019

Reg. No: **KS2019/1842**

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

HIỆU TRƯỞNG

Trường Đại học Bách khoa Hà Nội

cấp

BẰNG KỸ SƯ

CÔNG NGHỆ THÔNG TIN

(Chương trình đào tạo tài năng)

Cho Ông *Nguyễn Văn Sơn*

Ngày sinh: *07.07.1996*

Năm tốt nghiệp: *2019*

Xếp loại tốt nghiệp: *Giỏi*

Hình thức đào tạo: *Chính quy*

Hà Nội, ngày 08 tháng 08 năm 2019

HIỆU TRƯỞNG

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

PGS.TS. Hoàng Minh Sơn

Số hiệu: E 026682

Số vào sổ cấp bằng: **KS2019/1842**

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATIONS TECHNOLOGY

Best Thesis Award 2019

Presented to

Nguyen Van Son

For his Thesis entitled

Infinite Dropout: An Efficient Bayesian Method for Discovering The Hidden Semantics in Data Streams

THE DEAN

BỘ GIÁO DỤC VÀ ĐÀO TẠO
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Prof. Ta Hai Tung

ĐẠI HỌC BÁCH KHOA
25 YEARS ANNIVERSARY SOICT

No.: 70/QĐ-ĐHBK-CNTT&TT

4. Copies of Transcripts

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
Hanoi University of Science and Technology

PHỤ LỤC VĂN BẰNG
Appendix of Degree

Họ tên/ Full name: **Nguyễn Văn Sơn**
MSHV/ Student ID: **CB190203**

Ngày sinh/ Date of birth: **07/07/1996**
Ngày nhập học/ Date of admission: **08/10/2019**
Ngày tốt nghiệp/ Date of graduation: **26/07/2021**

Ngành đào tạo/ Field of study: **Khoa học máy tính/Computer Science**
Chuyên ngành/ Specialization: **Khoa học dữ liệu/Data Science**
Trình độ đào tạo/ Degree: **Thạc sĩ (Bậc trình độ 7)/ Master**
Hình thức đào tạo/ Mode of study: **Chính quy/ Full-time**
Ngôn ngữ đào tạo/ Instruction Language: **Tiếng Việt/ Vietnamese**
Số hiệu bằng tốt nghiệp/ Degree number: **TH2021/0216**

TT No.	Mã HP Course ID	Tên học phần	Course Title	Tín chỉ Credits	Điểm Grade
1	IT5421	Chuyên đề nghiên cứu 1	Advanced seminar 1	3	A
2	IT5422	Chuyên đề nghiên cứu 2	Advanced seminar 2	3	A
3	IT5404	Điện toán đám mây	Cloud Computing	3	R
4	IT5428	Học máy với dữ liệu lớn	Machine Learning With Big Data	3	A
5	IT5426	Học sâu	Deep Learning	2	R
6	IT5429	Phân tích đồ thị với dữ liệu lớn	Graph Analytics for Big Data	2	B
7	IT5425	Quản trị dữ liệu và trực quan hóa	Data Management and Visualization	2	R
8	IT5427	Tích hợp và xử lý dữ liệu lớn	Big Data Integration and Processing	3	B
9	IT5408	Tính toán hiệu năng cao	High-Performance Computing	3	R
10	SS6013	Triết học	Philosophy	3	A
11	IT5406	Xử lý ngôn ngữ tự nhiên	Natural Language Processing	3	R
12	LV6001	Luận văn tốt nghiệp	Thesis	15	A
13	FL6010	Tiếng Anh	English		P

Tổng số tín chỉ/Credits in Total: **45**

Điểm trung bình tích lũy toàn khóa: **3.84 (qui đổi tương đương sang thang điểm 10: 9.27)**
Cumulative grade-point average: 3.84 (convertible to 10-scale: 9.27)
Đề tài luận văn/ Master thesis title: Giải quyết một số vấn đề trong mạng Nơ-ron bằng phương pháp biến phân sử dụng mô hình trộn Gauss và Dropout/ Overcoming some limitations of deep neural networks with Variational method using Gaussian mixture model and Dropout

Hạng tốt nghiệp: **Giỏi**
Degree classification: Very good
MSHV/ Student ID: CB190203

Hà Nội, ngày 26 tháng 07 năm 2021
Hanoi, 26 July 2021
KT. HIỆU TRƯỞNG/ P.P. PRESIDENT
PHÓ HIỆU TRƯỞNG/ VICE PRESIDENT

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

PGS.TS./Assoc.Prof.Dr. Nguyễn Phong Điền

Ghi chú/ Notes:
(1) All dates appear as dd/mm/yyyy
(2) Hệ thống thang điểm/ Grading system:

Điểm chữ/Grade:	A	B	C	D	F	P	R
Điểm số/Grade points:	4	3	2	1	0	Đạt/Passed	Điểm miễn/ Transfer Credits

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
Hanoi University of Science and Technology

BẢNG ĐIỂM TỐT NGHIỆP ĐẠI HỌC
Graduate Transcript

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Họ tên/Name: **Nguyễn Văn Sơn**
Ngày sinh/Date of birth: **07/07/1996**
MSSV/Student ID: **20143863**
Thời gian học/Time of study: **8/2014 - 6/2019**
Chương trình học: **Kỹ sư Công nghệ thông tin (Chương trình đào tạo tài năng)**
Viện Công nghệ Thông tin và Truyền thông
Degree program: **Engineer in Information Technology (Talent Program)**
School of Information and Communication Technology

TT	Mã HP Course ID	Tên học phần	Course Title	Tín chỉ Credits	Điểm Grade
1	MI1110	Giải tích I	Analysis I	4	A
2	MI1140	Đại số	Algebra	4	A
3	PH1110	Vật lý đại cương I	Physics I	3	C+
4	SSH1110	Những NLCB của CNML I	Fundamental Principles of Marxism-Leninism I	2	A
5	SSH1170	Pháp luật đại cương	General Law	2	C
6	PH1120	Vật lý đại cương II	Physics II	3	C
7	MI1120	Giải tích II	Analysis II	3	A+
8	MI1130	Giải tích III	Analysis III	3	A+
9	EM1010	Quản trị học đại cương	Introduction to Management	2	B+
10	SSH1120	Những NLCB của CNML II	Fundamental Principles of Marxism-Leninism II	3	B+
11	PH3330	Vật lý điện tử	Electronic Physics	3	C
12	SSH1050	Tư tưởng HCM	Ho-Chi-Minh Thought	2	B
13	SSH1130	Đường lối CM của ĐCSVN	Revolution Policy of VCP	3	B
14	IT1110	Tin học đại cương	Introduction to Informatics	4	B+
15	IT2000	Nhập môn CNTT và TT	Introduction to ICT	3	A+
16	MI2020	Xác suất thống kê	Probability and Statistics	3	A
17	IT3010	Cấu trúc dữ liệu và giải thuật	Data Structures and Algorithms	3	D
18	IT3020	Toán rời rạc	Discrete Mathematics	3	A
19	IT3030	Kiến trúc máy tính	Computer Architectures	3	A
20	IT3040	Kỹ thuật lập trình	Programming Techniques	2	A
21	IT3070	Nguyên lý hệ điều hành	Operating Systems	3	B
22	IT3053	Tiếng Anh CN CNTT	English for IT	3	A
23	IT3080	Mạng máy tính	Computer Networks	3	A
24	IT3090	Cơ sở dữ liệu	Database	3	B
25	IT3100	Lập trình hướng đối tượng	Object-Oriented Programming	2	B+
26	IT3110	Linux và phần mềm nguồn mở	Linux and Open Source Software	2	B
27	IT3133	Điện tử số	Digital Electronics	2	B+
28	IT3910	Project I	Project I	3	A+
29	IT3060	Toán chuyên đề	Special Topics in Math	2	A
30	IT3120	Phân tích và thiết kế hệ thống	System Analysis and Design	2	C+
31	IT3920	Project II	Project II	3	A+
32	IT4040	Trí tuệ nhân tạo	Artificial Intelligence	3	B
33	IT4160	Vi xử lý	Microprocessors	3	A
34	IT4173	Xử lý số tín hiệu	Digital Signal Processing	3	A
35	IT4590	Lý thuyết thông tin	Information Theory	2	A
36	IT4013	An toàn thông tin	Information Security	3	B+
37	IT4053	Phân tích và thiết kế thuật toán	Algorithm Analysis and Design	2	A
38	IT4074	Lý thuyết ngôn ngữ và phương pháp dịch	Language Theory and Interpretation	3	B+
39	IT4080	Nhập môn công nghệ phần mềm	Introduction to Software Engineering	2	A
40	IT4090	Xử lý ảnh	Image Processing	2	B+
41	IT4852	Thiết kế và quản trị cơ sở dữ liệu	Database Design and Management	3	A
42	IT4100	Đồ họa và hiện thực ảo	Computer Graphics and Virtual Reality	3	A
43	IT4200	Kỹ thuật ghép nối máy tính	Computer Interfacing	3	C
44	IT4440	Tương tác Người –Máy	Human-Computer Interaction	3	B
45	IT4610	Hệ phân tán	Distributed System	2	A
46	IT4844	Xử lý thông tin mờ	Fuzzy Information Processing	3	A
47	IT4991	Thực tập kỹ thuật	Engineering Practicum	2	A+
48	IT4290	Xử lý tiếng nói	Speech Processing	2	D

TT	Mã HP Course ID	Tên học phần	Course Title	Tín chỉ Credits	Điểm Grade
49	IT4340	Hệ trợ giúp quyết định	Decision Support Systems	3	A
50	IT4520	Kinh tế công nghệ phần mềm	Software Engineering Economics	2	B+
51	IT4680	Truyền thông đa phương tiện và ứng dụng	Multimedia Communications and Applications	2	B
52	IT4752	Tính toán song song	Parallel Computing	2	A
53	IT4940	Project 3	Project 3	3	A+
54	IT5130	Đồ án tốt nghiệp Kỹ sư	Graduation Project	12	A+

Tổng số tín chỉ/Credits in Total: 154

Điểm trung bình tích luỹ toàn khoá: 3.5 (quy đổi tương đương sang thang điểm 10: 8.75)

Cumulative grade-point average: 3.5 (convertible to 10-scale: 8.75)

Xếp loại bằng tốt nghiệp: Giỏi

Degree classification: Very good

MSSV/Student ID: 20143863

Hà Nội, ngày 08 tháng 08 năm 2019

TL. HIỆU TRƯỞNG

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

PHÓ TRƯỞNG PHÒNG ĐÀO TẠO

TS. Nguyễn Xuân Tùng

Ghi chú:

(1) Sinh viên được cấp chứng chỉ riêng cho các môn học Giáo dục thể chất và Giáo dục quốc phòng-an ninh.

(2) Hệ thống thang điểm được quy định như sau:

Notes:

(1) Separate certificates have been issued for Physical Education and Civil Service Education.

(2) The grading system is as follows:

Điểm chữ/Grade	A+	A	B+	B	C+	C	D+	D	F	R
Điểm số/Grade points	4.0	4.0	3.5	3.0	2.5	2.0	1.5	1.0	0	Điểm miễn/Transfer Credits
Thang 10/10-Scale	9.5-10	8.5-9.4	8.0-8.4	7.0-7.9	6.5-6.9	5.5-6.4	5.0-5.4	4.0-4.9	0.0-3.9	

5. A summary/abstract of the master thesis

"Improving Bayesian inference in deep neural networks with Variational Structured Dropout"

Abstract

Bayesian methods promise to address many shortcomings of deep learning. However, core learning algorithms of Bayesian deep models usually employ approximate inference frameworks, which exhibits a dilemma of how to yield high fidelity posterior approximations while maintaining computational efficiency and scalability. We tackle this challenge by introducing a new variational structured approximation inspired by the interpretation of Dropout training as approximate inference in Bayesian probabilistic models. Concretely, we focus on restrictions of the factorized structure of Dropout posterior which is inflexible to capture rich correlations among weight parameters of the true posterior, and we then propose a novel method called Variational Structured Dropout (VSD) to overcome this limitation. VSD employs an orthogonal transformation to learn a structured representation on the variational Dropout noise and consequently induces statistical dependencies in the approximate posterior. We further gain expressive Bayesian modeling for VSD via proposing a hierarchical Dropout procedure that corresponds to the joint inference in a Bayesian network. Theoretically, VSD successfully addresses the pathologies of previous Variational Dropout methods and thus offers a standard Bayesian justification. We further show that VSD induces an adaptive regularization term with several desirable properties which contribute to better generalization. Moreover, we can scale up VSD to modern deep convolutional networks in a direct way with low computational cost. Finally, we conduct extensive experiments on standard benchmarks to demonstrate the effectiveness of VSD over state-of-the-art methods on both predictive accuracy and uncertainty estimation.

5. A list of publications

Structured Dropout Variational Inference for Bayesian Neural Networks

Son Nguyen[†,1] **Duong Nguyen**[3] **Khai Nguyen**[1] **Khoat Than**[3,1] **Hung Bui**[*,1] **Nhat Ho**[*,2]

[1] VinAI Research, Viet Nam
[2] University of Texas, Austin
[3] Hanoi University of Science and Technology

Abstract

Approximate inference in Bayesian deep networks exhibits a dilemma of how to yield high fidelity posterior approximations while maintaining computational efficiency and scalability. We tackle this challenge by introducing a novel variational structured approximation inspired by the Bayesian interpretation of Dropout regularization. Concretely, we focus on the inflexibility of the factorized structure in Dropout posterior and then propose an improved method called *Variational Structured Dropout* (VSD). VSD employs an orthogonal transformation to learn a structured representation on the variational Gaussian noise with plausible complexity, and consequently induces statistical dependencies in the approximate posterior. Theoretically, VSD successfully addresses the pathologies of previous Variational Dropout methods and thus offers a standard Bayesian justification. We further show that VSD induces an adaptive regularization term with several desirable properties which contribute to better generalization. Finally, we conduct extensive experiments on standard benchmarks to demonstrate the effectiveness of VSD over state-of-the-art variational methods on predictive accuracy, uncertainty estimation, and out-of-distribution detection.

1 Introduction

Bayesian Neural Networks (BNNs) [49, 63] offer a probabilistic interpretation for deep learning models by imposing a prior distribution on the weight parameters and aiming to infer a posterior distribution instead of only point estimates. By marginalizing over this posterior for prediction, BNNs perform a procedure of ensemble learning. These principles improve the model generalization, robustness and allow for uncertainty quantification. However, exactly computing the posterior of non-linear BNNs is infeasible and approximate inference has been devised. The core challenge is how to construct an expressive approximation for the true posterior while maintaining computational efficiency and scalability, especially for modern deep learning architectures.

Variational inference is a popular deterministic approximation approach to deal with this challenge. The first practical methods were proposed in [22, 8, 39], in which the approximate posteriors are assumed to be fully factorized distributions, also called mean-field variational inference. In general, the mean-field approximation family encourages several advantages in inference including computational tractability and effective optimization with the stochastic gradient-based methods. However, it ignores the strong statistical dependencies among random weights of neural nets, leading to the inability to capture the complicated structure of the true posterior and to estimate the true model uncertainty.

[*]These two authors contributed equally. [†]Correspondence to Son Nguyen: <v.sonnv27@vinai.io>

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

To overcome this limitation, many extensive studies proposed to provide posterior approximations with richer expressiveness. For instance, [47] treats the weight matrix as a whole via a matrix variate Gaussian [24] and approximates the posterior based on this parametrization. Several later works have exploited this distribution to investigate different structured representations for the variational Gaussian posterior, such as Kronecker-factored [89, 71, 72], k-tied distribution [77], non-centered or rank-1 parameterization [21, 15]. Another original idea to represent the true covariance matrix of Gaussian posterior is by employing the low-rank approximation [67, 35, 80]. For robust approximation with multimodality, [48] adopted hierarchical variational model framework [69] for inferring an implicit marginal distribution in high dimensional Bayesian setting. Despite significant improvements in both predictive accuracy and uncertainty calibration, some of these methods incur a large computational complexity and are difficult to integrate into deep convolutional networks.

Motivations. In this paper, we approach the structured posterior approximation in Bayesian neural nets from a different perspective which has been inspired by the Bayesian interpretation of Dropout training [74, 51]. More specifically, the methods proposed in [39, 20] reinterpret Dropout regularization as approximate inference in Bayesian deep models and base on this connection to learn a variational Dropout posterior over the weight parameters. From the literature, inference approaches based on Bayesian Dropout have shown competitive performances in terms of predictive accuracy on various tasks, even compared to the structured Bayesian methods aforementioned, but with much cheaper computational complexity. Moreover, with the solid and intriguing theories on effective regularization [81, 26, 83], generalization bound [52, 59], convergence rate and robust optimization [55, 54, 7], Dropout principle offers several potentials to further improve approximate inference in Bayesian deep networks. However, since these Bayesian Dropout methods also employed simple structures of the mean-field family, their approximations often fail to obtain satisfactory uncertainty estimates [17]. In addition, Variational Dropout methods based on multiplicative Gaussian noise also suffer from theoretical pathologies, including improper prior leading to ill-posed true posterior, and singularity of the approximate posterior making the variational objective undefined [31].

Contributions. With the above insights, we propose a novel structured variational inference framework, which rationally acquires complementary benefits of the flexible Bayesian inference and Dropout inductive bias. Our method adopts an orthogonal approximation called Householder transformation to learn a structured representation for multiplicative Gaussian noise in Variational Dropout method [39, 57]. As a consequence of the Bayesian interpretation, we go beyond the mean-field family and obtain a variational Dropout posterior with structured covariance. Furthermore, to make our framework more expressive, we deploy a hierarchical Dropout procedure, which is equivalent to inferring a joint posterior in a hierarchical Bayesian neural nets. We name the proposed method as *Variational Structured Dropout* (VSD) and summarize its advantages as follows:

1. Our structured approximation is implemented on low dimensional space of variational noise with considerable computational efficiency. VSD can be employed for deep CNNs in a direct way while maintaining the backpropagation in parallel and optimizing efficiently with gradient-based methods.

2. Especially, VSD has a standard Bayesian justification, in which our method can overcome the critiques from the non-Bayesian perspective of previous Variational Dropout methods. Our inference framework uses a proper prior, non-singular approximate posterior and derives a tractable variational lower bound without further simplified approximation.

3. Compared with previous Bayesian Dropout methods which are relatively inflexible by some *strict conditions*, VSD is more efficient on both the criteria of expressive approximation and flexible hierarchical modeling. Therefore, VSD is promising to be a general-purpose approach for Bayesian inference and in particular for BNNs.

4. To reinforce the complementary advantages unified in our proposal, we also investigate the inductive biases induced by the adaptive regularization of structured Dropout noise. We further provide an interpretation that VSD implicitly facilitates the networks to converge to a local minima with smaller spectral norms and stable rank. This properties suggests better generalization and we present empirical results to support this implication.

5. Finally, we carry out extensive experiments with standard datasets and different network architectures to validate the effectiveness of our method on many criteria, including scalability, predictive accuracy, uncertainty calibration, and out-of-distribution detection, in comparison to popular variational inference methods.

Notation. For a matrix $\mathbf{A}$, $\|\mathbf{A}\|_F$ and $\mathbf{A}^\top$ denotes the Frobenius norm and the transpose matrix, $\mathbf{A}_{i:}$ and $\mathbf{A}_{:j}$ denote the i-th row and the j-th column. For an interger i, $\mathbf{e}_i$ is the i-th standard basis, $\mathbf{1}_i \in \mathbb{R}^i$ is the vector of all ones. The diagonal matrix with diagonal entries as the elements of a vector $\mathbf{x}$ is denoted by diag($\mathbf{x}$). The inner product between two matrices $\mathbf{A}$ and $\mathbf{B}$ is denoted by $\langle \mathbf{A}, \mathbf{B} \rangle$.

2 Background

Variational inference for Bayesian neural networks: Given a dataset $\mathcal{D}$ consisting of input-output pairs $(\mathbf{X}, \mathbf{Y}) := \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$. In BNNs, we impose a prior distribution over random weights $p(\mathbf{W})$ whose form is in a tractable parametric family and aim to infer an intractable true posterior $p(\mathbf{W}|\mathcal{D})$. Variational inference (VI) [30, 33] can do this by specifying a variational distribution $q_\phi(\mathbf{W})$ with free parameter ϕ and then minimizing the Kullback-Leibler (KL) divergence $\mathbb{D}_{KL}(q_\phi(\mathbf{W})\|p(\mathbf{W}|\mathcal{D}))$. This optimization is equivalent to maximizing the Evidence Lower Bound (ELBO) with respect to variational parameters ϕ as follows:

$$\mathcal{L}(\phi) = \mathbb{E}_{q_\phi(\mathbf{W})} \log p(\mathcal{D}|\mathbf{W}) - \mathbb{D}_{KL}(q_\phi(\mathbf{W})\|p(\mathbf{W})). \tag{1}$$

By leveraging the reparameterization trick [40] combined with the Monte Carlo integration, we can derive an unbiased differentiable estimation for the variational objective above. Then, this estimation can be effectively optimized using stochastic gradient methods with the variance reduction technique such as the *local* reparameterization trick [39].

Variational Bayesian inference with Dropout regularization: Given a deterministic neural net with the weight parameter Θ of size $K \times Q$, training this model with stochastic regularization techniques such as Dropout [29, 74] can be interpreted as approximate inference in Bayesian probabilistic models. This is because injecting a stochastic noise into the input layer is equivalent to multiplying the rows of subsequent deterministic weight by the same random variable, namely with each datapoint $(\mathbf{x}_n, \mathbf{y}_n)$ and a noise vector ξ, we have: $\mathbf{y}_n = (\mathbf{x}_n \odot \xi)\Theta = \mathbf{x}_n \text{diag}(\xi)\Theta$. This induces a BNN with random weight matrix defined by $\mathbf{W} := \text{diag}(\xi)\Theta$. Applying VI to this Bayesian model, with some specific choices for prior and approximate posterior, the variational lower bound (1) can resemble the form of Dropout objective in the original deterministic network. This principle is referred to as the KL condition [18]. Gal et al. [20], Kingma et al. [39] used this principle to propose Bayesian Dropout inference methods such as MC Dropout (MCD) and Variational Dropout (VD).

Dropout inference is practical approximate framework especially in high dimensional setting. However, the scope of Bayesian inference in these methods is restricted in terms of flexibility of both prior and approximate posterior. Concretely, the Dropout posteriors $q_\phi(\mathbf{W})$ in MCD and VD both have simple structures of mean-field approximation which often underestimate the variance of true posterior, possibly leading to a poor uncertainty representation [17]. Moreover, in theory, VD employed an improper log-uniform prior which can result in an ill-posed true posterior and generally push the parameters towards the penalized maximum likelihood solution [31]. In addition, VD also suffers from the singularity issue of approximate posterior that makes the KL divergence term undefined. Our work gains an efficient remedy to these pathologies.

3 Variational Structured Dropout

We focus on Bayesian Dropout methods using multiplicative Gaussian noise with correlated parameterization [39]. This procedure induces a random weight $\mathbf{W} = \text{diag}(\xi)\Theta$, where the Dropout noise ξ is a multivariate Gaussian with diagonal covariance $q_\alpha(\xi) = \mathcal{N}(\mathbf{1}_K, \text{diag}(\alpha))$, and α is the droprate vector. The corresponding Dropout posterior then is given by $q_\phi(\mathbf{W}) = \text{Law}(\text{diag}(\xi)\Theta)$. This distribution on each column exhibits a factorized structure with the form of $q(\mathbf{W}_{:j}) = \mathcal{N}(\Theta_{:j}, \text{diag}(\alpha \odot \Theta_{:j}^2))$, whilst allows a correlation on each row because each $\mathbf{W}_{i:}$ is shared by the same scalar noise ξ_i respectively. The parameters $\phi := (\alpha, \Theta)$ are optimized via maximizing a variational lower bound as follows: $\mathcal{L}(\phi) := \mathbb{E}_{q_\phi(\mathbf{W})} \log p(\mathcal{D}|\mathbf{W}) - \mathbb{D}_{KL}(q_\phi(\mathbf{W})\|p(\mathbf{W})) = \mathbb{E}_{q_\alpha(\xi)} \log p(\mathcal{D}|\xi, \Theta) - \mathbb{D}_{KL}(q_\phi(\mathbf{W})\|p(\mathbf{W}))$, where the later equation is derived from the change of variables formula.

3.1 The orthogonal approximation for variational structured noise

Intuitively, a richer representation for the noise distribution can enrich the expressiveness of Dropout posterior via the Bayesian interpretation. We implement this intuition with an assumption that the

Dropout noise could be sampled from a Gaussian distribution with a full covariance matrix instead of a diagonal structure, namely, $q_{\mathbf{\Sigma}}(\xi) = \mathcal{N}(\mathbf{1}_K, \mathbf{\Sigma})$ with $\mathbf{\Sigma}$ is a positive definite matrix of size $K \times K$. To make this covariance matrix learnable, we first represent $\mathbf{\Sigma}$ in the form of the spectral decomposition: $\mathbf{\Sigma} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}^T$, where $\mathbf{P}$ is an orthogonal matrix with its eigenvectors in columns, $\mathbf{\Lambda}$ is a diagonal matrix where diagonal elements are the eigenvalues. By the basis-kernel representation theorem [6, 76], we parameterize the orthogonal matrix $\mathbf{P}$ as a product of Householder matrices in the following form: $\mathbf{P} = \mathbf{H}_{T^*}\mathbf{H}_{T^*-1}...\mathbf{H}_1$, where $\mathbf{H}_t = \mathbf{I} - 2\mathbf{v}_t\mathbf{v}_t^T/\|\mathbf{v}_t\|_2^2$, $\mathbf{v}_t$ is the Householder vector of size K, and T^* is the degree of $\mathbf{P}$. This parameterization relaxes the orthogonal constraint of matrix $\mathbf{P}$, and we can then directly optimize the covariance matrix $\mathbf{\Sigma}$ via gradient-based methods.

Notably, this transformation can be interpreted as a sequence of invertible mappings. More explicitly, we extract a zero-mean Gaussian noise $\eta^{(0)}$ from the original noise $\xi^{(0)} \sim \mathcal{N}(\mathbf{1}_K, \text{diag}(\alpha))$ in the form of $\xi^{(0)} = 1 + \eta^{(0)}$, and by successively transforming $\eta^{(0)}$ through a chain of T Householder reflections, we obtain the induced noise and the corresponding density at each step t as follows:

$$\xi^{(t)} := 1 + \mathbf{H}_t\mathbf{H}_{t-1}...\mathbf{H}_1\eta^{(0)} = 1 + \mathbf{U}\eta^{(0)}, \quad \text{and} \quad q_t(\xi) := \mathcal{N}(\mathbf{1}_K, \mathbf{U}\text{diag}(\alpha)\mathbf{U}^T).$$

By injecting the structured noise $\xi^{(t)}$ into the deterministic weight Θ, we obtain a random weight $\mathbf{W}^{(t)} := \text{diag}(\xi^{(t)})\Theta$ and an induced Dropout posterior $q_t(.) = \text{Law}(\text{diag}(\xi^{(t)})\Theta)$ with *fully correlated* representation, in which the marginal column distribution is given by: $q_t(\mathbf{W}_{:j}) = \mathcal{N}(\Theta_{:j}, \text{diag}(\Theta_{:j})\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\Theta_{:j}))$. Detailed discussion about the expressiveness of this correlated structure is presented in Appendix A.1. With the above derivations, we use variational inference with approximate posterior $q_t(\mathbf{W})$ and optimize the variational lower bound as follows:

$$\begin{aligned}\mathcal{L}(\phi) &:= \mathbb{E}_{q_t(\mathbf{W})} \log p(\mathcal{D}|\mathbf{W}) - \mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})) \\ &= \mathbb{E}_{q_\alpha(\xi)} \log p(\mathcal{D}|\Theta, \xi^{(t)}) - \mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})). \end{aligned} \tag{2}$$

Overcoming the singularity issue of approximate posterior. In Variational Dropout method with correlated parameterization, there is a mismatch in support between the approximate posterior and the prior, thus making the KL term $\mathbb{D}_{KL}(q(\mathbf{W})||p(\mathbf{W}))$ undefined [31]. Specifically, the form $\mathbf{W} = \text{diag}(\xi)\Theta$ is equivalent to multiplying each row $\mathbf{W}_{i:}$ by the same scalar noise ξ_i, namely $\mathbf{W}_{i:} = \xi_i\Theta_{i:}$ with $q(\xi_i) = \mathcal{N}(1, \alpha_i)$. This means that the approximate distribution always assigns all its mass on subspaces defined by the directions aligned with the rows of Θ. These subspaces have Lebesgue measure zero causing the singularities in approximate posterior, and the KL term will be undefined whenever the prior $p(\mathbf{W})$ puts zero mass to these subspaces. However, in VSD the scalar noises are not treated separately due to the structured correlation, namely VSD would injects each ξ_i into the whole matrix Θ instead of some individual directions. Indeed, we have: $\mathbf{W}^{(VD)} := \text{diag}(\xi^{(0)})\Theta = \Theta + \text{diag}(\eta^{(0)})\Theta = \Theta + \sum_{i=1}^K \eta_i^{(0)}(\text{diag}(\mathbf{e}_i)\Theta) = \Theta + \sum_{i=1}^K \eta_i^{(0)}\Theta_{(i)}$, and $\mathbf{W}^{(VSD)} := \text{diag}(\xi^{(t)})\Theta = \Theta + \text{diag}(\mathbf{U}\eta^{(0)})\Theta = \Theta + \sum_{i=1}^K \eta_i^{(0)}(\text{diag}(\mathbf{U}_{i:})\Theta)$, where $\Theta_{(i)}$ is the matrix Θ with only the i-th row retained. While VD causes *singular* components represented by $\{\Theta_{(i)}\}_{i=1}^K$, VSD maintains a trainable orthogonal matrix $\mathbf{U}$ which prevents the approximate posterior from having degenerate supports with measure zero, thereby avoiding the singularity issue. In the following section, we will present an appropriate choice of the prior distribution $p(\mathbf{W})$ such that the KL term is well-defined, and then derive a tractable objective function satisfying the KL condition in Bayesian Dropout frameworks.

3.2 Derivation of tractable variational objective

We consider employing an isotropic Gaussian as the prior distribution, namely $p(\mathbf{W}) = \prod_{j=1}^Q p(\mathbf{W}_{:j})$ with $p(\mathbf{W}_{:j}) = \mathcal{N}(0, \text{diag}(\beta_{:j}^{-1}))$ and β is a hyper-parameter matrix of the same size with $\mathbf{W}$. With the previous analysis, our approximate posterior $q_t(\mathbf{W})$ would be absolutely continuous w.r.t the prior $p(\mathbf{W})$, and thus the KL term $\mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W}))$ is defined. Furthermore, the Gaussian prior helps our proposal avoid the pathologies of improper true posterior and ill-posed inference in VD.

Note that, the prior $p(\mathbf{W})$ is a fully factorized Gaussian which usually facilitates simple analysis and efficient computation. This factorized structure is chosen also because we have no reason for the correlation between non-identical weights at first. Moreover, several arguments indicate that a simple prior over parameter $p(\mathbf{W})$, when interacts with neural nets architecture $f(\mathcal{D}; \mathbf{W})$, induces a sophisticated prior over function $p(f(\mathcal{D}; \mathbf{W}))$, with desirable properties and useful inductive

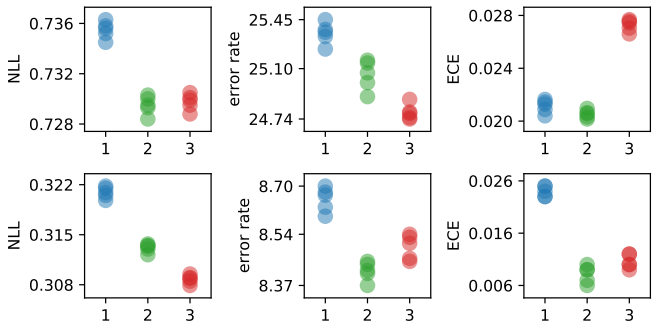


Figure 1: The performance of VSD when using the number of transformations $T \in \{1, 2, 3\}$. Evaluation over 5 runs on CIFAR10 (above) and SVHN (below) with LeNet architecture.

biases [86]. However, when we are interested in structured approximations in parameter space, the factorized prior may raise some contradictions. By relative entropy decomposition, we have:

$$\mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})) = \sum_{j=1}^{Q} \mathbb{D}_{KL}(q_t(\mathbf{W}_{:j})||p(\mathbf{W}_{:j})) + \mathbf{I}(\mathbf{W}_{:1}, \mathbf{W}_{:2}, ..., \mathbf{W}_{:Q}), \tag{3}$$

where $\mathbf{I}(.)$ is the mutual information measured by the distribution $q_t(.)$, and this term is validly defined in VSD. Maximizing the variational lower bound tends to encourage smaller KL term, and hence constrains the components in RHS of equation (3). Intuitively, a relatively small mutual information can break the strong correlations between the columns of $\mathbf{W}$. Several studies have focused on this limitation and suggested using richer priors such as matrix variate Gaussian [75, 91], doubly semi-implicit distribution [58]. Our solution for this scenario is derived naturally from equation (3), in which we leverage the mutual information as an additional regularization term. Concretely, we maximize an alternative variational objective as follows:

$$\begin{aligned}\mathcal{L}_{MI}(\phi) &:= \mathbb{E}_{q_\alpha(\xi)} \log p(\mathcal{D}|\Theta, \xi^{(t)}) - \mathbb{D}_{KL}(q_t(\mathbf{W})||p(\mathbf{W})) + \mathbf{I}(\mathbf{W}_{:1}, \mathbf{W}_{:2}, ..., \mathbf{W}_{:Q}) \\ &= \mathbb{E}_{q_\alpha(\xi)} \log p(\mathcal{D}|\Theta, \xi^{(t)}) - \mathbb{D}_{KL}(q_t^\star(\mathbf{W})||p(\mathbf{W})),\end{aligned} \tag{4}$$

where $q_t^\star(\mathbf{W}) := \prod_{j=1}^{Q} q_t(\mathbf{W}_{:j})$ is the product of marginal column distributions. From the information-theoretic perspective, augmenting the mutual information is a standard principle for structure learning in Bayesian networks [41]. Particularly in our derivation, maximizing the alternative objective $\mathcal{L}_{MI}(\phi)$ would help to sustain the dependence structure between columns of the network weights, and thus fixes appropriately the *broken* ELBO as mentioned above. Interestingly, this technique is utilized reasonably in our method. This is because our dependence structure allows to specify explicitly the marginal distribution on each column of $\mathbf{W}$, leading to a tractable objective in equation (4) whose KL divergence between two multivariate Gaussian can be calculated analytically in closed-form. We note that a similar application to other structured approximations, such as low-rank, Kronecker-factored or matrix variate Gaussian, can be non-trivial. We also remark that our alternative variational objective might be not necessarily a valid lower bound of the original model evidence, but would be the lower bound of new model evidence defined on an alternative prior $\hat{p}(\mathbf{W})$, which satisfies $\mathbb{D}_{KL}(q_t(\mathbf{W})||\hat{p}(\mathbf{W})) = \mathbb{D}_{KL}(q_t^\star(\mathbf{W})||p(\mathbf{W}))$. Indeed, the correlated prior determined by $\hat{p}(\mathbf{W}) \propto p(\mathbf{W}) * q_t(\mathbf{W})/q_t^\star(\mathbf{W})$ meets this constraint, and thus we could reinterpret the use of mutual information as adopting this prior at each iteration of the training procedure. To clarify, our idea was partly motivated by the similar technique that has been extensively adopted in deep latent variable models, in which a mutual information maximization is also added to the variational lower bound to mitigate the degenerate issue of amortized inference in these models [1, 92].

The KL condition in VSD. With the new variational objective in equation (4), to offer VSD complementary advantages of structured Dropout and Bayesian inference, we need to ensure $\mathbb{D}_{KL}(q_t^\star(\mathbf{W})||p(\mathbf{W}))$ satisfies the KL condition. We solve this prerequisite by specifying the precision parameter β of the prior $p(\mathbf{W})$ via the Empirical Bayes (EB) approach. As a result, we obtain an optimal value for this hyperparameter in an analytical form β^*. The optimal β^* is then substituted back into the prior, and thereby we get the optimal KL term with a form independent of the deterministic weight Θ as follows:

$$\mathbb{D}_{KL}^{EB}(q_t^\star(\mathbf{W})||p(\mathbf{W})) = \frac{Q}{2} \sum_{i=1}^{K} \log \frac{1 + \sum_{j=1}^{K} \alpha_j U_{ij}^2}{\alpha_i}. \tag{5}$$

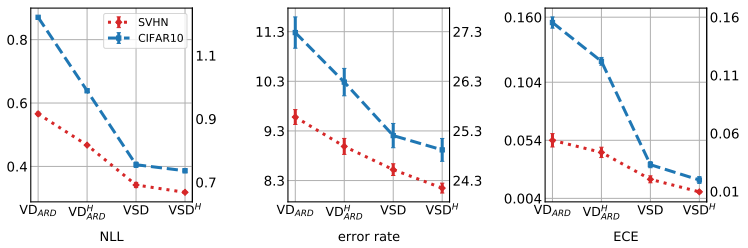


Figure 2: The performance of ARD-VD and VSD when using the prior hierarchy (coressponding to the labels VD^{H}_{ARD} and VSD^{H}). The left and right $y-$axis are represented for SVHN and CIFAR10 dataset, respectively.

A formal proof of equation (5) is in Appendix B.

The number of transformations steps T in VSD. An appropriate choice of T is essential in our method. For deep learning architectures, the degree T^* of the orthogonal matrix P might be relatively large, so it is quite challenging to adjust the empirical value of T in a principled way to meet the basis-kernel representation theorem. We can employ an efficient parameterization introduced in [90], in which we only need the Householder vectors $\{\mathbf{v}_t, t \geq 1\}$ with sizes much smaller than the order of the matrix P. This will facilitate tuning T with a larger range in an applicable computation time. In our method, we instead only adopt a small $T \in \{1, 2, 3\}$ as a form of *"low-degree"* approxiamtion, and to make the reflections more expressive, we use a fully connected layer between successive Householder vectors, i.e. $\mathbf{v}_t = \mathbf{FC}(\mathbf{v}_{t-1})$. The low dimension of variational noise in VSD leads to a good adaptation, but with a little trade-off in computational complexity when increasing the number of transformation steps T. We show the performance of VSD when using this neural parameterization with $T \in \{1, 2, 3\}$ in Figure 1, where a larger T could potentially improve the results on predictive measures. However, to maintain computational efficiency, we recommend using $T = 1$ or 2 in large-scale experiments. Indeed, this neural parameterization has been successfully implemented in the context of learning the latent space in deep latent variable models [79, 5].

3.3 Joint inference with hierarchical prior

We further promote our proposal by introducing a prior hierarchy in VSD framework and then obtain a joint approximation for the Dropout posterior. This will facilitate the flexibility of Bayesian inference in our method in terms of the expressiveness of both prior distribution and approximate posterior. We design a two-level hierarchical prior given by: $p(\mathbf{W}, \mathbf{z}) = p(\mathbf{W}|\mathbf{z}, \beta)p(\mathbf{z})$, where $p(\mathbf{W}|\mathbf{z}, \beta) = \prod_{j=1}^{Q} p(\mathbf{W}_{:j}|\mathbf{z}, \beta_{:j})$ and $p(\mathbf{W}_{:j}|\mathbf{z}, \beta_{:j}) = \mathcal{N}(0, \text{diag}(\mathbf{z} \odot \beta_{:j}^{-1}))$; the hyperprior $p(\mathbf{z})$ is a distribution with positive support such as Gamma or half-Cauchy distribution; the latent $\mathbf{z}$ has the size of the number of rows and is shared across columns of the weight matrix $\mathbf{W}$; the hyper-parameter matrix β is treated as a scaling factor. This prior family has a centered parameterization and induces several compelling properties such as facilitating feature sparsity [46, 12], model selection [21] or improving robustness, uncertainty calibration [15].

We implement variational inference with a joint approximate posterior, also referred to as the joint Dropout posterior, which is parameterized as follows:

$$q_\phi(\mathbf{W}, \mathbf{z}) = q_\psi(\mathbf{z})q_\phi(\mathbf{W}|\mathbf{z}) \quad \text{with} \quad q_\phi(\mathbf{W}_{:j}|\mathbf{z}) = \mathcal{N}(\mathbf{z} \odot \Theta_{:j}, \text{diag}(\mathbf{z}^2 \odot \alpha \odot \Theta_{:j}^2)),$$

where $q_\phi(\mathbf{W}|\mathbf{z})$ is the conditional Dropout posterior, $q_\psi(\mathbf{z})$ is chosen depending on the family of prior $p(z)$ so that the reparametrization trick can be utilized. Sampling the random weight $\mathbf{W}$ from the joint variational posterior $q_\phi(\mathbf{W}, \mathbf{z})$ includes two steps: $\mathbf{z}^* \sim q_\psi(\mathbf{z})$ and $\mathbf{W}^* \sim q_\phi(\mathbf{W}|\mathbf{z}^*)$, in which the second one can be reparameterized as: $\mathbf{W}^* = \text{diag}(\mathbf{z}^*)\text{diag}(\xi)\Theta = \text{diag}(\mathbf{z}^* \odot \xi)\Theta$, with the noise $\xi \sim \mathcal{N}(\mathbf{1}_K, \text{diag}(\alpha))$. This new representation adapts to the vanilla Dropout procedure but allows our method to regularize each unit layer with different levels of stochasticity. We derive some insights about the role of hierarchical prior in our framework in Appendix A.2. We apply the Householder transformation to the variational noise ξ and obtain a new joint approximate posterior:

$$q_t(\mathbf{W}, \mathbf{z}) = q_\psi(\mathbf{z})q_t(\mathbf{W}|\mathbf{z}) \quad \text{with} \quad q_t(\mathbf{W}_{:j}|\mathbf{z}) = \mathcal{N}(\mathbf{z} \odot \Theta_{:j}, \mathbf{V}_j\mathbf{U}\text{diag}(\alpha)(\mathbf{V}_j\mathbf{U})^T),$$

where $\mathbf{V}_j = \text{diag}(\mathbf{z} \odot \Theta_{:j})$. Similarly, we define $q_t^\star(\mathbf{W}|\mathbf{z}) = \prod_{j=1}^{Q} q_t(\mathbf{W}_{:j}|\mathbf{z})$ and then optimize an alternative variational objective given by:

$$\mathcal{L}(\phi, \psi) := \mathbb{E}_{q_\psi(\mathbf{z})q_t(\mathbf{W}|\mathbf{z})} \log p(\mathcal{D}|\mathbf{W}) - \mathbb{E}_{q_\psi(\mathbf{z})}(\mathbb{D}_{KL}(q_t^\star(\mathbf{W}|\mathbf{z}, \phi)||p(\mathbf{W}|\mathbf{z}, \beta)) - \mathbb{D}_{KL}(q_\psi(\mathbf{z})||p(\mathbf{z})).$$

Table 1: Computational complexity per layer of MAP and different variational methods.

Method	Time	Memory
MAP	$\mathcal{O}(KL\|\mathcal{B}\|)$	$\mathcal{O}(L\|\mathcal{B}\|)$
BBB	$\mathcal{O}(sKL\|\mathcal{B}\|)$	$\mathcal{O}(sKL + L\|\mathcal{B}\|)$
BBB-LTR	$\mathcal{O}(2KL\|\mathcal{B}\|)$	$\mathcal{O}(2L\|\mathcal{B}\|)$
VMG	$\mathcal{O}(m^3 + 2KL\|\mathcal{B}\|)$	$\mathcal{O}(KL\|\mathcal{B}\|)$
SLANG	$\mathcal{O}(r^2KL + rsKL\|\mathcal{B}\|)$	$\mathcal{O}(rKL + sKL\|\mathcal{B}\|)$
ELRG	$\mathcal{O}(r^3 + (r+2)KL\|\mathcal{B}\|)$	$\mathcal{O}((r+2)L\|\mathcal{B}\|)$
VSD	$\mathcal{O}(K^2 + KL\|\mathcal{B}\|)$	$\mathcal{O}(K^2 + K\|\mathcal{B}\|)$
VSD-low rank	$\mathcal{O}(rK + KL\|\mathcal{B}\|)$	$\mathcal{O}(K^2 + K\|\mathcal{B}\|)$

Table 2: Computation time of variational methods compared to standard MAP (1x).

Methods	Time/epochs (s)		
	LeNet5	AlexNet	ResNet18
BBB-LTR	1.53x	1.75x	3.28x
MNF	2.86x	3.40x	4.88x
VD	1.18x	1.15x	1.32x
VSD $T = 1$	1.25x	1.32x	1.86x
VSD $T = 2$	1.35x	1.49x	2.90x

We have chosen the (inverse) Gamma and log-Normal distribution for $p(z)$ and $q(z)$ respectively. These distributions have positive supports, can be reparametrized and the KL-divergence between them also has a closed-form due to the conjugacy. A full derivation of $\mathcal{L}(\phi, \psi)$ including the KL condition is given in Appendix C. The parameterization of the prior hierarchy in our method is flexible without any simplifying assumptions about hyperprior $p(\mathbf{z})$. We can directly apply it for ARD-Variational Dropout framework (ARD-VD) [36] (a derivation is in Appendix F). As the experimental results are shown in Figure 2, the hierarchical prior significantly improves the performance of both ARD-VD and VSD on predictive metrics. Therefore, we aim to introduce VSD with hierarchical prior as an unified framework and a general-purpose approach for Bayesian inference, particularly for BNNs.

3.4 Scalability of Variational Structured Dropout

Approximating a structured posterior directly on the random weights of deep convolutional models is challenging. Besides expensive computation, it is difficult to employ the local reparameterization trick [39], leading to the high variance issue in training. We apply VSD to convolutional layer by learning a structured noise with the size of the number of kernels and imposing it to convolutional weights: $\xi \sim \mathcal{N}(\mathbf{1}_K, \mathbf{U}\text{diag}(\alpha)\mathbf{U}^T)$ and $\mathbf{W}_{ijk} = \xi_k \Theta_{ijk}$, with i, j, k are the indexes representing height, width, and kernel respectively. This simple solution greatly reduces computational complexity while being able to captures the dependencies among kernels of the convolutional layer.

We present the complexity of MAP and VI methods in terms of computational cost and memory storage in Table 1, with the detailed analysis given in Appendix D. VSD adopts the advantage of Dropout training and maintains an efficiency on both criteria. We also give more results in Table 2 about the empirical computation time of VSD and some other methods. Based on these tables, VSD shows more effective running time even than the mean-field BBNs. Although there is a trade-off when using a larger number of T, VSD does not incur much extra computation time compared to VD.

3.5 On explicit regularization of Variational Structured Dropout

There are several compelling theories to explain the tremendous success of Dropout technique, in which regularization-based is one of the most active approaches [81, 26, 55, 53, 83, 10]. We would follow this direction to investigate inductive biases induced by the structured Dropout in VSD, and from which to consolidate our claim of complementary advantages unified in the proposed method. To characterize the regularization of VSD, we consider a deep linear neural net with L layers parameterized by $\{\Theta^{(i)}\}_{i=1}^{L}$, and define some notations as: $\mathbf{x}$ is an input data, $\mathcal{B}$ is the data batch; $\mathbf{h}_i$ is the i-th hidden layer; $\mathbf{J}_i(\mathbf{x})$ denotes the Jacobian of network output w.r.t $\mathbf{h}_i(\mathbf{x})$; $\mathbf{H}_i(\mathbf{x})$ and $\mathbf{H}_{\text{out}}(\mathbf{x})$ denotes the Hessian of the loss w.r.t $\mathbf{h}_i(\mathbf{x})$ and the network output, respectively. Then we have $\mathbf{J}_i = (\prod_{l=i}^{L} \Theta^{(l)})^T \triangleq \Theta^{[i:L]}$ the transposition of linear multiplication of weight matrices from i-th layer to the last one. From a detailed derivation presented in Appendix E, VSD induces an explicit regularization given by:

$$R_{VSD} = \mathbb{E}_{(\mathbf{x}\sim\mathcal{B})} \sum_{i=1}^{L} \left\langle \mathbf{H}_i, \text{diag}(\mathbf{h}_i)\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\mathbf{h}_i) \right\rangle .$$

Note that, $\mathbf{H}_i$ can be approximated by $\mathbf{J}_i^T \mathbf{H}_{\text{out}} \mathbf{J}_i$ after ignoring the non-PSD term which is less important empirically [73]. This regularizer offers some intriguing but popular interpretations related

Table 3: Results for VSD and baselines on vectorized MNIST, CIFAR10 and SVHN. Results are averaged over 5 random seeds. For all metrics, lower is better.

Method	MNIST						CIFAR10			SVHN		
	FC 400x2			FC 750x3			CNN 32x64x128					
	NLL	err. rate	ECE	NLL	err. rate	ECE	NLL	err. rate	ECE	NLL	err. rate	ECE
MAP	0.098	1.32	0.011	0.109	1.27	0.011	2.847	34.04	0.272	0.855	12.26	0.086
BBB	0.109	1.59	0.011	0.140	1.50	0.013	1.202	30.11	0.098	0.545	10.57	0.017
MCD	0.049	1.26	0.007	0.057	1.22	0.007	0.794	26.91	0.024	0.365	9.23	0.013
VD	0.051	1.21	0.007	0.061	1.17	0.008	1.176	27.45	0.156	0.534	9.47	0.055
ELRG	0.053	1.54	-	-	-	-	0.871	29.43	-	-	-	-
VSD	**0.042**	**1.08**	**0.006**	**0.048**	**1.09**	**0.006**	**0.730**	**24.92**	**0.020**	**0.299**	**8.39**	**0.008**
D.E	0.057	1.29	0.009	0.063	1.21	0.009	1.815	26.44	0.042	0.783	9.31	0.070
SWAG	0.044	1.27	0.008	**0.043**	1.25	0.007	0.799	26.94	**0.012**	0.312	8.42	0.021

to the curvature of loss landscape [83, 10] (see a detailed explanation in Appendix E). We now show novel properties of VSD induced by the orthogonal matrix $\mathbf{U}$.

VSD imposes a Tikhonov-like regularization and reshapes the gradient of network weights: We rewrite our regularization corresponding to layer i-th by: $R^{(i)}_{VSD} = \mathbb{E}_{(\mathbf{x}\sim\mathcal{B})}\|\mathbf{H}_i^{1/2}\text{diag}(\mathbf{h}_i)\mathbf{U}\text{diag}(\alpha^{1/2})\|_F^2$. This form can be interpreted as the Tikhonov-*like* regularization imposed on the square root of Hessian matrix $\mathbf{H}_i$, in which the Tikhonov matrix $\Gamma := \text{diag}(\mathbf{h}_i)\mathbf{U}\text{diag}(\alpha^{1/2})$ is automatically learned during training. This principle can improve the conditioning of the estimation problem. Furthermore, when considering the case of regression problem, we have:

$$R^{(i)}_{VSD} = \mathbb{E}_{(\mathbf{x}\sim\mathcal{B})}\left[\Theta^{[i:L]}\text{diag}(\mathbf{h}_i)\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\mathbf{h}_i)\Theta^{[i:L].T}\right]. \tag{6}$$

This is a data-dependent regularization with adaptive structure determined by the matrix $\Gamma\Gamma^T = \text{diag}(\mathbf{h}_i)\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\mathbf{h}_i)$. From the algorithmic perspective, this regularizer allows VSD to reshape the gradient of network weights according to the geometry of the data based on both scale and direction information [14, 25]. Meanwhile $\Gamma\Gamma^T = \text{diag}(\mathbf{h}_i)\text{diag}(\alpha)\text{diag}(\mathbf{h}_i)$ only plays as a scaling factor in VD.

VSD penalizes implicitly the spectral norm of weight matrices: Let $\Omega_i := \text{diag}(\mathbf{h}_i)\mathbf{J}_i^T\mathbf{H}_{\text{out}}\mathbf{J}_i\text{diag}(\mathbf{h}_i)$, then our regularizer can be rewritten as: $R^{(i)}_{VSD} = \mathbb{E}_{(\mathbf{x}\sim\mathcal{B})}\sum_{k=1}^{K}\alpha_k^2\mathbf{U}_{:k}^T\Omega_i\mathbf{U}_{:k}$. Since the trainable matrix $\mathbf{U}$ satisfies $\mathbf{U}_{:k}^T\mathbf{U}_{:k} = 1$ for any k, a penalty on $\mathbf{U}_{:k}^T\Omega_i\mathbf{U}_{:k}$ implies that VSD likely prefers a solution with smaller spectral norms of the matrix $\mathbf{H}_{\text{out}}^{1/2}\mathbf{J}_i\text{diag}(\mathbf{h}_i)$ and thus of the network weights. This implication points us to the well-studied theories about generalization bound based on the spectral norm [4, 65]. Concretely, Neyshabur et al. [65] suggests that smaller spectral norm and stable rank can lead to better generalization. This expectation can be observed empirically in VSD through Table 9 in Appendix E. A more solid investigation about the generalization of VSD is of interest.

4 Experiments

In this section, we provide experimental evaluations to show the effectiveness of our proposed methods compared with the existing methods in terms of both predictability and scalability. We focus mainly on variational inference methods of the following two directions: the first one is direct approximations of the posterior on the random weights of Bayesian nets, including Bayes by Backprop (BBB) [8], Variational Matrix Gaussian (VMG) [47], low-rank approximations (SLANG, ELRG) [56, 80]; and the other one is the Bayesian Dropout methods with MC Dropout (MCD) [19, 20], Variational Dropout (VD) [39], and our method-Variational Structured Dropout (VSD). In addition, we evaluate the performance of point estimate framework MAP and two standard non-variational baselines Deep Ensemble (D.E) [44] and SWAG [50]. Details about data descriptions, network architectures, hyper-parameter tuning are presented in Appendix I.

4.1 Image classification

We now compare the predictive performance of the aforementioned methods for classification tasks on three standard image datasets: MNIST [45], CIFAR10 [43], and SVHN [64]. We evaluate the

Table 4: Image classification using AlexNet architecture. Results are averaged over 5 random seeds.

AlexNet	CIFAR10			CIFAR100			SVHN			STL10		
	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE
MAP	1.038	69.58	0.121	4.705	40.23	0.393	0.418	87.56	0.033	2.532	65.70	0.267
BBB	0.994	65.38	0.062	2.659	32.41	**0.049**	0.476	87.30	0.094	1.707	65.46	0.222
MCD	0.717	75.22	0.023	2.503	42.91	0.151	0.401	88.03	0.023	1.059	63.65	**0.052**
VD	0.702	77.28	**0.028**	2.582	43.10	0.106	0.327	90.76	0.010	2.130	65.48	0.195
ELRG	0.723	76.87	0.065	2.368	42.90	0.099	0.312	90.66	**0.006**	1.088	59.99	**0.018**
VSD	**0.656**	**78.21**	**0.046**	**2.241**	**46.85**	0.112	**0.290**	**91.62**	**0.008**	**1.019**	**67.98**	0.079
D.E	0.872	77.56	0.115	3.402	46.42	0.314	0.319	90.30	**0.008**	2.229	**68.51**	0.241
SWAG	**0.651**	**78.14**	0.059	**1.958**	**49.81**	**0.028**	0.331	90.04	0.031	1.522	**68.41**	0.161

Table 5: Image classification using ResNet18 architecture. Results are averaged over 5 random seeds.

ResNet18	CIFAR10			CIFAR100			SVHN			STL10		
	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE
MAP	0.644	86.34	0.093	2.410	55.38	0.243	0.232	95.32	0.028	1.401	71.26	0.199
BBB	0.697	76.63	0.071	2.239	41.07	0.100	0.218	94.53	0.047	1.290	71.55	0.179
MCD	0.534	87.47	0.084	2.121	59.28	0.227	0.207	95.78	0.026	1.333	72.28	0.188
VD	0.451	87.68	**0.024**	2.888	56.80	0.284	0.164	96.11	0.017	1.084	73.29	0.084
ELRG	**0.382**	87.24	**0.018**	1.634	58.14	**0.096**	0.145	96.03	**0.003**	0.811	73.66	**0.080**
VSD	0.464	87.44	0.061	**1.504**	**60.15**	0.116	**0.140**	**96.41**	**0.003**	**0.769**	**74.50**	**0.083**
D.E	0.488	**88.91**	0.069	1.913	**60.16**	0.203	0.171	**96.36**	0.020	1.197	73.16	0.177
SWAG	**0.330**	**88.77**	0.026	**1.417**	**62.45**	**0.028**	**0.130**	**96.72**	0.016	0.843	73.15	**0.069**

predictive probabilities using negative log-likelihood (NLL), error rate, and expected calibration error (ECE) [61, 23]. Details on experimental settings are available in Appendix I.4.

The synthesis results of this experiment are in Table 3. On MNIST, VMG achieves err. rates of 1.17% and 1.27% with FC 400x2 and FC 750x3 respectively, while SLANG reports 1.72% err. rate with FC 400x2. In general, VSD outperforms consistently other variational methods in most settings. For D.E and SWAG, VSD exhibits competitive results on all three metrics. Especially, the figures on NLL and ECE indicate well-calibrated probabilities in our model. This is also a common but noteworthy behavior in structured approximations. On the other hand, for the remaining methods such as MAP and BBB, the error rates are worse by a large margin compared to VSD (respectively about 9% and 5% on CIFAR10, 4% and 2% on SVHN). On CIFAR10 and SVHN, these two methods and VD all show poor results on both NLL and ECE measures, implying that it will be difficult for them to reason properly about the model uncertainty especially in the out-of-distribution context. For MC Dropout, we observe a pretty good performance with the second-best result in variational methods that is similar to those reported of other works [56, 68, 80]. These results of the Bayesian Dropout methods are competitive with structured methods such as VMG, SLANG, ELRG. This further reinforces our motivation about the potential of Dropout methods for improving the predictive performance.

4.2 Scaling up Bayesian deep convolutional networks

We conduct additional experiments to integrate VSD into large-scale convolutional networks. We reproduce the experiments proposed in [80] (ELRG), in which we trained AlexNet and ResNet18 on 4 datasets CIFAR10, SVHN, CIFAR100 [43], and STIL10 [11] to evaluate the predictive performance of our proposal compared to other methods.

The final results are given in Table 4 and Table 5, in which the top two results will be highlighted in bold. For AlexNet, the performance of VSD is more consistent and higher than other variational methods. Modern deeper architectures facilitate deterministic estimates like MAP to better learn discriminative information extracted from training data but also makes its predictions more confident when picking excessively on unique optima. Therefore, although MAP has comparable predictive accuracy on some settings, it comes at a trade-off with the worst results on both NLL and ECE. Meanwhile, ELRG with a low-rank structure on the variational posterior gains desirable properties on uncertainty metrics. Its performance on NLL and ECE are competitive to that of VSD, however, our method obtains significant improvements on accuracy metric. For the remaining methods, BBB performs poorly on CIFAR10 and CIFAR100 in predictive accuracy, but it still has better performance than MAP in terms of NLL and ECE.

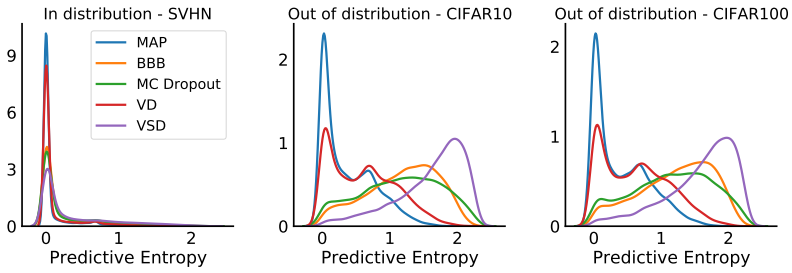


Figure 3: Histograms of predictive entropy for LeNet architecture trained on SVHN dataset.

For ResNet18 architecture, while MAP and BBB still exhibit the same behavior as mentioned above, VSD continues to achieve the convincing results, namely, it has the best performance on CIFAR100 and SVHN over all three metrics when compared to variational-based methods. On CIFAR10 and STL10, VSD also gets competitive statistics on NLL and accuracy. Overall, compared with MCD, VD, and ELRG, our method maintains good performance with better stability.

4.3 Predictive entropy performance

We now evaluate the predictive uncertainty of each model on out-of-distribution settings that have been implemented in previous works [44, 48, 68, 80]. We evaluate the entropy of predictive distribution $p(y^*|x^*, \mathcal{D})$ and use the density of this entropy to assess the quality of uncertainty estimates. Basically, an accurate and well-calibrated model is expected to represent entropy values being concentrated mostly around 0 (i.e. high confidence) when the test data comes from the same underlying distribution as the training data, and in the opposite case, the predictive entropies should be evenly distributed (i.e. higher uncertainty). In fact, the deep learning models do not achieve simultaneously on both expectations at the most ideal, but instead, accurate and well-calibrated ones tend to exhibit a moderate level of confidence on in-distribution data, and then provide a reasonable representation for uncertainty estimates on out-of-distribution data.

For LeNet, we train the model on SVHN dataset and then consider out-of-distribution data from CIFAR10 and CIFAR100. The results are shown in Figure 3. All methods work well on in-distribution data SVHN with the entropy value being distributed mostly around zero. However, the entropy densities of MAP and VD are concentrated excessively. This indicates that these methods would tend to make overconfident predictions on out-of-distribution data. This claim is consolidated by the qualitative results on CIFAR10 and CIFAR100 datasets. In contrast, MCD, BBB, and VSD are well-calibrated with a moderate level of confidence for in-distribution data. On CIFAR10 and CIFAR100 datasets, VSD gains better results with entropy values being distributed over a larger support, meaning that the predictions of VSD are closer to uniform on unseen classes.

We run a similar experiment, in which we train AlexNet on CIFAR10 and use SVHN, CIFAR100 as out-of-distribution data. The results are shown in the top row of Figure 4 in Appendix G.1. While MAP and VD still exhibit the same overconfident phenomenon as on LeNet, we observe the underconfident predictions of BBB and MC Dropout even on in-distribution data, which possibly leads to a high uncertainty on out-of-distribution data. We hypothesize that this is because the models trained with these methods are likely underfit with a low accuracy on the in-distribution training data. In contrast, VSD estimates reasonably the predictive entropy in both settings. The remaining scenario with ResNet18 trained on CIFAR100 is shown in the bottom row of Figure 4 with the same behaviors.

5 Conclusions

We proposed a novel approximate inference framework for Bayesian deep nets, named Variational Structured Dropout (VSD). In VSD, we learn a structured approximate posterior via the Dropout principle. VSD is able to yield a flexible inference while maintaining computational efficiency and scalability for deep convolutional models. The extensive experiments have evidenced the advantages of VSD such as well-calibrated prediction, better generalization, good uncertainty estimation. Given a consistent performance of VSD as presented throughout the paper, an extension of that method to other problems, such as Bayesian active learning or reinforcement learning, is of interest.

Acknowledgements

We would like to thank the anonymous reviewers for their insightful comments and valuable suggestions. We also thank Ngo Trung Nghia (VinAI Research) for helpful discussions throughout this work. Khoat Than was funded by Gia Lam Urban Development and Investment Company Limited, Vingroup, and supported by Vingroup Innovation Foundation (VINIF) under project code VINIF.2019.DA18.

References

[1] A. A. Alemi, B. Poole, I. S. Fischer, J. V. Dillon, R. A. Saurous, and K. P. Murphy. Fixing a broken elbo. In *International Conference on Machine Learning*, 2018.

[2] O. Arenz, M. Zhong, and G. Neumann. Trust-region variational inference with Gaussian mixture models. *Journal of Machine Learning Research*, 21(163):1–60, 2020.

[3] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[4] P. Bartlett, D. J. Foster, and M. Telgarsky. Spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1706.08498*, 2017.

[5] R. v. d. Berg, L. Hasenclever, J. M. Tomczak, and M. Welling. Sylvester normalizing flows for variational inference. *arXiv preprint arXiv:1803.05649*, 2018.

[6] C. H. Bischof and X. Sun. On orthogonal block elimination. *Preprint MCS-P450-0794, Mathematics and Computer Science Division, Argonne National Laboratory*, 1994.

[7] J. Blanchet, Y. Kang, J. L. M. Olea, V. A. Nguyen, and X. Zhang. Machine learning's dropout training is distributionally robust optimal. *arXiv preprint arXiv:2009.06111*, 2020.

[8] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622. PMLR, 2015.

[9] M. Burger and A. Neubauer. Analysis of tikhonov regularization for function approximation by neural networks. *Neural Networks*, 16(1):79–90, 2003.

[10] A. Camuto, M. Willetts, U. Şimşekli, S. Roberts, and C. Holmes. Explicit regularisation in Gaussian noise injections. *arXiv preprint arXiv:2007.07368*, 2020.

[11] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011.

[12] T. Cui, A. Havulinna, P. Marttinen, and S. Kaski. Informative Gaussian scale mixture priors for Bayesian neural networks. *arXiv preprint arXiv:2002.10243*, 2020.

[13] E. Daxberger, E. Nalisnick, J. U. Allingham, J. Antorán, and J. M. Hernández-Lobato. Expressive yet tractable bayesian deep learning via subnetwork inference. *arXiv preprint arXiv:2010.14689*, 2020.

[14] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[15] M. Dusenberry, G. Jerfel, Y. Wen, Y. Ma, J. Snoek, K. Heller, B. Lakshminarayanan, and D. Tran. Efficient and scalable Bayesian neural nets with rank-1 factors. In *International conference on machine learning*, pages 2782–2792. PMLR, 2020.

[16] B. Efron. *Large-scale inference: empirical Bayes methods for estimation, testing, and prediction*, volume 1. Cambridge University Press, 2012.

[17] A. Y. Foong, D. R. Burt, Y. Li, and R. E. Turner. On the expressiveness of approximate inference in Bayesian neural networks. *arXiv preprint arXiv:1909.00719*, 2019.

[18] Y. Gal. Uncertainty in deep learning. *University of Cambridge*, 1(3), 2016.

[19] Y. Gal and Z. Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015.

[20] Y. Gal and Z. Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[21] S. Ghosh, J. Yao, and F. Doshi-Velez. Structured variational learning of bayesian neural networks with horseshoe priors. In *International Conference on Machine Learning*, pages 1744–1753. PMLR, 2018.

[22] A. Graves. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356, 2011.

[23] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

[24] A. K. Gupta and D. K. Nagar. *Matrix variate distributions*, volume 104. CRC Press, 2018.

[25] V. Gupta, T. Koren, and Y. Singer. A unified approach to adaptive regularization in online and stochastic optimization. *arXiv preprint arXiv:1706.06569*, 2017.

[26] D. P. Helmbold and P. M. Long. On the inductive bias of dropout. *The Journal of Machine Learning Research*, 16(1):3403–3454, 2015.

[27] D. P. Helmbold and P. M. Long. Surprising properties of dropout in deep networks. *The Journal of Machine Learning Research*, 18(1):7284–7311, 2017.

[28] J. M. Hernández-Lobato and R. Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.

[29] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[30] G. E. Hinton and D. Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13, 1993.

[31] J. Hron, A. Matthews, and Z. Ghahramani. Variational Bayesian dropout: pitfalls and fixes. In *International Conference on Machine Learning*, pages 2019–2028. PMLR, 2018.

[32] P. Izmailov, W. J. Maddox, P. Kirichenko, T. Garipov, D. Vetrov, and A. G. Wilson. Subspace inference for bayesian deep learning. In *Uncertainty in Artificial Intelligence*, pages 1169–1179. PMLR, 2020.

[33] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[34] N. S. Keskar and R. Socher. Improving generalization performance by switching from adam to sgd. *arXiv preprint arXiv:1712.07628*, 2017.

[35] M. Khan, D. Nielsen, V. Tangkaratt, W. Lin, Y. Gal, and A. Srivastava. Fast and scalable Bayesian deep learning by weight-perturbation in adam. In *International Conference on Machine Learning*, pages 2611–2620. PMLR, 2018.

[36] V. Kharitonov, D. Molchanov, and D. Vetrov. Variational dropout via empirical Bayes. *arXiv preprint arXiv:1811.00596*, 2018.

[37] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[38] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29:4743–4751, 2016.

[39] D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. In *Advances in neural information processing systems*, pages 2575–2583, 2015.

[40] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[41] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[42] R. Krishnan, M. Subedar, and O. Tickoo. Specifying weight priors in bayesian deep neural networks with empirical bayes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4477–4484, 2020.

[43] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[44] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *arXiv preprint arXiv:1612.01474*, 2016.

[45] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[46] C. Louizos, K. Ullrich, and M. Welling. Bayesian compression for deep learning. *arXiv preprint arXiv:1705.08665*, 2017.

[47] C. Louizos and M. Welling. Structured and efficient variational deep learning with matrix Gaussian posteriors. In *International Conference on Machine Learning*, pages 1708–1716, 2016.

[48] C. Louizos and M. Welling. Multiplicative normalizing flows for variational Bayesian neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2218–2227. JMLR. org, 2017.

[49] D. J. MacKay. A practical Bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

[50] W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson. A simple baseline for Bayesian uncertainty in deep learning. *Advances in Neural Information Processing Systems*, 32:13153–13164, 2019.

[51] S.-i. Maeda. A Bayesian encourages dropout. *arXiv preprint arXiv:1412.7003*, 2014.

[52] D. McAllester. A pac-Bayesian tutorial with a dropout bound. *arXiv preprint arXiv:1307.2118*, 2013.

[53] P. Mianjy and R. Arora. On dropout and nuclear norm regularization. In *International Conference on Machine Learning*, pages 4575–4584. PMLR, 2019.

[54] P. Mianjy and R. Arora. On convergence and generalization of dropout training. *Advances in Neural Information Processing Systems*, 33, 2020.

[55] P. Mianjy, R. Arora, and R. Vidal. On the implicit bias of dropout. In *International Conference on Machine Learning*, pages 3537–3545, 2018.

[56] A. Mishkin, F. Kunstner, D. Nielsen, M. Schmidt, and M. E. Khan. Slang: Fast structured covariance approximations for Bayesian deep learning with natural gradient. In *Advances in Neural Information Processing Systems*, pages 6245–6255, 2018.

[57] D. Molchanov, A. Ashukha, and D. Vetrov. Variational dropout sparsifies deep neural networks. In *International Conference on Machine Learning*, pages 2498–2507. PMLR, 2017.

[58] D. Molchanov, V. Kharitonov, A. Sobolev, and D. Vetrov. Doubly semi-implicit variational inference. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2593–2602, 2019.

[59] W. Mou, Y. Zhou, J. Gao, and L. Wang. Dropout training, data-dependent regularization, and generalization bounds. In *International Conference on Machine Learning*, pages 3645–3653, 2018.

[60] J. Mukhoti, P. Stenetorp, and Y. Gal. On the importance of strong baselines in Bayesian deep learning. *arXiv preprint arXiv:1811.09385*, 2018.

[61] M. P. Naeini, G. Cooper, and M. Hauskrecht. Obtaining well calibrated probabilities using Bayesian binning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[62] E. Nalisnick, J. M. Hernández-Lobato, and P. Smyth. Dropout as a structured shrinkage prior. In *International Conference on Machine Learning*, pages 4712–4722. PMLR, 2019.

[63] R. M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.

[64] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[65] B. Neyshabur, S. Bhojanapalli, and N. Srebro. A pac-Bayesian approach to spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1707.09564*, 2017.

[66] C. Oh, K. Adamczewski, and M. Park. Radial and directional posteriors for Bayesian neural networks. *arXiv preprint arXiv:1902.02603*, 2019.

[67] V. M.-H. Ong, D. J. Nott, and M. S. Smith. Gaussian variational approximation with a factor covariance structure. *Journal of Computational and Graphical Statistics*, 27(3):465–478, 2018.

[68] K. Osawa, S. Swaroop, A. Jain, R. Eschenhagen, R. E. Turner, R. Yokota, and M. E. Khan. Practical deep learning with Bayesian principles. *arXiv preprint arXiv:1906.02506*, 2019.

[69] R. Ranganath, D. Tran, and D. Blei. Hierarchical variational models. In *International Conference on Machine Learning*, pages 324–333, 2016.

[70] D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pages 1530–1538. PMLR, 2015.

[71] H. Ritter, A. Botev, and D. Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018.

[72] S. Rossi, S. Marmin, and M. Filippone. Walsh-hadamard variational inference for Bayesian deep learning. *arXiv preprint arXiv:1905.11248*, 2019.

[73] L. Sagun, U. Evci, V. U. Guney, Y. Dauphin, and L. Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*, 2017.

[74] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[75] S. Sun, C. Chen, and L. Carin. Learning structured weight uncertainty in Bayesian neural networks. In *Artificial Intelligence and Statistics*, pages 1283–1292, 2017.

[76] X. Sun and C. Bischof. A basis-kernel representation of orthogonal matrices. *SIAM journal on matrix analysis and applications*, 16(4):1184–1196, 1995.

[77] J. Swiatkowski, K. Roth, B. Veeling, L. Tran, J. Dillon, J. Snoek, S. Mandt, T. Salimans, R. Jenatton, and S. Nowozin. The k-tied normal distribution: A compact parameterization of Gaussian mean field posteriors in Bayesian neural networks. In *International Conference on Machine Learning*, pages 9289–9299. PMLR, 2020.

[78] M. Teye, H. Azizpour, and K. Smith. Bayesian uncertainty estimation for batch normalized deep networks. In *International Conference on Machine Learning*, pages 4907–4916. PMLR, 2018.

[79] J. M. Tomczak and M. Welling. Improving variational auto-encoders using convex combination linear inverse autoregressive flow. *arXiv preprint arXiv:1706.02326*, 2017.

[80] M. Tomczak, S. Swaroop, and R. Turner. Efficient low rank Gaussian variational inference for neural networks. *Advances in Neural Information Processing Systems*, 33, 2020.

[81] S. Wager, S. Wang, and P. S. Liang. Dropout training as adaptive regularization. In *Advances in Neural Information Processing Systems*, pages 351–359, 2013.

[82] S. Wang and C. Manning. Fast dropout training. In *international conference on machine learning*, pages 118–126. PMLR, 2013.

[83] C. Wei, S. Kakade, and T. Ma. The implicit and explicit regularization effects of dropout. In *International Conference on Machine Learning*, pages 10181–10192. PMLR, 2020.

[84] F. Wenzel, K. Roth, B. S. Veeling, J. Świątkowski, L. Tran, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, and S. Nowozin. How good is the Bayes posterior in deep neural networks really? *arXiv preprint arXiv:2002.02405*, 2020.

[85] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht. The marginal value of adaptive gradient methods in machine learning. *arXiv preprint arXiv:1705.08292*, 2017.

[86] A. G. Wilson and P. Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *arXiv preprint arXiv:2002.08791*, 2020.

[87] A. Wu, S. Nowozin, E. Meeds, R. E. Turner, J. M. Hernandez-Lobato, and A. L. Gaunt. Deterministic variational inference for robust bayesian neural networks. *arXiv preprint arXiv:1810.03958*, 2018.

[88] M. Yin and M. Zhou. Semi-implicit variational inference. In *International Conference on Machine Learning*, pages 5660–5669. PMLR, 2018.

[89] G. Zhang, S. Sun, D. Duvenaud, and R. Grosse. Noisy natural gradient as variational inference. In *International Conference on Machine Learning*, pages 5852–5861. PMLR, 2018.

[90] J. Zhang, Q. Lei, and I. Dhillon. Stabilizing gradients for deep neural networks via efficient svd parameterization. In *International Conference on Machine Learning*, pages 5806–5814. PMLR, 2018.

[91] H. Zhao, Y.-H. H. Tsai, R. Salakhutdinov, and G. J. Gordon. Learning neural networks with adaptive regularization. *arXiv preprint arXiv:1907.06288*, 2019.

[92] S. Zhao, J. Song, and S. Ermon. Infovae: Balancing learning and inference in variational autoencoders. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5885–5892, 2019.

[93] P. Zhou, J. Feng, C. Ma, C. Xiong, S. Hoi, et al. Towards theoretically understanding why sgd generalizes better than adam in deep learning. *arXiv preprint arXiv:2010.05627*, 2020.

[94] O. Zobay et al. Variational Bayesian inference with Gaussian-mixture approximations. *Electronic Journal of Statistics*, 8(1):355–389, 2014.

Supplement to "Structured Dropout Variational Inference for Bayesian Neural Networks"

In this supplementary material, we collect proofs and remaining materials that were deferred from the main paper. In Appendix A, we analyze the expressiveness of Variational Structured Dropout (VSD) through the approximate posterior structure and the parameterization of prior hierarchy. In Appendix B, we provide proof for the KL condition in VSD. In Appendix C, we derive in details the variational objective of VSD with hierarchical prior. Details of computational complexity of different Bayesian methods are in Appendix D. In Appendix E, we present the explicit regularization of VSD and then provide several theoretical insights induced by our regularizer. In Appendix F, we provide a new and complementary Bayesian justification for Variational Dropout methods. The next Appendices are for additional experiments of VSD. Finally, we give further discussions and investigations of VSD compared to non-variational methods.

A The expressiveness of Variational Structured Dropout

A.1 The fully correlated structure of approximate posterior

A essential question is how expressive the Dropout posterior in VSD is. In VSD framework, we inject a structured noise $\xi^{(t)}$ into the deterministic weight Θ, then obtain a random weight $\mathbf{W}^{(t)} = \text{diag}(\xi^{(t)})\Theta$ and an induced posterior $q_t(\mathbf{W})$. Because each scalar noise $\xi_i^{(t)}$ is shared across the row $\mathbf{W}_{i:}$ respectively, it results in a correlation on each row of $\mathbf{W}$. Meanwhile, the marginal distribution on each column $\mathbf{W}_{j:}$ is a Gaussian distribution with full covariance matrix: $q_t(\mathbf{W}_{:j}) = \mathcal{N}(\Theta_{:j}, \text{diag}(\Theta_{:j})\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\Theta_{:j}))$. Therefore, the Dropout posterior in VSD has a fully correlated structure.

The correlation structure of VSD has a natural interpretation which bridges our approach to other structured approximations. Indeed, a full correlation over the whole random matrix can be parameterized separately into the correlations among the rows and columns of that matrix, which implicitly affects the correlations among the input and output hidden units. Interestingly, this connection can be exhibited explicitly in our method. When performing the forward pass, Dropout procedure will generally introduce the correlations between elements at pre-activation layer, namely output hidden units. In addition, the structured noise in our method can be considered as an auxiliary variable, which has the ability to captures the correlations among neurons of input hidden units, or otherwise, it will encourage neurons to borrow statistical strength from one another through variational learning. On the other hand, we know that due to the statistical noise, the correlations among hidden units appear naturally, and this therefore rationalizes the original proposal in our paper.

A.2 The role of hierarchical prior

In Bayesian inference, the prior distribution plays an important role in representing the capacity of Bayesian neural networks. By appropriately employing the informative priors, we can significantly improve the predictive performance [84, 15]. This distribution also allows incorporating external domain knowledge or specific properties such as feature sparsity, into the Bayesian deep models [12]. We derive here some detailed discussions about the role of hierarchical prior particularly in our method.

Gaussian scale mixture prior and mixture approximate posterior. The well-known property of expanding a model hierarchically is that it induces new dependencies between the data, either through shrinkage or an explicitly correlated prior [16]. The hierarchical representation in our method is a *center parameterization*, and by integrating out the latent variable $\mathbf{z}$, we obtain a marginal prior distribution as follows:

$$p(\mathbf{W}_{:j}|\beta_{:j}, \tau) = \int \mathcal{N}(\mathbf{W}_{:j}|0, \text{diag}(\mathbf{z} \odot \beta_{:j}^{-1})p(\mathbf{z}|\tau)d\mathbf{z}, \tag{7}$$

where $p(\mathbf{z}|\tau)$ is treated as the mixing distribution. The equation (7) can also be written in an equivalent *expanded* or *non-centered parameterization* as: $\mathbf{W}_{:j} = \gamma \odot \mathbf{z}$ with $\gamma \sim \mathcal{N}(0, \text{diag}(\beta_{:j}^{-1}))$ and $\mathbf{z} \sim p(\mathbf{z}|\tau)$. This prior family have been widely used in BNN literature [46, 21, 66, 15]. By

approximating the above integral with Monte Carlo sampling $\mathbf{z}^{(i)} \sim p(\mathbf{z}|\eta)$, we can resemble an informative prior known as Gaussian scale mixtures (GSM) [8, 12] with the following term:

$$p(\mathbf{W}_{:j}|\beta_{:j}, \tau) \approx \frac{1}{M}\sum_{i=1}^{M} \mathcal{N}(\mathbf{W}_{:j}|0, \text{diag}(\mathbf{z}^{(i)} \odot \beta_{:j}^{-1})). \tag{8}$$

Therefore, our hierarchical framework can provide an appealing connection between multiplicative Gaussian noise with GSM priors. This interpretation is close to the recent work in [62], in which the authors show that multiplicative noise is equivalent to the structured shrinkage prior and interestingly, MC Dropout objective is a lower bound on the scale mixture model's marginal MAP objective.

Instead of investigating extensively the expressiveness of this prior through different parameterizations, in the scope of this work, we focus on the advantages of the joint inference that can make a mixture approximation in the variational objective function:

$$\mathbb{E}_{q_\psi(\mathbf{z})q_t(\mathbf{W}|\mathbf{z})} \log p(\mathcal{D}|\mathbf{W}) \approx \frac{1}{M}\sum_{i=1}^{M} \mathbb{E}_{q_t(\mathbf{W}|\mathbf{z}^{(i)})} \log p(\mathcal{D}|\mathbf{W}).$$

This approximation is equivalent to leveraging a mixture of structured covariance posterior, which has a reasonable potential to recover the multimodality of the true Bayesian posterior. Moreover, this mixture is also practical in the high dimensional setting of Bayesian deep models, because it does not require the additional computational cost from multiplying the components. We also suggest that hierarchical parameterization with joint inference is a promising approach to improve the predictive performance of Bayesian deep models, especially compared with non-Bayesian methods such as Deep Ensemble, which is evidenced by a recent work in [15].

Hierarchical prior imposes a global stochastic noise and enforces a stronger regularization. At each iteration of the training process, while we draw $\mathbf{W}$ from the conditional posterior $q_t(\mathbf{W}|\mathbf{z})$ separately for each data as the Dropout procedure, the latent $\mathbf{z} \sim q_\psi(\mathbf{z})$ is shared across the entire data batch. This means for each data input $\mathbf{x}_n$, we introduce a joint-structured noise $\widehat{\xi}_n^{(t)}$ with the following form:

$$\begin{aligned} &\mathbf{z} \sim q_\psi(\mathbf{z}), \quad \xi_n^{(t)} \sim \mathcal{N}(\mathbf{1}_K, \mathbf{U}\text{diag}(\alpha)\mathbf{U}^T), \\ &\widehat{\xi}_n^{(t)} = \mathbf{z} \odot \xi_n^{(t)}, \quad \mathbf{W} = \text{diag}(\widehat{\xi}_n^{(t)})\Theta. \end{aligned} \tag{9}$$

The above reinterpretation demonstrates that by the Monte Carlo estimation, the joint variational inference with hierarchical prior in our method adapts to the vanilla Dropout procedure. The new representation of Dropout noise allows our method to regularize each unit layer with different levels of stochasticity. The latent $\mathbf{z}$ under this representation can be considered as a global variational noise and by learning its variational distribution $q_\psi(\mathbf{z})$, we can capture correlation characteristics of input samples in each data batch. Furthermore, in our prior hierarchy, whilst the hyperparameter β has the same size with the network weight $\mathbf{W}$ that might induce a relatively poor regularization, the latent $\mathbf{z}$ is designed with the size of the number of rows and is shared across columns of the matrix $\mathbf{W}$. This row-partitioning will discourage allowing too many degrees of freedom in the parameterization. Basically, such a technique applied to variational Bayesian inference will enforce a stronger regularization in the objective and then prevent the model from the overfitting issue.

B The KL condition in Variational Structured Dropout

The alternative variational objective of Variational Structured Dropout (VSD) is given as follows:

$$\begin{aligned} \mathcal{L}(\alpha, \Theta, .) &= \mathbb{E}_{q_\phi(\mathbf{W})} \log p(\mathcal{D}|\mathbf{W}^{(t)}) - \mathbb{D}_{KL}(q_t^\star(\mathbf{W})||p(\mathbf{W})) \\ &= \mathbb{E}_{q_\alpha(\xi)} \log p(\mathcal{D}|\Theta, \xi^{(t)}) - \sum_{j=1}^{Q} \mathbb{D}_{KL}(q_t(\mathbf{W}_{:j})||p(\mathbf{W}_{:j})). \end{aligned} \tag{10}$$

By applying Monte Carlo sampling combined with the reparameterization trick to approximate the expected log-likelihood, we perform a procedure being equivalent to injecting a structured noise $\xi^{(t)}$ into the model parameters Θ. However, to make this Monte Carlo estimation follows the Dropout

training procedure, we separately draw one realization $\xi_n^{(t)}$ for each data point $(\mathbf{x}_n, \mathbf{y}_n)$. This can be interpreted as arising from the local reparameterization trick [39], in which the global parameter uncertainty is translated backward into local unit uncertainty at the pre-linear layer instead of the post-linear one. Finally, to ensure the KL condition, we will specify the KL term in the form independent of Θ. Then, the above lower bound recovers the Dropout objective function.

We have the Dropout posterior and the prior determined on the j-th column of $\mathbf{W}$, respectively:

$$q_t(\mathbf{W}_{:j}) = \mathcal{N}(\Theta_{:j}, \text{diag}(\Theta_{:j})\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\Theta_{:j})) \quad \text{and} \quad p(\mathbf{W}_{:j}|\beta) = \mathcal{N}(0, \text{diag}(\beta_{:j}^{-1})).$$

Let $\boldsymbol{\mu}_1 = \Theta_{:j}$, $\boldsymbol{\Sigma}_1 = \text{diag}(\Theta_{:j})\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\Theta_{:j})$ and $\boldsymbol{\mu}_2 = 0$, $\boldsymbol{\Sigma}_2 = \text{diag}(\beta_{:j}^{-1})$. The KL divergence can then be calculated as follows:

$$\mathbb{D}_{KL}(q_t(\mathbf{W}_{:j})||p(\mathbf{W}_{:j})) = \frac{1}{2}\bigg[\log\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|} - K + \text{Trace}(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1) + (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)\bigg].$$

Since $\mathbf{U}$ is a orthogonal matrix, we have:

$$\log\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|} = -\sum_{i=1}^{K}\log\beta_{ij} - \sum_{i=1}^{K}\log\alpha_i\Theta_{ij}^2,$$

$$\text{Trace}(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1) = \text{Trace}(\text{diag}(\beta_{:j}\odot\Theta_{:j}^2)\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T) = \sum_{i=1}^{K}\beta_{ij}\Theta_{ij}^2\left(\sum_{j=1}^{K}\alpha_j U_{ij}^2\right),$$

$$(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^T\boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1) = \Theta_{:j}^T\text{diag}(\beta_{:j})\Theta_{:j} = \sum_{i=1}^{K}\beta_{ij}\Theta_{ij}^2.$$

Given the above equations, the KL term on the j-th column of $\mathbf{W}$ can be rewritten by:

$$\mathbb{D}_{KL}(q_t(\mathbf{W}_{:j})||p(\mathbf{W}_{:j})) = \frac{1}{2}\sum_{i=1}^{K}\bigg[-\log\beta_{ij} - \log\alpha_i\Theta_{ij}^2 - 1 + \beta_{ij}\Theta_{ij}^2\left(1 + \sum_{j=1}^{K}\alpha_j U_{ij}^2\right)\bigg]. \tag{11}$$

We can find that the orthogonality of Householder transformations facilitates the tractable calculation for the KL term without complicated analyses. Next, we choose the prior hyper-parameter β via the Empirical Bayes approach which is achieved by optimizing β based upon the data. More specifically, taking the partial derivative of the RHS in equation (11) with respect to β, we get:

$$\frac{\partial\mathbb{D}_{KL}}{\partial\beta_{ij}} = \frac{1}{2}\bigg[-\frac{1}{\beta_{ij}} + \Theta_{ij}^2\left(1 + \sum_{j=1}^{K}\alpha_j U_{ij}^2\right)\bigg].$$

Letting this derivative to zero, we obtain the optimal value for β in the analytical form: $\beta_{ij}^* = 1/\left(\Theta_{ij}^2(1 + \sum_{j=1}^{K}\alpha_j U_{ij}^2)\right)$. Substitute this value in the expression of the KL term, we get the form independent of the weight parameter Θ as follows:

$$\mathbb{D}_{KL}^{EB}(q_t(\mathbf{W}_{:j})||p(\mathbf{W}_{:j})) = \frac{1}{2}\sum_{i=1}^{K}\log\frac{1 + \sum_{j=1}^{K}\alpha_j U_{ij}^2}{\alpha_i}. \tag{12}$$

As a consequence, we obtain the conclusion in the main text about the KL condition in VSD.

Discussion of the effect of the Empirical Bayes in our method. The Empirical Bayes procedure presented above is equivalent to an iterative optimization algorithm for our objective, in which β will be updated until convergence after every single update of other parameters. However, disappearing this precision parameter by directly substituting its Empirical Bayes values would help to clarify the KL-condition guarantee in VSD as analyzed. The data-dependent choice of this parameter is made explicitly through the dependence on the learned droprate α, the matrix U, and the deterministic weight Θ. In the literature, several studies have also offered interesting perspectives for the Empirical Bayes. Generally, this procedure usually suffers from the main criticism of using data twice that is

illegal in a strict Bayesian formalism. And particularly in mean-field BNNs, the Empirical Bayes was claimed to be able to yield slow convergence, introduce strange local minima and thus lead to miscalibrated predictive distributions [8]. However, there should be a more comprehensive study to investigate the effects of this procedure, especially in a complicated context of deep learning models, which are surrounded by other data-related techniques such as temperature scaling, data augmentation, and even Dropout - a data-dependent regularization. On the other hand, Empirical Bayes has been embraced and widely adopted in Bayesian machine learning, and especially in the seminal work on Bayesian neural nets. This technique has been employed as a principled approach to learning prior hyperparameters [87, 42], or embodying automatic-relevance determination [36].

C The derivation of the variational objective with hierarchical prior

In Dropout approximate inference, the KL condition restricts the scope of prior distribution family. Our works has overcome this limitation by proposing a unified framework using varational structured Dropout combined with hierarchical prior, in which we guarantees the KL condition without any simplifying assumptions about the prior family. Concretely, with our proposed prior hierarchy, we maximize a variational lower bound from the joint variational inference as follows:

$$\begin{aligned}\mathcal{L}(\alpha, \Theta, \psi, .) &= \mathbb{E}_{q_\psi(\mathbf{z})q_t(\mathbf{W}|\mathbf{z})} \log p(\mathcal{D}|\mathbf{W}) \\ &\quad - \mathbb{E}_{q_\psi(\mathbf{z})}(\mathbb{D}_{KL}(q_t^\star(\mathbf{W}|\mathbf{z}, \phi)||p(\mathbf{W}|\mathbf{z}, \beta)) - \mathbb{D}_{KL}(q_\psi(\mathbf{z})||p(\mathbf{z})). \end{aligned} \tag{13}$$

For the latent variable $\mathbf{z}$, we choose the prior $p(\mathbf{z}|\eta)$ and the variational distribtuion $q(\mathbf{z})$ as (inverse) Gamma(a, b) and log-Normal(γ, δ) distribution respectively. These distributions have positive support and can be reparametrized. The KL-divergence between them also has a closed-form expression, which is given by:

$$\mathbb{D}_{KL}(q_\psi(\mathbf{z})\|p(\mathbf{z})) = a\log b - \log\Gamma(a) - a\gamma - \beta\exp(-\gamma + 0.5\delta) + 0.5(\log\delta + 1 + \log(2\pi)).$$

For the KL-divergence between the conditional posterior $q_t^\star(\mathbf{W}|\mathbf{z}, \phi)$ and the conditional prior $p(\mathbf{W}|\mathbf{z}, \beta)$, similarly, we need a form that does not depend on the deterministic weight Θ. Since:

$$q_t(\mathbf{W}_{:j}|\mathbf{z}) = \mathcal{N}(\mathbf{z}\odot\Theta_{:j}, \mathbf{V}_j\mathbf{U}\text{diag}(\alpha)(\mathbf{V}_j\mathbf{U})^T) \;\text{ and }\; p(\mathbf{W}_{:j}|\mathbf{z}, \beta_{:j}) = \mathcal{N}(0, \text{diag}(\mathbf{z}\odot\beta_{:j}^{-1}))$$

where $\mathbf{V}_j = \text{diag}(\mathbf{z}\odot\Theta_{:j})$, similar to the analysis in the previous section, we have:

$$\begin{aligned}&\mathbb{D}_{KL}(q_t^\star(\mathbf{W}_{:j}|\mathbf{z}, \phi)||p(\mathbf{W}_{:j}|\mathbf{z}, \beta_{:j})) = \\ &\qquad \frac{1}{2}\sum_{i=1}^{K}\Bigg[-\log z_i - 1 - \log\beta_{ij} - \log\alpha_i\Theta_{ij}^2 + \beta_{ij}z_i\Theta_{ij}^2\Bigg(1 + \sum_{j=1}^{K}\alpha_j U_{ij}^2\Bigg)\Bigg].\end{aligned}$$

Because β is referred to as the scaling factor, we can choose it by: $\beta_{ij}^* = 1/\Bigg(\Theta_{ij}^2(1+\sum_{j=1}^{K}\alpha_j U_{ij}^2)\Bigg)$.

The above KL then can be rewritten in the following form:

$$\mathbb{D}_{KL}^{EB}(q_t^\star(\mathbf{W}_{:j}|\mathbf{z}, \phi)||p(\mathbf{W}_{:j}|\mathbf{z}, \beta_{:j})) = \frac{1}{2}\sum_{i=1}^{K}\Bigg[z_i - \log z_i - 1 - \log\frac{1+\sum_{j=1}^{K}\alpha_j U_{ij}^2}{\alpha_i}\Bigg]. \tag{14}$$

As a result, this form satisfies the KL condition.

D Computational complexity and low-rank approximation

We describe here in detail the computational complexity of the different algorithms, in which the computation is considered when performing a forward pass through a single layer of the network. We also discuss the memory usage while constructing the dynamic computation graph. To ease the presentation, we briefly recall the abbreviations of methods from the main text, in particular: Bayes by Backprop (BBB) [8], Variational Matrix Gaussian (VMG) [47], Multiplicative Normalizing Flow (MNF) [48], low-rank approximations (SLANG, ELRG) [56, 80]; and the Bayesian Dropout methods including MC Dropout (MCD) [19, 20], Variational Dropout (VD) [39].

Computational complexity. Assume the weight matrix of the layer is of size $K \times L$, in which K

Table 6: Computational complexity per layer of MAP and different variational methods.

Method	Time	Memory
MAP	$\mathcal{O}(KL\|\mathcal{B}\|)$	$\mathcal{O}(L\|\mathcal{B}\|)$
BBB	$\mathcal{O}(sKL\|\mathcal{B}\|)$	$\mathcal{O}(sKL + L\|\mathcal{B}\|)$
BBB-LTR	$\mathcal{O}(2KL\|\mathcal{B}\|)$	$\mathcal{O}(2L\|\mathcal{B}\|)$
VMG	$\mathcal{O}(m^3 + 2KL\|\mathcal{B}\|)$	$\mathcal{O}(KL\|\mathcal{B}\|)$
SLANG	$\mathcal{O}(r^2KL + rsKL\|\mathcal{B}\|)$	$\mathcal{O}(rKL + sKL\|\mathcal{B}\|)$
ELRG	$\mathcal{O}(r^3 + (r+2)KL\|\mathcal{B}\|)$	$\mathcal{O}((r+2)L\|\mathcal{B}\|)$
VSD	$\mathcal{O}(K^2 + KL\|\mathcal{B}\|)$	$\mathcal{O}(K^2 + K\|\mathcal{B}\|)$
VSD-low rank	$\mathcal{O}(rK + KL\|\mathcal{B}\|)$	$\mathcal{O}(K^2 + K\|\mathcal{B}\|)$

Table 7: Computation time of variational methods compared to standard MAP (1x).

Methods	**Time/epoch** (s)		
	LeNet5	AlexNet	ResNet18
BBB-LTR	1.53x	1.75x	3.28x
MNF	2.86x	3.40x	4.88x
VD	1.18x	1.15x	1.32x
VSD $T = 1$	1.25x	1.32x	1.86x
VSD $T = 2$	1.35x	1.49x	2.90x
time-scaling	1.08	1.13	1.56

denotes the number of rows in fully connected layer or the number of channels in convolutional layer, respectively. First, MAP estimation performs a matrix multiplication with time cost $K \times L$ to forward each input $\mathbf{x}_i$ of size K in data batch $\mathcal{B}$. MAP estimation needs to store the output of these calculations which gives a memory cost $L|\mathcal{B}|$. Next, BBB with naive reparameterization trick, in practice, needs to use $s \geq 2$ sampled weights of dimension $K \times L$ to reduce the variance of gradient estimator. This makes the computation hard to be performed in parallel, thus incurs multiple costs of both time and memory with $\mathcal{O}(sKL|\mathcal{B}|)$ and $\mathcal{O}(sKL + L|\mathcal{B}|)$ respectively. On the other hand, with the local reparameterization trick that translates uncertainty about global random weights into local noise in pre-activation unit, BBB can gain an alternative unbiased estimator with low variance while maintaining low complexity via sampling only a local noise. However, it requires two forward passes to obtain means and variances of the pre-activation. For VMF, SLANG, and ELRG, the detailed analysis can be found on the original papers, and note that SLANG is a method that fails to employ the local reparameterization trick, thereby leading to very high complexity on both time and memory.

VSD adopts the advantage of Dropout training (VD) via just sampling the low dimension noise instead of whole random weights. An additional benefit is that VSD only requires one forward pass in parallel compared with two steps of the local reparameterization trick. When using a fully connected layer (FC) size of $K \times K$ to parameterize the Householder vector, namely:

$$\mathbf{v}_t = \mathbf{FC}(\mathbf{v}_{t-1}), \qquad \mathbf{S}_t = \left(\mathbf{I} - 2\frac{\mathbf{v}_t\mathbf{v}_t^T}{\|\mathbf{v}_t\|_2^2}\right)\mathbf{S}_{t-1} = \mathbf{H}_t\mathbf{S}_{t-1},$$

for $t = 1, ..., T$, it will induce a complexity of $\mathcal{O}(K^2)$ to our method in terms of both time and memory cost. However, we also reduce the number of parameters of this FC by adding a low dimensional hidden layer. This simple solution results in lower computational time of $\mathcal{O}(rK + KL|\mathcal{B}|)$ without sacrificing much the performance (see Table 8 in Appendix D). In general, VSD has shown better computational efficiency than other structured approximation methods, even more practical than the mean-field BBNs.

Note that, taking the advantage of low dimensional space to enrich the quality of approximation is an appealing idea. The recent advances in variational inference have offered many modern techniques to exploit this idea, such as normalizing flow [70, 38, 5], auxiliary random variable, implicit distribution [69, 88], or mixture approximation [94, 2]. Nevertheless, the novelty depends on the sophistication when applied to specific models with certain constraints. More specifically, in the context of the problem we aim for, applying these above techniques to Bayesian Dropout frameworks requires dealing with some challenges including the difficulty of parallel backpropagation, high computational complexity, and more importantly, how to ensure the KL condition. The Householder parameterization helps our approach address these challenges in both theory and applicability. Moreover, we can extend our method to other parameterizations, for example Sylvester-based flows [5], as long as the orthogonality of matrix $\mathbf{U}$ is preserved.

Practical runtime. We show in Table 7 the empirical computation time of VSD and some other methods, in which BBB-LTR and MNF are *direct* approximatitons of BNNs, while VD and VSD represent Dropout inference frameworks. BBB-LTR and VD maintain a mean-field structure for the approximation, while MNF and VSD share the same intuition of enriching the variational approximate distribution via low dimensional space. However, we remark that there are some key considerations that distinguish VSD from MNF including: (1) MNF facilitates flexible approximation via normalizing flows (NFs) which is much more expensive compared with the orthogonal parameterizations of VSD, MNF even used two sequences of NFs to tighten the variational lower bound; (2) VSD exploits

Table 8: The performance of VSD when using low-rank approximation, where r is the dimension of hidden unit, $T = 2$ is the number of Householder transformations. Random seed $= 1$. For all metrics, lower is better.

$T = 2$	**MNIST**			**CIFAR10**			**SVHN**		
	FC 750x3			CNN 32x64x128					
	NLL	err. rate	ECE	NLL	err. rate	ECE	NLL	err. rate	ECE
$r = 2$	0.049	**1.12**	0.007	0.7298	25.45	**0.022**	**0.3007**	**8.36**	0.008
$r = 5$	0.046	1.15	0.006	**0.7199**	**24.91**	0.023	0.3024	**8.36**	0.009
$r = 10$	0.049	1.15	0.007	0.7365	25.24	0.024	0.3048	8.47	0.009
full rank	0.045	1.13	0.006	0.7297	25.18	0.023	0.3021	8.41	0.008

Bayesian hierarchical modeling for the Dropout inference framework and then learns a joint posterior, while MNF adopts a implicit-marginal distribution for approximate weight posterior. The settings we use to implement MNF are given in the original paper [48]. Going back to Table 7, VSD with $T = 2$ exhibits extra computation time compared to $T = 1$, the increase on ResNet18 is more evident than on LeNet and AlexNet (see the time-scaling values). This is because the quantities K of ResNet18 are larger than that of LeNet and AlexNet. However, on more modern architectures such as PreResNet110 which prefers to evolve in depth rather than width (namely using fewer channels), VSD with $T = 2$ should not endure much extra computation time and thus would make a good adaptation. Indeed, we verify this argument by measuring the computation time of VSD trained with PreResNet110 on CIFAR10 dataset, from which the figures obtained for $T = 1$ and $T = 2$ are 2.68x and 3.60x, then the corresponding time-scaling value is 1.34.

Low-rank approximation for VSD. We investigate a *low-rank* structure in the fully connected layer used to parameterize the Householder vectors in our method. Instead of using one layer with full size $K \times K$, we add a low dimensional hidden layer with ReLU activation. The size of this hidden layer is $r \in \{2, 5, 10\}$. This idea is quite natural because it reduces significantly the number of parameters in our method while ensuring flexible parametrization for the Householder vectors thanks to the nonlinearity in hidden activation.

We show the performance of VSD with low-rank approximation in Table 8, where we repeat the experiment on image classification in Section 4.1 of our main paper. We can see that although the rank r is very small, the decrease in performance is negligible (still outperforms the baselines). This natural idea even improves the results on some settings such as the ECE metric in SVHN dataset.

E The derivation of the induced regularization of VSD

In this appendix, we present the explicit regularization induced by the structured noise in our framework. Let $\mathcal{X} \subseteq \mathbb{R}^{d_{\text{in}}}$ and $\mathcal{Y} \subseteq \mathbb{R}^{d_{\text{out}}}$ denote the input and output spaces, respectively. We consider a deep linear neural net $f : \mathcal{X} \to \mathbb{R}^{d_{\text{out}}}$ with L layers parameterized by $\{\Theta^{(i)}\}_{i=1}^{L}$, and then define a corresponding surrogate loss $\ell : \mathbb{R}^{d_{\text{out}}} \times \mathcal{Y} \to \mathbb{R}$. The goal of learning is to find a hypothesis f that minimizes the population risk: $L_0 := \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{X}\times\mathcal{Y}}\ell(f(\mathbf{x}), \mathbf{y})$. The expected term can be written when we instead optimize the empirical risk using data batch $\mathcal{B}$ as follows: $\widehat{L} := \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{B}}\ell(f(\mathbf{x}), \mathbf{y})$. For convenience, we will ignore output $\mathbf{y}$ in the subsequent analyses.

Note that, while the overall function is linear, the representation in factored form makes the optimization landscape non-convex and hence, challenging to analyze. For simplicity, we start by considering Dropout applied to a single layer i of the network and define some detailed notations as: $\mathbf{h}_i$ is the i-th hidden layer; $f_i(.)$ denotes the composition of the layers after $\mathbf{h}_i$, that means $f_i(\mathbf{h}_i(\mathbf{x})) = f(\mathbf{x})$; $\mathbf{J}_i(\mathbf{x})$ denotes the Jacobian of network output w.r.t $\mathbf{h}_i(\mathbf{x})$; $\mathbf{H}_i(\mathbf{x})$ and $\mathbf{H}_{\text{out}}(\mathbf{x})$ denotes the Hessian of the loss w.r.t $\mathbf{h}_i(\mathbf{x})$ and the network output, respectively. Then we have $\mathbf{J}_i = (\prod_{l=i}^{L}\Theta^{(l)})^T \triangleq \Theta^{[i:L]}$ the transposition of linear multiplication of weight matrices from i-th layer to the last one. Then, we define the loss function with multiplicative structured Dropout by:

$$\widehat{L}_{\text{drop}} := \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\mathbb{E}_{\xi^{(t)}}\ell(f(\mathbf{x}\odot\xi^{(t)})),$$

where $\xi^{(t)} = \{\xi^{(t,i)}\}_{i=1}^{L}$ with the noise variable $\xi^{(t,i)}$ is specified to the i-th layer respectively (the definition of $\xi^{(t)}$ here is a bit different from that in the main text, but it is clear from the context).

Then we define an explicit regularizer induced by Dropout as follows:

$$R_{VSD} := \widehat{L}_{\text{drop}} - \widehat{L} = \frac{1}{L}\sum_{i=1}^{L} \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\left[\mathbb{E}_{\xi^{(t,i)}}\ell(f(\mathbf{x}\odot\xi^{(t,i)})) - \ell(f(\mathbf{x}))\right].$$

Let $\xi^{(t,i)} = 1 + \eta^{(t,i)}$, we then rewrite the loss after applying dropout on $\mathbf{h}_i$ by: $\ell(f(\mathbf{x}\odot\xi^{(t,i)})) = \ell(f_i(\mathbf{h}_i(\mathbf{x}) + \delta^{(t,i)}))$ with $\delta^{(t,i)} = \mathbf{h}_i(\mathbf{x})\odot\eta^{(t,i)}$. To analyze the effect of this perturbation, we apply the Taylor expansion around $\delta^{(t,i)} = \vec{0}$ as follows:

$$\ell(f(\mathbf{x}\odot\xi^{(t,i)})) - \ell(f(\mathbf{x})) \approx D_{\mathbf{h}_i}(\ell\circ f_i)[\mathbf{h}_i(\mathbf{x})]\delta^{(t,i)} + \delta^{(t,i)T}D^2_{\mathbf{h}_i}(\ell\circ f_i)[\mathbf{h}_i(\mathbf{x})]\delta^{(t,i)},$$

where $\mathcal{D}_{(.)}$ denotes the derivative operator. Take the expectations on both sides of the above equation with the note that $\delta^{(t,i)}$ is a zero-mean random variable, we obtain an approximation of the Dropout explicit regularizer in VSD corresponding to i-th layer:

$$\begin{aligned}
R^{(i)}_{VSD} &:= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\mathbb{E}_{\delta^{(t,i)}}\left[\delta^{(t,i)T}D^2_{\mathbf{h}_i}(\ell\circ f_i)[\mathbf{h}_i(\mathbf{x})]\delta^{(t,i)}\right] \\
&= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\left\langle D^2_{\mathbf{h}_i}(\ell\circ f_i)[\mathbf{h}_i(\mathbf{x})], \mathbb{E}_{\delta^{(t,i)}}[\delta^{(t,i)}\delta^{(t,i)T}]\right\rangle \\
&= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\left\langle \mathbf{H}_i(\mathbf{x}), \mathbb{E}_{\delta^{(t,i)}}[\delta^{(t,i)}\delta^{(t,i)T}]\right\rangle.
\end{aligned}$$

Because $\delta^{(t,i)} = \mathbf{h}_i(\mathbf{x})\odot\eta^{(t,i)}$ and $\eta^{(t,i)} \sim \mathcal{N}(0, \mathbf{U}\text{diag}(\alpha)\mathbf{U}^T)$, we have:

$$\mathbb{E}_{\delta^{(t,i)}}[\delta^{(t,i)}\delta^{(t,i)T}] = \mathbb{V}_{\delta^{(t,i)}}[\delta^{(t,i)}] = \text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\mathbf{h}_i(\mathbf{x})).$$

Meanwhile, for the term $\mathbf{H}_i(\mathbf{x}) = D^2_{\mathbf{h}_i}(\ell\circ f_i)[\mathbf{h}_i(\mathbf{x})]$, we can decompose into two components as:

$$D^2_{\mathbf{h}_i}(\ell\circ f_i)[\mathbf{h}_i(\mathbf{x})] = \mathbf{J}_i^T(\mathbf{x})\mathbf{H}_{\text{out}}(\mathbf{x})\mathbf{J}_i(\mathbf{x}) + \sum_j (D_f\ell[f(\mathbf{x})])_j D^2_{\mathbf{h}_i}(f_i)_j[\mathbf{h}_i(\mathbf{x})], \tag{15}$$

where j indicates the j-th coordinate in the output. The first term in the RHS of equation (15) is positive-semidefinite (when ℓ is the mean-square error or the cross-entropy loss), but the second term is likely to be non positive-semidefinite since it involves the Hessian of a non-convex model. However, [73] suggests ignoring this quantity because it is less important empirically. Then, our regularizer can be approximated by:

$$\begin{aligned}
R^{(i)}_{VSD} &= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\left\langle \mathbf{H}_i(\mathbf{x}), \text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\mathbf{h}_i(\mathbf{x}))\right\rangle &(16)\\
&\approx \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\left\langle \mathbf{J}_i^T(\mathbf{x})\mathbf{H}_{\text{out}}(\mathbf{x})\mathbf{J}_i(\mathbf{x}), \text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\mathbf{h}_i(\mathbf{x}))\right\rangle, &(17)
\end{aligned}$$

which fulfills the criteria for a valid regulariser.

We next will derive some variants of our regularizer and from which provide several interpretations of the algorithmic aspects. To simplify the presentation, we denote the matrix $\Gamma := \text{diag}(\mathbf{h}_i)\mathbf{U}\text{diag}(\alpha^{1/2})$ and the matrix $\mathbf{Q} := \Gamma\Gamma^T = \text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\mathbf{h}_i(\mathbf{x}))$.

Interpretation 1: *Dropout regularizer encourages the flatness of local minima.* In deep linear network, the second derivatives of the loss w.r.t the hidden unit and the weight parameter are tightly correlated to each other. Indeed, assume $\mathbf{Z}$ is a weight matrix following hidden layer $\mathbf{h}_i$ in the network architecture, namely $f(\mathbf{x}) = f_i(\mathbf{h}_i(\mathbf{x})) = f_{i+1}(\mathbf{Z}\mathbf{h}_i(\mathbf{x}))$, then we have:

$$D_{\mathbf{Z}}(\ell(f(\mathbf{x}))[\mathbf{Z}] = \mathbf{h}_i(\mathbf{x})D_{\mathbf{h}_{i+1}}(\ell\circ f_i)[\mathbf{h}_{i+1}(\mathbf{x})].$$

Thus, the loss derivatives w.r.t weight parameters can be expressed in terms of those w.r.t the hidden layers. For ReLU-like activations such as ELU, Softplus, our functions are at most linear, so this above equation still holds. Therefore, when the regularization of a deep linear network is measured by Hessian matrix $\mathbf{H}_i$, it will tend to penalize the magnitudes of the eigenvalues of the second derivative of the loss w.r.t the weight parameters, intuitively leading to small curvature of the corresponding local loss landscape. This helps the network converges to the flat minima which contributes to better generalization. We can also analyze this property directly from equation (17), specifically we penalize the Jacobian based on the norm of $\mathbf{H}_{\text{out}}$. This enables the learning algorithm to maintain a low empirical Lipschitz constant, and so facilitates the optimizer to exploit flat regions of the loss function.

Interpretation 2: *The structured Dropout regularizer adapts a Tikhonov-like regularization and reshapes the gradient of network weights.* From equation (16) and the relation between trace operator and inner product, we have:

$$\begin{aligned} R_{VSD}^{(i)} &\approx \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\mathbf{Trace}\left(\text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{H}_i(\mathbf{x})\right) \\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\|\mathbf{H}_i^{1/2}(\mathbf{x})\text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{U}\text{diag}(\alpha^{1/2})\|_F^2. \end{aligned}$$

This form can be interpreted as Tikhonov-like regularization imposed on the square root of Hessian matrix $\mathbf{H}_i$, in which the Tikhonov matrix is $\Gamma = \text{diag}(\mathbf{h}_i)\mathbf{U}\text{diag}(\alpha^{1/2})$. Thanks to variational learning in our method, the matrix Γ is automatically learned instead of being manually designed. This can bring beneficial properties for training objective by encoding a notion of the smoothness of the loss function [9]. In addition, when considering the case of regression problem, from equation (17) we have $\mathbf{H}_{\text{out}} = 1$ and:

$$\begin{aligned} R_{VSD}^{(i)} &\approx \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\mathbf{Trace}\left(\mathbf{Q}\mathbf{J}_i^T(\mathbf{x})\mathbf{J}_i(\mathbf{x})\right) \\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\mathbf{Trace}\left(\mathbf{J}_i(\mathbf{x})\mathbf{Q}\mathbf{J}_i^T(\mathbf{x})\right) \\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\left(\mathbf{J}_i(\mathbf{x})\mathbf{Q}\mathbf{J}_i^T(\mathbf{x})\right) \\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\left(\Theta^{[i:L]}\mathbf{Q}\Theta^{[i:L].T}\right), \end{aligned} \tag{18}$$

in which the penultimate equation is because the quantity in the trace operator is scalar. This form is a data-dependent regularization with adaptive structure determined by the matrix $\mathbf{Q}$. With vanilla Dropout, the matrix $\mathbf{Q} = \text{diag}(\alpha \odot \mathbf{h}_i(\mathbf{x})^2)$ plays a role as a scaling factor which allows Dropout regularizer to capture highly discriminative characteristics in each data feature. This interpretation generalizes some prior works studying Dropout for simple linear models [81, 27, 59]. Moreover, in VSD, the trainable orthogonal matrix $\mathbf{U}$ offers a more distinctive regularization effect. Specifically, the regularizer R_{VSD} in our method makes the training algorithm capable of adapting to non-isotropic the geometric shape of data distribution. In other words, it reshapes the gradient of network weights according to the geometry of the data based on both scale and direction information. This property also connects our method with subgradient methods for online convex optimization in [14, 25], from which our regularization is closely related to adaptive proximal functions in these online frameworks.

Interpretation 3: *VSD penalizes implicitly the spectral norm of weight matrices, which has connection to generalization.* Let $\Omega_i := \text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{J}_i^T(\mathbf{x})\mathbf{H}_{\text{out}}(\mathbf{x})\mathbf{J}_i(\mathbf{x})\text{diag}(\mathbf{h}_i\mathbf{x})$, then also from equation (17), we have:

$$\begin{aligned} R_{VSD}^{(i)} &\approx \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\mathbf{Trace}\left(\text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{J}_i^T(\mathbf{x})\mathbf{H}_{\text{out}}(\mathbf{x})\mathbf{J}_i(\mathbf{x})\right) \\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\mathbf{Trace}\left(\mathbf{H}_{\text{out}}(\mathbf{x})^{1/2}\mathbf{J}_i(\mathbf{x})\text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{U}\text{diag}(\alpha)\mathbf{U}^T\text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{J}_i^T(\mathbf{x})\mathbf{H}_{\text{out}}(\mathbf{x})^{1/2}\right) \\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\|\mathbf{H}_{\text{out}}(\mathbf{x})^{1/2}\mathbf{J}_i(\mathbf{x})\text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{U}\text{diag}(\alpha^{1/2})\|_F^2 \\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\sum_{k=1}^{K}\alpha_k\|\mathbf{H}_{\text{out}}(\mathbf{x})^{1/2}\mathbf{J}_i(\mathbf{x})\text{diag}(\mathbf{h}_i(\mathbf{x}))\mathbf{U}_{:k}\|_2^2 \\ &= \mathbb{E}_{\mathbf{x}\sim\mathcal{B}}\sum_{k=1}^{K}\alpha_k\mathbf{U}_{:k}^T\Omega_i\mathbf{U}_{:k}. \end{aligned} \tag{19}$$

Since the trainable matrix $\mathbf{U}$ satisfies $\mathbf{U}_{:k}^T\mathbf{U}_{:k} = 1$, the above form suggests that our regularization encourages the model to converge to solution with smaller spectral norms of the matrix $\mathbf{H}_{\text{out}}(\mathbf{x})^{1/2}\mathbf{J}_i(\mathbf{x})\text{diag}(\mathbf{h}_i(\mathbf{x}))$ and thus of the network weights.

Our analysis here is well-motivated by the theoretical study about generalization bound of neural nets based on the spectral norm. Concretely, Neyshabur et al. [65] and Bartlett et al. [4] showed that the generalization error is upper bounded by $\mathcal{O}\left(\sqrt{\prod_{i=1}^{L}\|\Theta^{(i)}\|_2^2\sum_{i=1}^{L}\text{srank}(\Theta^{(i)})}\right)$ that depends on two parameter-dependent quantities: a) the scale-dependent Lipschitz constant upper-bound $\prod_{i=1}^{L}\|\Theta^{(i)}\|_2^2$ (product of spectral norms) and b) the sum of scale-independent stable ranks $\sum_{i=1}^{L}\text{srank}(\Theta^{(i)})$. Essentially, this upper bound implies that smaller spectral norm and stable rank can lead to better generalization. We empirically investigate this implication and show the results

Table 9: Comparisons of Spectral Norms (SN) and Stable Ranks (SR) from different methods. Lower is better.

Methods	MLP		LeNet				AlexNet	
	MNIST		SVHN		CIFAR10		SVHN	
	SN	SR	SN	SR	SN	SR	SN	SR
MAP	**1.47**	6.33	3.95	6.64	3.15	5.45	1.62	6.78
MCD	1.93	4.36	3.30	3.99	3.35	5.19	1.83	5.42
VD	1.88	5.28	2.94	6.04	2.71	5.39	1.83	5.30
VSD	2.05	**4.19**	**2.28**	**2.54**	**1.94**	**4.94**	**1.36**	**5.23**

in Table 9, in which we measure both spectral norm and stable rank of the weight matrix in the last layer of MLP, LeNet, and AlexNet architecture trained on MNIST, CIFAR10 and SVHN respectively. It can be seen that our method leads to a consistent reduction in terms of both the spectral norm and stable rank when compared with weight decay (MAP) and vanilla Dropout methods (MCD, VD). This also is evidenced in [4] that weight decay does not significantly impact margins or generalization.

F A complementary Bayesian justification for Variational Dropout

In this section, we will bring a new Bayesian perspective for Variational Dropout methods [39, 36] and then derive a variational objective to implement them in our experiments. Note that, the analysis below does not directly address the issues of original VD, thus have not yet provided any standard Bayesian justification for this method. Our new interpretation, however, is consistent with some recent research in the community.

Variational Gaussian Dropout as Subspace inference. Reusing the analysis in Section 3.1 in the main text, we have:

$$\mathbf{W}^{(VD)} = \text{diag}(\xi)\Theta = \Theta + \text{diag}(\eta)\Theta = \Theta + \sum_{i=1}^{K} \eta_i(\text{diag}(\mathbf{e}_i)\Theta) = \Theta + \sum_{i=1}^{K} \eta_i \Theta_{(i)}, \tag{20}$$

where $\xi \sim \mathcal{N}(1, \text{diag}(\alpha))$, $\xi = 1 + \eta$ and $\Theta_{(i)}$ is the matrix Θ with only the i-th row retained. It turns out this representation can be well-interpreted under the subspace inference frameworks proposed in [32, 13]. Specifically, at each iteration of the inference phase, the weight parameter can be treated as the shift matrix, $\{\Theta_{(i)}\}_{i=1}^{K}$ are basic vectors of the subspace $\mathcal{S}$ and the noise $\eta = \{\eta_i\}_{i=1}^{K}$ is the low dimensional subspace parameter. Because $\{\Theta_{(i)}\}_{i=1}^{K}$ is linearly independent, so we can consider $\mathcal{S}$ as a projected space of the full parameter one.

We then perform variational inference over the low dimensional parameter η, also over ξ, in which the true posterior and the approximate posterior are defined by $p(\xi|\mathcal{D})$ and $q_\alpha(\xi)$ respectively. The variational objective function is given by:

$$\mathbb{E}_{q_\alpha(\xi)} \log p(\mathcal{D}|\xi, \Theta) - \mathbb{D}_{KL}(q_\alpha(\xi)\|p(\xi)). \tag{21}$$

This form suggests us a similar procedure using approximate inference on the variational noise ξ. However, our interpretation can show a distinctness in terms of model specification. Concretely, to employ variational inference on the noise *in a principle way*, we need to define a probabilistic model where the noise should play a role as a latent variable. While plausible, we are mildly cautious when introducing a new Bayesian model with an implicit treatment.

Adjust the KL divergence term with temperature scaling. Note that, the equation (21) clarifies the derivation of type-A version in Variational Dropout paper [39], in which the authors used $\mathbb{D}_{KL}(q_\alpha(\xi)\|p(\xi))$ instead of the undefined term $\mathbb{D}_{KL}(q_\phi(\mathbf{W})\|p(\mathbf{W})$ for implementation, but it seems just a heuristic way. On the other hand, [31] claimed that optimizing the above objective function might lead to significant overfitting due to the lack of regularization of Θ. This issue has actually been mentioned in the subspace inference framework [32] with a similar way, from which we can propose to leverage the temperature technique to prevent the posterior from concentrating around the maximum likelihood estimate. In particular, we use the tempered posterior:

$$p_T(\xi|\mathcal{D}) \propto p(\mathcal{D}|\xi)^{1/\tau} p(\xi), \tag{22}$$

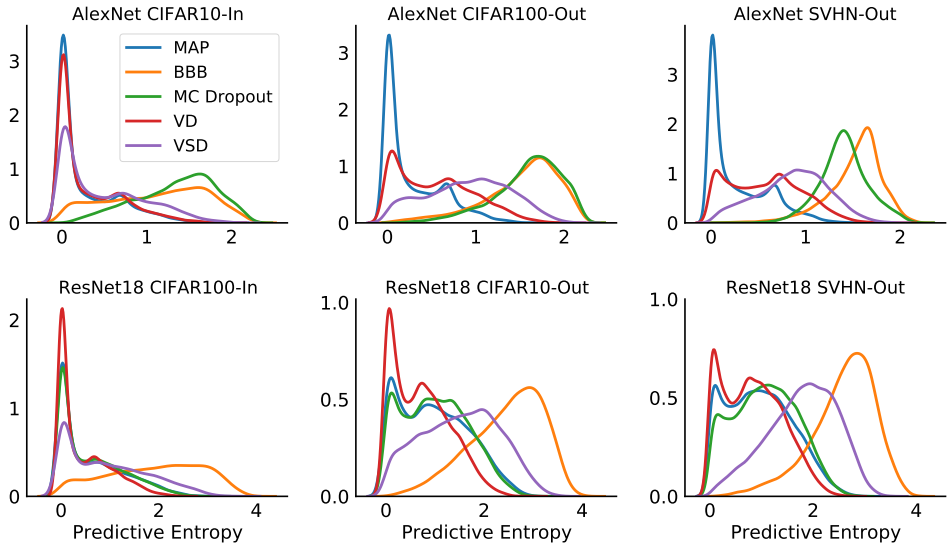


Figure 4: Histograms of predictive entropy for AlexNet (top) and ResNet18 (bottom) trained on CIFAR10 and CIFAR100 respectively.

with the temperature hyperparameter $\tau \gg 1$ chosen through cross-validation. It is also well-known that this technique is equivalent to the KL annealing (a detailed discussion is given in Appendix I.1), in which we optimize a fixed objective as:

$$\mathbb{E}_{q_\alpha(\xi)} \log p(\mathcal{D}|\xi, \Theta) - \tau * \mathbb{D}_{KL}(q_\alpha(\xi)\|p(\xi)). \tag{23}$$

We use this form for our implementation, the hyperparameter τ is tuned in the wide range instead of just picking values greater than 1. In particular for ARD-Variational Dropout method proposed in [36], the prior $p(\xi)$ is assigned by an isotropic Gaussian with the hyperparameter chosen via the Empirical Bayes, and then we obtain $\mathbb{D}_{KL}(q_\alpha(\xi)\|p(\xi)) = 0.5 \sum_{i=1}^{K} \log(1 + \alpha_i^{-1})$. It can be found that this term is a degenerate case of VSD when Householder transformation is deactivated, namely the orthogonal matrix $\mathbf{U}$ is just an identity matrix.

G Additional empirical results

G.1 Predictive entropy performance on ResNet18 architecture

We recap here a consistent qualitative description for predictive entropy performance. Basically, an accurate and well-calibrated model is expected to represent entropy values being concentrated mostly around 0 (i.e. high confidence) when the test data comes from the same underlying distribution as the training data, and in the opposite case, the predictive entropies should be evenly distributed (i.e. higher uncertainty). In fact, the deep learning models do not achieve simultaneously on both expectations at the most ideal, but instead, accurate and well-calibrated ones tend to exhibit a moderate level of confidence on in-distribution data, and then provide a reasonable representation for uncertainty estimates on out-of-distribution data.

In the bottom row of Figure 4, we show the predictive entropy of methods when training ResNet18 architechture on CIFAR100 and testing out-of-distribution on CIFAR10 and SVHN. While most methods underestimate uncertainty in out-of-distribution data, our method - VSD, calibrates the prediction with moderate confidence on in-distribution data and provides proper uncertainty on out-of-distribution settings. We also observe that BBB fails to estimate the predictive uncertainty even on in-distribution data (same behavior as on AlexNet), and with a very low accuracy, this baseline has exhibited very poor results of the mean-field BNNs compared with the Bayesian Dropout methods in terms of predictive performance.

Table 10: Average test performance for UCI regression task. Results are reported with RMSE and Std. Errors.

Dataset	BBB	VMG	MNF	SLANG	MCD	VD	D.E	VSD
Boston	3.43 ± 0.20	2.70 ± 0.13	2.98 ± 0.06	3.21 ± 0.19	2.83 ± 0.17	2.98 ± 0.18	3.28 ± 0.22	**2.64 ± 0.17**
Concrete	6.16 ± 0.13	4.89 ± 0.12	6.57 ± 0.04	5.58 ± 0.19	4.93 ± 0.14	5.16 ± 0.13	6.03 ± 0.13	**4.72 ± 0.11**
Energy	0.97 ± 0.09	0.54 ± 0.02	2.38 ± 0.07	0.64 ± 0.03	1.08 ± 0.03	0.64 ± 0.02	2.09 ± 0.06	**0.47 ± 0.01**
Kin8nm	0.08 ± 0.00	0.08 ± 0.00	0.09 ± 0.00	0.08 ± 0.00	0.09 ± 0.00	0.08 ± 0.00	0.09 ± 0.00	**0.08 ± 0.00**
Naval	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	**0.00 ± 0.00**
Power Plant	4.21 ± 0.03	4.04 ± 0.04	4.19 ± 0.01	4.16 ± 0.04	4.00 ± 0.04	3.99 ± 0.03	4.11 ± 0.04	**3.92 ± 0.04**
Wine	0.64 ± 0.01	0.63 ± 0.01	**0.61 ± 0.00**	0.65 ± 0.01	0.61 ± 0.01	0.62 ± 0.01	0.64 ± 0.00	0.63 ± 0.01
Yacht	1.13 ± 0.06	0.71 ± 0.05	2.13 ± 0.05	1.08 ± 0.06	0.72 ± 0.05	1.09 ± 0.09	1.58 ± 0.11	**0.69 ± 0.06**

Table 11: Average test performance for UCI regression task. Results are reported with test LL and Std. Errors.

Dataset	BBB	VMG	MNF	SLANG	MCD	VD	D.E	VSD
Boston	-2.66 ± 0.06	-2.46 ± 0.09	-2.51 ± 0.06	-2.58 ± 0.05	-2.40 ± 0.04	-2.39 ± 0.04	-2.41 ± 0.06	**-2.35 ± 0.05**
Concrete	-3.25 ± 0.02	-3.01 ± 0.03	-3.35 ± 0.04	-3.13 ± 0.03	**-2.97 ± 0.02**	-3.07 ± 0.03	-3.06 ± 0.04	**-2.97 ± 0.02**
Energy	-1.45 ± 0.10	-1.06 ± 0.03	-3.18 ± 0.07	-1.12 ± 0.01	-1.72 ± 0.01	-1.30 ± 0.01	-1.38 ± 0.05	**-1.06 ± 0.01**
Kin8nm	1.07 ± 0.00	1.10 ± 0.01	1.04 ± 0.00	1.06 ± 0.00	0.97 ± 0.00	1.14 ± 0.01	1.20 ± 0.00	**1.17 ± 0.01**
Naval	4.61 ± 0.01	2.46 ± 0.00	3.96 ± 0.01	4.76 ± 0.00	4.76 ± 0.01	4.81 ± 0.00	5.63 ± 0.00	**4.83 ± 0.01**
Power Plant	-2.86 ± 0.01	-2.82 ± 0.01	-2.86 ± 0.01	-2.84 ± 0.01	**-2.79 ± 0.01**	-2.82 ± 0.01	-2.79 ± 0.01	**-2.79 ± 0.01**
Wine	-0.97 ± 0.01	-0.95 ± 0.01	-0.93 ± 0.00	-0.97 ± 0.01	**-0.92 ± 0.01**	-0.94 ± 0.01	-0.94 ± 0.03	-0.95 ± 0.01
Yacht	-1.56 ± 0.02	-1.30 ± 0.02	-1.96 ± 0.05	-1.88 ± 0.01	-1.38 ± 0.01	-1.42 ± 0.02	-1.18 ± 0.05	**-1.14 ± 0.02**

G.2 Regression with UCI datasets

We implement a standard experiment for Bayesian regression task on UCI dataset [3] proposed in [28]. We follow the original setup used in [20]. Detailed descriptions of the data and experimental setting can be found in Appendix I.3. We present the performance of methods based on standard metrics including root mean square error (RMSE) in Table 10 and predictive log-likelihood (LL) in Table 11. As shown in the tables, VSD performs better than baselines on most datasets in terms of both criteria (5/8 tasks on RMSE and 7/8 tasks on predictive LL). Especially, in comparison with other structured approximations on BNNs such as VMG, MNF and SLANG, our method presents much more convincing results, while MNF even shows no noticeable improvement compared to mean-filed approximation (BBB). VSD also achieves better results with a significant margin compared with VD and Deep Ensemble (D.E), specifically on *Boston, Concrete, Energy, Yacht* datasets. This demonstrates the effectiveness of learning a structured representation for multiplicative Gaussian noise instead of using a diagonal distribution as in VD.

For predictive log-likelihood measure, although VSD is comparable to MCD and VMG on some datasets such as *Concrete, Power Plant*, however our method overall shows better results consistently on almost all settings. For instance, MCD gets poor results on *Energy, Kin8nm* and *Yacht*, while VMG is worse than VSD in *Boston, Naval* and *Yacht* by large margins. In comparison to VD and MCD, our method improves considerably the performance on *Concrete, Energy* and *Yacht*.

G.3 Uncertainty with toy regression

We provide an additional experiment to assess qualitatively the predictive uncertainty of methods using a synthetic regression dataset introduced in [28]. We generated 20 training inputs from $\mathcal{U}[-4, 4]$ and assigned the corresponding target as $y_n = x_n^3 + \epsilon_n$ where $\epsilon_n \sim \mathcal{N}(0, 9)$. We then fitted a neural network with a single hidden layer of 100 units. We also fixed the variance of likelihood regression to the true variance of noise ϵ. We compare the performance of methods including BBB, MCD, VD and VSD. For the Dropout-based methods, we do not apply the dropout noise for the input layer since it is 1-dimensional. At the test time of all methods, we use 1000 MC samples to approximate the predictive distribution. The results are shown in Figure 5.

We would expect that in the area of observed data, the models should obtain the predictive means closer to the ground truth with high confidence, and at the same time, increase the predictive variance when moving away from the data. Thus we can see that our method, VSD, provides a more realistic predictive distribution than the remaining ones.

For MCD, we fixed the dropout rate p at default 0.5, because tuning manually this value does not increase the variance of noise distribution Bernoulli$(1 - p)$, and this will lead to less variance in subsequent pre-activation unit by the Central Limit Theorem [82]. Therefore, with the shallow

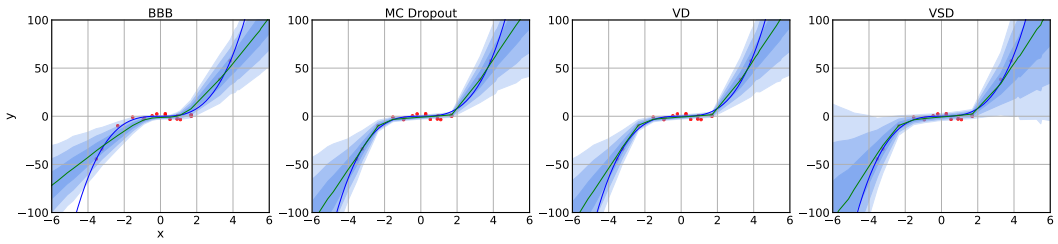


Figure 5: Predictive distributions for the toy dataset. The observations are shown as red dots. The blue line represents the true data generating function and the mean predictions are shown as the green line. Blue areas correspond to ±3 standard deviation around the mean.

architecture of one hidden unit in this experiment, any tuning of the droprate p can result into a decrease in the predictive variance of model.

Whereas with VD, the droprate α needs to be restricted in range $(0, 1)$ during the training to avoid the poor local optima and to prevent the objective function from being degenerate [39, 57, 31]. As a consequence, this can reduce the ensemble diversity in the predictions of this method that leads to underestimate the uncertainty of predictive distribution.

On the contrary, the parameters in our method can be optimized without any limited assumptions. Moreover, with a structured representation for Gaussian perturbation, our method can capture rich statistical dependencies in the true posterior that facilitates fidelity posterior approximation and then provides proper estimates of the true model uncertainty.

In addition, we also conduct one more experiment to investigate the ability of VSD to estimate in-between uncertainty. This experiment is motivated by a recent work of Foong et al. [17], in which the authors proved that for shallow Bayesian neural nets, neither mean-field Gaussian nor Dropout posterior are capable of expressing meaningful in-between uncertainty in many situations. Due to the acquired distinctiveness in the structure of Dropout posterior distribution, we suggest that our method-VSD can theoretically overcome this limitation of MC Dropout and Variational Dropout. We validate empirically this statement by considering a regression problem with the dataset consisting of two well-separated clusters of covariates. We apply VSD to train a ReLU network with one hidden layer of size 50 and then present the predictive distributions in Figure 6. Contrary to the behavior of MC Dropout and VD reported in [17], VSD can represent a reasonable uncertainty between two data clusters. A more comprehensive analysis would be promising work for future research.

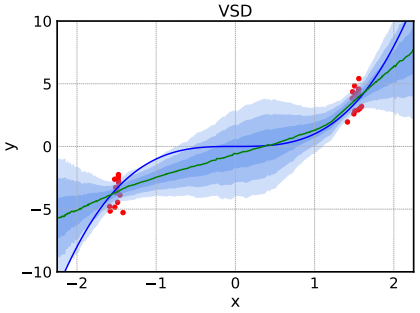


Figure 6: In-between uncertainty.

H Further discussion of uncertainty measures

We present here several approaches used to measure or assess the quality of the predictive uncertainty of models. We also present some insights into what those metrics mean. For simplicity, we consider the multi-class classification problems on the supervised dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$. For Bayesian methods, we can compute the predictive probabilities for each sample $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}$ that belongs to class c as:

$$\hat{p}_{ic} := \int p(\mathbf{y}_i = c|\mathbf{x}_i, \mathbf{W}) p(\mathbf{W}|\mathcal{D}) d\mathbf{W}$$

$$\approx \int p(\mathbf{y}_i = c|\mathbf{x}_i, \mathbf{W}) q(\mathbf{W}) d\mathbf{W} \approx \frac{1}{S} \sum_{s=1}^{S} p(\mathbf{y}_i = c|\mathbf{x}_i, \mathbf{W}^{(s)}) \tag{24}$$

with $\{\mathbf{W}^{(s)}\}_{s=1}^S$ are variational Monte Carlo samples. The following metrics are all based on this predictive probability.

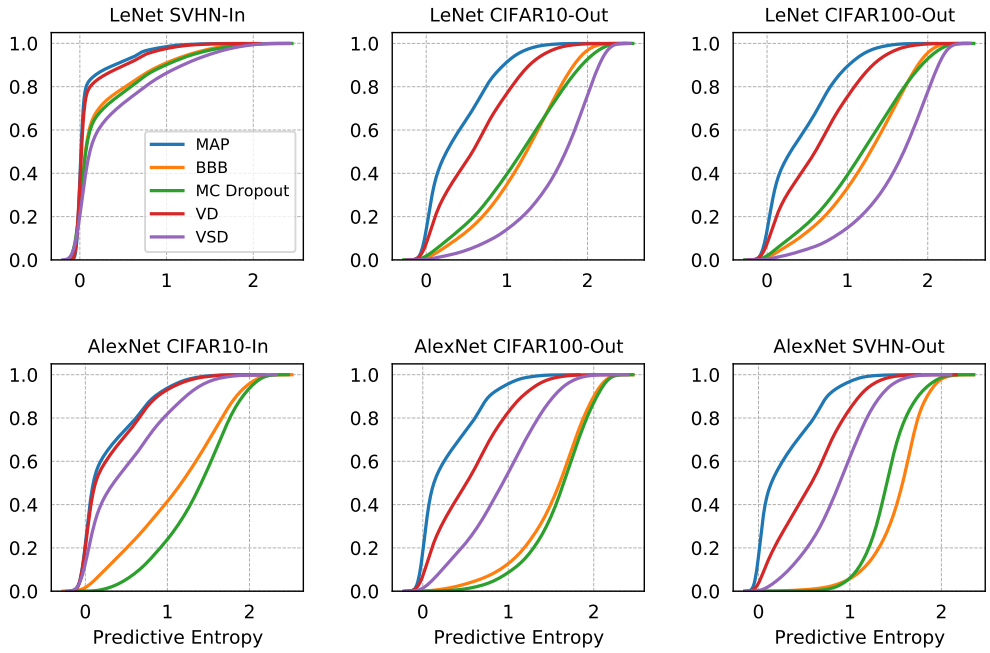


Figure 7: Empirical CDF of the predictive entropy forfor LeNet (top) and AlexNet (bottom) trained on SVHN and CIFAR10 respectively.

H.1 Negative log-likelihood

The negative log-likelihood (NLL) is a standard measure of a probabilistic model's quality and also a common uncertainty metric which is defined by: $\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{K} -y_{ic}\log\hat{p}_{ic}$. If the predicted probabilities are overconfident, NLL will severely penalize incorrect predictions, causing this quantity to become large even if the test error rate is low. In contrast, an underconfident prediction contributes a substantial amount to the NLL regardless of whether the prediction is correct or not. Therefore, a model that achieves a good test NLL tends to make predictions with sufficiently high confidence on easy samples and hesitant predictions on hard, easy-to-fail samples.

These arguments can be evidenced by the results of experiments on modern convolutional networks (Table 4 and Table 5). Although MAP has the predictive accuracy being competitive with our method, it comes at a trade-off with the worst results on NLL. Thus the predictions of MAP are more likely to be overconfident. Corresponding experiments on out-of-distribution settings (Figure 3 bottom and Figure 4) confirmed this.

H.2 Predictive entropy

Predictive entropy determined on a input sample $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}$ is given by $\frac{1}{K}\sum_{c=1}^{K} -\hat{p}_{ic}\log\hat{p}_{ic}$. Underconfident models give noisy predictive predictions which result in high entropy on even in-distribution data. In contrast, overconfident models with spike predictive distributions tend to produce near zero predictive entropies.

In Figures 3 and Figure 4 we plot the histogram of predictive entropies to quantify uncertainty estimation of the methods on out-of-distribution settings. In some cases, when the histograms are difficult to distinguish from each other, we instead use the empirical CDF which may be more informative [48]. We redraw the Figure 3 in the main text by empirical CDF in Figure 7. The distance between lines gives more visual views on the performance differences between the methods.

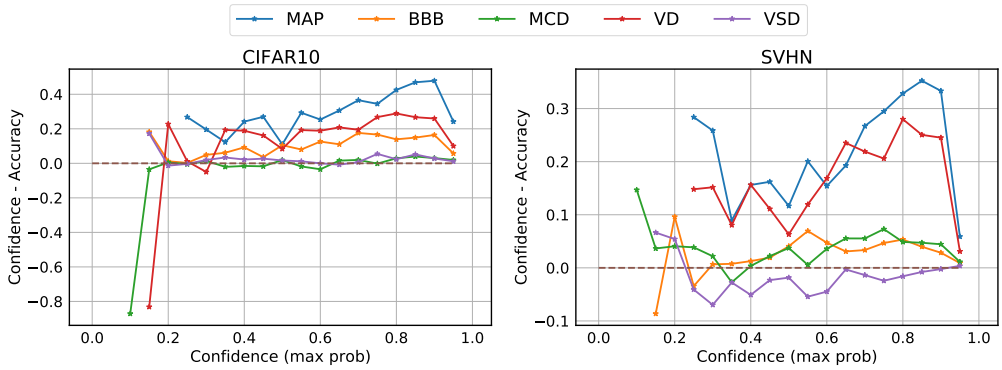


Figure 8: Reliability diagrams of LeNet-5 on CIFAR10 and SVHN dataset.

H.3 Expected Calibration Error

Expected Calibration Error (ECE) [23] captures the discrepancy between model's predicted probability estimates and the actual accuracy. This quantity is computed by first binning the predicted probabilities into M distinct bins and calculate the accuracy of each bin. Let B_m be the set of indices of samples whose prediction confidence falls into the interval $I_m = (\frac{m-1}{M}, \frac{m}{M}]$. The accuracy of B_m and the average confidence within bin B_m are defined as follows:

$$\text{acc}(B_m) := \frac{1}{|B_m|} \sum_{i \in B_m} \mathbf{1}[\hat{\mathbf{y}}_i = \mathbf{y}_i], \quad \text{conf}(B_m) := \frac{1}{|B_m|} \sum_{i \in B_m} q(\mathbf{x}_i),$$

where $q(\mathbf{x}_i)$ is the confidence for sample i. In this work, we define the confidence score q as the maximum predictive probability on each data of the classifier. ECE is then computed by taking a weighted average of the bins' accuracy/confidence difference:

$$\text{ECE} := \sum_{m=1}^{M} \frac{|B_m|}{n} |\text{acc}(B_m) - \text{conf}(B_m)|. \tag{25}$$

Reliability Diagrams [23] in Figure 8 are visual representations of model calibration. These diagrams plot accuracy as a function of confidence. Any deviation from a perfect horizontal zero-line represents miscalibration. We can observe that MAP and VD exhibit overconfident prediction (low accuracy on many bins of high confidence). Meanwhile, VSD calibrates well the predictive probabilities resulting in the lowest ECE in both settings.

H.4 Out-of-distribution detection metrics

We measure some metrics to evaluate the model's ability of distinguishing in distribution and out-of-distribution images.

An ideal classifier should have a low probability of false alarm, corresponding to a low False Positive Rate (FPR), while maintaining a high sensitivity, corresponding to a high True Positive Rate (TPR). **FPR at 95% TPR** is a suitable metric to measure the reality of this desire.

Receiver Operating Characteristic (ROC) curve plots the TPR against the FPR at all possible decision thresholds. The larger the Area Under the ROC curve (**AUROC**), the better the classifier's ability to separate the negative and positive samples. More intuitively, AUROC indicates the chance that the classifier produces a higher score on a random positive sample than a random negative one.

Besides the ROC curve, which represents the trade-off between the sensitivity and the probability of false alarm, Precision-Recall (PR) curves are often plotted to represent a trade-off between making accurate positive predictions and covering a majority of all positive results. We report the Area Under the Precision-Recall curve (AUPR) in two scenarios: (i) in-distribution samples are used as the positive samples (**AUPR-In**), (ii) out-of-distribution samples are used as the positive samples (**AUPR-Out**).

Table 12: Quality of out-of-distribution detection on image classification tasks. (Top) LeNet-5 train on SVHN, evaluate on CIFAR10, CIFAR100. (Middle) AlexNet train on CIFAR10, evaluate on CIFAR100, SVHN. (Bottom) ResNet-18 train on CIFAR, evaluate on CIFAR10, CIFAR100. ↑ (AUROC, AUPR IN, AUPR OUT) indicates larger value is better, and ↓ (FPR, Detection error) indicates lower value is better. VSD performs the best on almost all metrics and datasets.

LeNet-5 (SVHN)	**CIFAR10**					**CIFAR100**				
	FPR	Det. err.	AUROC	AUPR IN	AUPR OUT	FPR	Det. err.	AUROC	AUPR IN	AUPR OUT
MAP	0.78	0.23	0.83	0.93	0.58	0.76	0.22	0.84	0.93	0.60
BBB	0.56	0.17	0.90	0.96	0.73	0.54	0.17	0.90	0.96	0.75
MCD	0.50	0.15	0.92	**0.97**	0.78	0.49	0.15	**0.92**	**0.97**	0.78
VD	0.62	0.17	0.89	0.96	0.71	0.64	0.17	0.89	0.96	0.71
VSD	**0.45**	**0.14**	**0.93**	**0.97**	**0.81**	**0.47**	**0.14**	**0.92**	**0.97**	**0.79**

AlexNet (CIFAR10)	**CIFAR100**					**SVHN**				
	FPR	Det. err.	AUROC	AUPR IN	AUPR OUT	FPR	Det. err.	AUROC	AUPR IN	AUPR OUT
MAP	0.88	0.35	0.70	0.73	0.65	**0.89**	0.33	0.71	0.59	**0.83**
BBB	0.93	0.46	0.55	0.54	0.54	0.99	0.45	0.53	0.33	0.70
MCD	0.91	0.41	0.63	0.63	0.60	0.97	0.39	0.59	0.47	0.74
VD	0.87	0.35	0.69	0.72	0.64	0.89	0.32	0.72	0.60	**0.83**
VSD	**0.85**	**0.33**	**0.72**	**0.76**	**0.68**	0.91	**0.30**	**0.73**	**0.65**	**0.83**

ResNet-18 (CIFAR100)	**CIFAR10**					**SVHN**				
	FPR	Det. err.	AUROC	AUPR IN	AUPR OUT	FPR	Det. err.	AUROC	AUPR IN	AUPR OUT
MAP	0.89	**0.37**	0.67	0.70	0.63	0.91	0.36	0.68	0.56	0.81
BBB	0.93	0.41	0.62	0.66	0.58	0.89	0.37	0.68	0.51	0.82
MCD	0.89	**0.37**	0.68	0.71	0.63	0.89	0.34	0.71	0.58	0.83
VD	0.90	0.38	0.66	0.70	0.62	0.87	0.34	0.70	0.58	0.83
VSD	**0.87**	**0.37**	**0.69**	**0.72**	**0.65**	**0.83**	**0.31**	**0.76**	**0.65**	**0.86**

Finally, we report the **detection error**, a measure of the minimum expected probability that the model incorrectly detects whether data samples come from in or out of training data distribution. This quantity is defined as $\min_\delta\{0.5P_{in}(q(\mathbf{x}) \leq \delta) + 0.5P_{out}(q(\mathbf{x}) > \delta)\}$ where δ is the decision threshold and $q(\mathbf{x})$ is the maximum value of softmax probability.

I Details for experimental settings

I.1 Training techniques

Initialization and Learning rate scheduling. It is well known that good initialization and proper learning rate can improve both the speed and quality of convergence in the training process. In our experiments, we adopt `init.xavier_uniform` in PyTorch to initialize the weight parameters of all methods, of which we use 5 random seeds for different initializations and then report average results.

For positive-valued parameters such as dropout rate α or Gaussian variance σ^2, we optimize in the logarithmic form to avoid numerical issues and bad local optima. We use Adam optimizer with initial learning rate $lr \in \{0.001, 0.002\}$ on all of our experiments, and then we also apply `MultiStepLR` scheduler with multiplicative factor `gamma`=0.3 to adjust the learning rate after every 10 epochs.

KL annealing. This technique re-weights the expected log-likelihood and regularized term by a scaling factor λ as follows:

$$\mathbb{E}_{q_\phi \mathbf{W}} \log p(\mathcal{D}|\mathbf{W}) - \lambda \mathbb{D}_{KL}(q_\phi(\mathbf{W})||p(\mathbf{W})). \tag{26}$$

The KL annealing in many contexts is remarkably effective to the problems using variational Bayesian inference. It intuitively can prevent underfitting/over-regularization issue, or mitigate KL-vanishing phenomenon. The KL annealing has an interpretation of the Bayesian principle. Indeed, re-weighting the KL term by a λ is equivalent to tempering the posterior by a temperature factor $\tau = \lambda$, namely using $p_\tau(\mathbf{W}|\mathcal{D}) \propto p(\mathcal{D}|\mathbf{W})^{1/\tau} p(\mathbf{W})$ [84]. In another interpretation, one can use a different likelihood $p_\tau(\mathcal{D}|\mathbf{W}) = p(\mathcal{D}|\mathbf{W})^{1/\tau}$ instead of a tempered posterior $p_\tau(\mathbf{W}|\mathcal{D})$ above [86]. Even though the posteriors coincide, the predictive distribution differs for these two ways.

A temperature value $\tau < 1$, namely $\lambda < 1$, is corresponding to artificially sharpening the posterior distribution, which can be interpreted as overcounting the data $\mathcal{D}$ by a factor of $1/\tau$. So, the variational objective with KL annealing is a lower bound of τ times the model evidence defined on the overcounted data $\widehat{\mathcal{D}}$. Maximizing this objective is equivalent to minimizing the KL divergence between the approximate distribution $q_\phi(\mathbf{W})$ and the tempered posterior $p_\tau(\mathbf{W}|\mathcal{D})$.

We employ this technique for all methods in our paper and then tuning the weighting hyper-parameter λ by cross-validation (of course with $\lambda = 1$, we have no modification). The empirical values of λ is given in Appendix I.2.

I.2 Hyperparameter tuning for all methods

We present here in detail the hyper-parameter setups of each method used in our experiments. These methods include MAP, Bayes by Backprop (BBB), MC Dropout (MCD), Variational Dropout (VD), and our method-Variational Structured Dropout (VSD); for the remaining methods such as Variational Matrix Gaussian (VMG), low-rank approximations (SLANG, and ELRG), we inherited the results reported in the original paper with the same experimental settings.

For **BBB** without the local reparameterization trick, we use two Monte Carlo samples for all of our experiments. This method needs to tune the hyper-parameters in the scale mixture Gaussian prior $p(\mathbf{W}) = \pi\mathcal{N}(0, \sigma_1^2) + (1-\pi)\mathcal{N}(0, \sigma_2^2)$ with the search as follows: $-\log\sigma_1 \in \{0, 1, 2\}$, $-\log\sigma_2 \in \{6, 7, 8\}$ and mixture ratio $\pi \in \{0.25, 0.5, 0.75\}$.

For **MC Dropout** in image classification task, we tune the droprate $p \in \{0.2, 0.3, 0.4, 0.5, 0.6\}$ in Bernoulli distribution and the length-scale $l^2 \in \{0.0001, 0.001, 0.01, 0.1, 1, 10, 100\}$ in the isotropic Gaussian prior $p(\mathbf{W}) = \mathcal{N}(0, l^{-2}\mathbf{I}_K)$.

For **VD** and **VSD**, we find a good initialization for the Gaussian dropout rate $\alpha = p/(1-p)$. However, this droprate α in VD methods needs to be restricted in range $(0, 1)$ during the training to prevent the objective function from being degenerate which can lead to the poor local optima as mentioned in some related papers [39, 57, 31].

We also employ the KL annealing mentioned in the previous section I.1 for BBB, VD and VSD methods. This technique has actually been exploited in the original papers of BBB and VD to improve the predictive performance. The most appropriate values of annealing factor λ we have found are consistent with the reports of previous works in the literature that have been synthesized in [84]. Specifically, we used $\lambda \in \{0.1, 0.2, 0.5\}$ for the classification tasks, and λ from 10^{-5} to 10^{-2} for the regression tasks.

For **SWAG**, we train the models using SGD with momentum $\gamma = 0.9$. For the learning rate we adopt the same decaying schedule as in the paper: a constant learning rate of 5×10^{-2}, up to 50% of the total number of epochs, and then a linear decay down to a learning rate of 5×10^{-4} until 90% of the total number of epochs, where once again the learning rate is kept constant until the end of training. We use rank $K = 20$ for estimation of Gaussian covariance matrix. At test time we use 30 weight samples for Bayesian model averaging. We also tune the weight decay in the range of $\{1e-3, 5e-3, 1e-4, 5e-4\}$.

I.3 UCI regression settings

Following the original settings, we used a Bayesian neural network with one hidden layer of 50 units and ReLU activation functions. We also used the 20 splits of the data provided by [20][2] for training and testing. The models are trained to convergence using Adam optimizer [37] with the learning rate $lr = 0.001$, the batchsize $M = 128$ and 2000 epochs for all datasets. To make an ensemble prediction at the test time, we used 10000 Monte Carlo samples for the Bayesian Dropout methods as the suggestion in [20].

For VMG, SLANG and Deep Ensemble, we inherited the results reported in the original paper. The results of MNF is report in [78]. For a more fair comparison, we used the results of MC Dropout reported in [60] with the version using 4000 epochs for convergence training and tuning hyper-parameters by Bayesian Optimization.

[2]The splits are publicly available from `https://github.com/yaringal/DropoutUncertaintyExps`

Table 13: The performance of VSD and VOGN on CIFAR10. Results are averaged over 5 random seeds.

CIFAR10	AlexNet				ResNet18			
	NLL	ACC	ECE	time	NLL	ACC	ECE	time
VOGN	0.703 ± 0.006	75.48 ± 0.478	**0.016 ± 0.001**	3.25x	0.477 ± 0.006	84.27 ± 0.195	**0.040±0.002**	4.44x
VSD	**0.656 ± 0.009**	**78.21 ± 0.153**	0.046 ± 0.003	**1.32x**	**0.464 ± 0.019**	**87.44 ± 0.497**	0.061 ± 0.005	**1.86x**

In this experiment, we used the number of Householder transformations $T = 2$, and need to tune the precision τ of Gaussian likelihood $p(\mathbf{y}|\mathbf{W}, \mathbf{x}, \tau) = \mathcal{N}(\mathbf{y}|f(\mathbf{W}, \mathbf{x}), \tau)$. Similar to MC Dropout and SLANG, we used 40 iterations of Bayesian Optimization (BO) to tune this precision. For each iteration of BO, 5-fold cross-validation is used to evaluate the considered hyperparameter setting. This is repeated for each of the 20 train-test splits for each dataset. The final values of each dataset are reported with the mean and standard error from these 20 runs.

I.4 Image classification settings

With the MNIST dataset, 60,000 training points were split into a training set of 50,000 and a validation set of 10,000. We then vectorized the images and trained using two fully connected Bayesian neural networks with the size of hidden layers of 400x2 and 750x3 respectively.

For the remaining two datasets CIFAR10 and SVHN, we used the same simple convolutional neural network consisting of two convolutional layers with 32 and 64 kernels, followed by a fully connected network with one hidden layer of size 128.

We trained the models with the default Adam optimizer using learning rate 0.001, batchsize 100, and the number of epochs 100. At the test time, we used 100 Monte Carlo samples for all methods. We also used the number of Householder transformations $T \in \{1, 2, 3\}$ for our method on all three datasets. With Bayes by Backprop (BBB), we did not employ the local reparameterization trick, instead we used two MC samples during the training follow the code published by the authors. We utilized the available results of the baselines in the same setting, including NLL and error rate of ELRG on MNIST and CIFAR10 dataset, error rate of VMG and SLANG on MNIST dataset.

Note that, in these experiments, we did not implement fully Bayesian inference for the convolutional layer, we instead just have done it on fully connected layers. This is because our initial intention was to conduct a fair experiment to compare with some other structured approximations including Variational Matrix Gaussian (VMG) and SLANG. These methods are nontrivial when applied to convolutional layers, even no available results for CNNs have been reported. Unfortunately, however, we had not successfully implemented these methods even on fully connected networks (only some of their results are inherited in the main text)). On the other hand, we have performed fully Bayesian approximations on the large-scale architecture AlexNet and ResNet18 as the following description.

I.5 Scaling up modern CNNs settings

We follow the experimental setup for Bayesian deep convolutional networks proposed in [80]. We trained all algorithms for 300 epochs using a batch size of 200 and the ADAM optimizer with learning rate 0.001. We normalize datasets using empirical mean and standard deviation and then employ data augmentation for these experiments: random padding followed by flipping left/right (except SVHN). In the testing phase, we used 10 variational Monte Carlo samples for both AlexNet and ResNet18 architecture. We report average results over 5 random initializations. We refer to the code for MC Dropout and Bayes by Backprop on AlexNet and ResNet18 that is available at the repo `https://github.com/team-approx-bayes/dl-with-bayes` of VOGN paper [68]. We would also like to provide a few comparisons between VSD and VOGN as shown in Table 13, from which our method VSD has better performance than VOGN on almost all metrics, especially on the accuracy and computational time.

I.6 Empirical classification results with standard deviations

For the convenience of presentation, we have not included in the main text the standard deviations of empirical results in Tables 3-4-5. We provide them here for clarification.

Table 3: Results for VSD and baselines on vectorized MNIST, CIFAR10 and SVHN. Results are averaged over 5 random seeds. For all metrics, lower is better.

Method	MNIST						CIFAR10			SVHN		
	FC 400x2			FC 750x3			CNN 32x64x128					
	NLL	err. rate	ECE	NLL	err. rate	ECE	NLL	err. rate	ECE	NLL	err. rate	ECE
MAP	0.098 ± 0.009	1.32 ± 0.17	0.011 ± 0.002	0.109 ± 0.009	1.27 ± 0.18	0.011 ± 0.002	2.847 ± 0.327	34.04 ± 1.547	0.272 ± 0.008	0.855 ± 0.049	12.26 ± 0.522	0.086 ± 0.009
BBB	0.109 ± 0.015	1.59 ± 0.24	0.011 ± 0.002	0.140 ± 0.017	1.50 ± 0.21	0.013 ± 0.002	1.202 ± 0.114	30.11 ± 0.762	0.098 ± 0.006	0.545 ± 0.033	10.57 ± 0.236	0.017 ± 0.006
MCD	0.049 ± 0.005	1.26 ± 0.15	0.007 ± 0.001	0.057 ± 0.006	1.22 ± 0.14	0.007 ± 0.001	0.794 ± 0.035	26.91 ± 0.241	0.024 ± 0.003	0.365 ± 0.018	9.23 ± 0.287	0.013 ± 0.006
VD	0.051 ± 0.005	1.21 ± 0.18	0.007 ± 0.001	0.061 ± 0.007	1.17 ± 0.18	0.008 ± 0.001	1.176 ± 0.067	27.45 ± 0.222	0.156 ± 0.007	0.534 ± 0.042	9.47 ± 0.331	0.055 ± 0.007
ELRG	0.053 ± 0.006	1.54 ± 0.18	-	-	-	-	0.871 ± 0.011	29.43 ± 0.320	-	-	-	-
VSD	**0.042 ± 0.004**	**1.08 ± 0.07**	**0.006 ± 0.000**	**0.048 ± 0.004**	**1.09 ± 0.07**	**0.006 ± 0.000**	**0.730 ± 0.018**	**24.92 ± 0.355**	**0.020 ± 0.003**	**0.299 ± 0.009**	**8.39 ± 0.201**	**0.008 ± 0.001**
D.E	0.057 ± 0.005	1.29 ± 0.11	0.009 ± 0.001	0.063 ± 0.007	1.21 ± 0.11	0.009 ± 0.002	1.815 ± 0.218	26.44 ± 0.211	0.042 ± 0.003	0.783 ± 0.015	9.31 ± 0.298	0.070 ± 0.005
SWAG	0.044 ± 0.005	1.27 ± 0.13	0.008 ± 0.001	**0.043 ± 0.004**	1.25 ± 0.12	0.007 ± 0.001	0.799 ± 0.103	26.94 ± 0.176	**0.012 ± 0.002**	0.312 ± 0.011	8.42 ± 0.191	0.021 ± 0.003

Table 4: Image classification using AlexNet architecture. Results are averaged over 5 random seeds.

AlexNet	CIFAR10			CIFAR100			SVHN			STL10		
	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE
MAP	1.038 ± 0.013	69.58 ± 0.57	0.121 ± 0.003	4.705 ± 0.075	40.23 ± 0.56	0.393 ± 0.015	0.418 ± 0.022	87.56 ± 0.58	0.033 ± 0.007	2.532 ± 0.278	65.70 ± 0.88	0.267 ± 0.060
BBB	0.994 ± 0.008	65.38 ± 0.32	0.062 ± 0.004	2.659 ± 0.051	32.41 ± 1.39	**0.049 ± 0.010**	0.476 ± 0.015	87.30 ± 0.56	0.094 ± 0.005	1.707 ± 0.085	65.46 ± 0.52	0.222 ± 0.008
MCD	0.717 ± 0.010	75.22 ± 0.33	0.023 ± 0.003	2.503 ± 0.025	42.91 ± 0.36	0.151 ± 0.021	0.401 ± 0.008	88.03 ± 0.13	0.023 ± 0.003	1.059 ± 0.013	63.65 ± 1.10	**0.052 ± 0.021**
VD	0.702 ± 0.014	77.28 ± 0.30	**0.028 ± 0.002**	2.582 ± 0.085	43.10 ± 0.64	0.106 ± 0.009	0.327 ± 0.015	90.76 ± 0.44	0.010 ± 0.004	2.130 ± 0.038	65.48 ± 0.98	0.195 ± 0.010
ELRG	0.723 ± 0.010	76.87 ± 0.42	0.065 ± 0.006	2.368 ± 0.043	42.90 ± 0.84	0.099 ± 0.022	0.312 ± 0.007	90.66 ± 0.31	**0.006 ± 0.001**	1.088 ± 0.046	59.99 ± 2.15	**0.018 ± 0.008**
VSD	**0.656 ± 0.009**	**78.21 ± 0.15**	**0.046 ± 0.003**	**2.241 ± 0.026**	**46.85 ± 0.99**	0.112 ± 0.010	**0.290 ± 0.010**	**91.62 ± 0.33**	**0.008 ± 0.001**	**1.019 ± 0.039**	**67.98 ± 0.50**	0.079 ± 0.010
D.E	0.872 ± 0.008	77.56 ± 0.17	0.115 ± 0.004	3.402 ± 0.067	46.42 ± 0.33	0.314 ± 0.020	0.319 ± 0.019	90.30 ± 0.47	**0.008 ± 0.001**	2.229 ± 0.155	**68.51 ± 0.78**	0.241 ± 0.057
SWAG	**0.651 ± 0.014**	**78.14 ± 0.51**	0.059 ± 0.003	**1.958 ± 0.051**	**49.81 ± 0.62**	**0.028 ± 0.009**	0.331 ± 0.019	90.04 ± 0.21	0.031 ± 0.005	1.522 ± 0.097	**68.41 ± 0.68**	0.161 ± 0.013

Table 5: Image classification using ResNet18 architecture. Results are averaged over 5 random seeds.

ResNet18	CIFAR10			CIFAR100			SVHN			STL10		
	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE	NLL	ACC	ECE
MAP	0.644 ± 0.015	86.34 ± 0.36	0.093 ± 0.005	2.410 ± 0.012	55.38 ± 0.25	0.243 ± 0.003	0.232 ± 0.005	95.32 ± 0.16	0.028 ± 0.002	1.401 ± 0.014	71.26 ± 0.65	0.199 ± 0.022
BBB	0.697 ± 0.005	76.63 ± 0.32	0.071 ± 0.004	2.239 ± 0.005	41.07 ± 0.31	0.100 ± 0.001	0.218 ± 0.004	94.53 ± 0.24	0.047 ± 0.003	1.290 ± 0.143	71.55 ± 1.95	0.179 ± 0.019
MCD	0.534 ± 0.012	87.47 ± 0.24	0.084 ± 0.001	2.121 ± 0.020	59.28 ± 0.19	0.227 ± 0.001	0.207 ± 0.001	95.78 ± 0.02	0.026 ± 0.001	1.333 ± 0.024	72.28 ± 0.57	0.188 ± 0.027
VD	0.451 ± 0.028	87.68 ± 0.11	**0.024 ± 0.008**	2.888 ± 0.093	56.80 ± 0.77	0.284 ± 0.009	0.164 ± 0.006	96.11 ± 0.05	0.017 ± 0.006	1.084 ± 0.066	73.29 ± 0.47	0.084 ± 0.031
ELRG	**0.382 ± 0.010**	87.24 ± 0.37	**0.018 ± 0.002**	1.634 ± 0.019	58.14 ± 0.28	**0.096 ± 0.005**	0.145 ± 0.001	96.03 ± 0.05	**0.003 ± 0.000**	0.811 ± 0.018	73.66 ± 0.36	**0.080 ± 0.012**
VSD	0.464 ± 0.019	87.44 ± 0.50	0.061 ± 0.005	**1.504 ± 0.011**	**60.15 ± 0.20**	0.116 ± 0.002	**0.140 ± 0.003**	**96.41 ± 0.03**	**0.003 ± 0.000**	**0.769 ± 0.013**	**74.50 ± 0.55**	**0.083 ± 0.008**
D.E	0.488 ± 0.029	**88.91 ± 0.67**	0.069 ± 0.006	1.913 ± 0.015	**60.16 ± 0.25**	0.203 ± 0.004	0.171 ± 0.007	**96.36 ± 0.11**	0.020 ± 0.004	1.197 ± 0.017	73.16 ± 0.22	0.177 ± 0.025
SWAG	**0.330 ± 0.025**	**88.77 ± 0.89**	0.026 ± 0.007	**1.417 ± 0.024**	**62.45 ± 0.49**	**0.028 ± 0.003**	**0.130 ± 0.005**	**96.72 ± 0.06**	0.016 ± 0.003	0.843 ± 0.028	73.15 ± 0.44	**0.069 ± 0.012**

Table 14: The performance of VSD-Adam, VSD-SGD, and SWAG trained with AlexNet and ResNet18 on CIFAR10, CIFAR100 dataset. Results are averaged over 5 random seeds.

AlexNet	**CIFAR10**			**CIFAR100**		
	NLL	ACC	ECE	NLL	ACC	ECE
VSD with Adam	0.656 ± 0.009	78.21 ± 0.153	0.046 ± 0.003	2.241 ± 0.026	46.85 ± 0.99	0.112 ± 0.010
VSD with SGD	**0.579 ± 0.011**	**80.41 ± 0.275**	**0.010 ± 0.002**	**1.934 ± 0.018**	**51.68 ± 0.87**	0.091 ± 0.009
SWAG	0.651 ± 0.014	78.14 ± 0.514	0.059 ± 0.003	**1.958 ± 0.051**	49.81 ± 0.62	**0.028 ± 0.009**

ResNet18	**CIFAR10**			**CIFAR100**		
	NLL	ACC	ECE	NLL	ACC	ECE
VSD with Adam	0.464 ± 0.019	87.44 ± 0.497	0.061 ± 0.005	1.504 ± 0.011	60.15 ± 0.20	0.116 ± 0.002
VSD with SGD	0.395 ± 0.022	87.17 ± 0.785	**0.018 ± 0.004**	1.440 ± 0.014	60.33 ± 0.45	**0.015 ± 0.001**
SWAG	**0.330 ± 0.025**	**88.77 ± 0.889**	0.026 ± 0.007	**1.417 ± 0.024**	**62.45 ± 0.49**	0.028 ± 0.003

J Further discussions and investigations

J.1 Discussion of VSD and non-variational methods such as SWAG

In comparison with the non-variational baseline such as SWAG in Tables 3-4-5, VSD shows better performances on both predictive accuracy and uncertainty calibration on moderate deep models (FCs, LeNet). However, on more modern architectures (AlexNet, ResNet18), VSD performs worse than SWAG with noticeable gaps, especially on CIFAR10 and CIFAR100 dataset. Even on computational complexity, SWAG is also more appealing with no significant computational overhead compared to the conventional training schemes. Also, several recent reports have shown much better performances of SWAG and Deep ensemble than some variational inference methods (BBB, MC Dropout, etc.) in the field of deep learning uncertainty. Arguably, by considerable improvements of VSD compared to other variational methods, at least in the frame of reference for SWAG (and Deep ensemble), our method has made an important step to close the gap with non-variational baselines.

On the other hand, another important aspect worth noting that in our work, VSD is introduced as a general-purpose approach for Bayesian inference and in particular for BNNs. From that, we were not aiming for a state-ofthe-art in deep learning uncertainty (of SWAG baseline), but we instead proceed to demonstrate the effectiveness of VSD on many typical fronts: generalization, uncertainty calibration, robustness (OOD detection). We wish to emphasize that variational inference (VI) methods for BNNs always have a regime of their own for the meaningful problems they aim to address (compression or model selection for instance).

J.2 Some additional investigations for VSD and improvements

VI represents a large Bayesian subfield on its own, although it unfortunately tends to lag behind non-VI methods when it comes to uncertainty in deep learning. Many questions for it remain controversial such as what is the best configuration for VI-BNN: weight parameterization, prior distribution, true and approximate posterior. And VSD obviously is also included in these problems. Inspired by the underlying principle of SWAG, which was developed on the previous work SWA - an optimization procedure guided by the Bayesian theoretical analysis of the stationary distribution of SGD iterations, we would investigate a similar perspective of optimization strategy for VSD. Indeed, optimizing a variational objective of BNN, in particular to VSD, has been a specialized problem. The current common algorithms, which adopt conventional gradients to give the direction of steepest descent of parameters in Euclidean space, might cause an unexpected difference in distribution space. This does not coincide with our original purpose of minimizing the distance between two high dimensional distributions in terms of KL divergence. In the literature of Bayesian deep learning, there are some relevant works using natural gradient [56, 68], which follows the direction of steepest descent in Riemannian space rather than Euclidean space, as a more appropriate solution. However, leveraging these algorithms is non-trivial that requires lots of complicated modifications in methodology and codebases.

We reckon that current optimization algorithms for VI in BNNs have not yet had an adequate theoretical investigation. A specific optimizer is non-ultimate and could have its own influence. Therefore, instead of using Adam optimizer for VSD as in our entire experiments, we here adopt

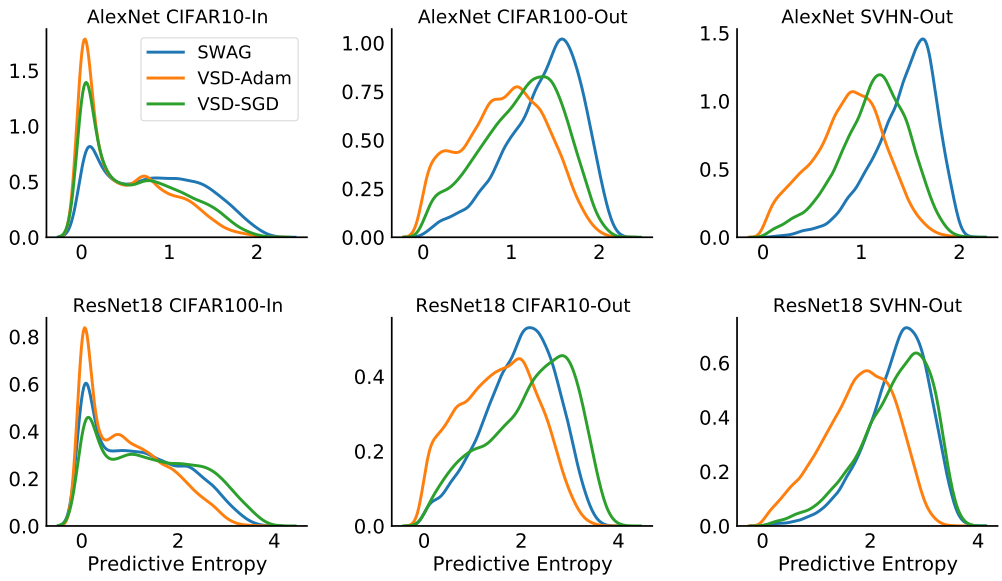


Figure 9: Histograms of predictive entropy for AlexNet (top) and ResNet18 (bottom) trained on CIFAR10 and CIFAR100 respectively.

SGD for VSD to elaborate some empirical observations and to compare with SWAG. We provide below some results as follows:

Classification results. We trained VSD by SGD-momentum in the experiment using AlexNet and ResNet18 with CIFAR10 and CIFAR100 dataset (SWAG outperforms VSD mainly on these settings in our paper). The results is given in Table 14, VSD with SGD gains better performances than SWAG on AlexNet, meanwhile on ResNet18 VSD with SGD shows slight improvements compared to using Adam. These observations consolidate our arguments of the influence of different optimizers, and specifically in this experiments SGD is more effective than Adam. Thus, understanding and devising efficient new optimization algorithms for VI in BNNs is necessary and promising.

Predictive entropy performance. We evaluated the predictive entropy performance of VSD-SGD compared to VSD-Adam and SWAG. We utilized the same settings in Figure 4 and show the new results in Figure 9. Interestingly, we find that VSD with SGD not only reduces the gap with SWAG on quantitative metrics, but also exhibits similar behavior on qualitative metrics. In the comparison between SWAG and VSD-Adam, we could observe a common trend, of which SWAG tends to maintain sufficient confidence on in-distribution data and provide higher uncertainties on out-of-distribution data (even on AlexNet, SWAG seems to be underconfident for in-distribution data). On the other hand, when using SGD optimizer, VSD archives a better balance (more well-calibrated) than VSD-Adam and SWAG on AlexNet, meanwhile on ResNet VSD-SGD tends to resemble the SWAG's behavior.

We once again argue that the optimization strategy could provide a compelling explanation for this phenomenon.

1. SWAG uses SGD with a cyclical or constant learning rate schedule to explore an optimal set of points corresponding to high-performing networks within a basin of attraction. SWAG then estimates the first two moments of the points in this optimal set to form an empirical mean and covariance of a Gaussian distribution. This means that Monte Carlo sampled models from the SWAG posterior have meaningfully different representations which provide complementary explanations for the data. Thus, when ensembling these models at the test time, we could obtain an average predictive probability that is not concentrated excessively on some certain classes (on in-distribution data, due to the high-performing essence, this average representation is still compelling for prediction). As a result, we get predictive entropy with larger values and thus explain the above behavior of SWAG.

2. For VSD-SGD, there is some evidence for its performance including: firstly, SGD minima is located in the same basin of attraction with the mean of SWAG posterior, especially SGD generally converges to a flat optima near the boundary of the manifold of the optimal set we mentioned above,

thus the local geometric information around SGD optima probably contains some similar properties as SWAG mean; and secondly, SGD with the implicit regularization has been proven to often find drastically different (and more desirable) solutions than Adam-like adaptive methods [85].

Conclusion. In the accuracy regime, many studies have pointed out the favor of SGD over adaptive optimizers [34, 93]. Our above observations further also advocate using SGD in particular for uncertainty calibration. This is consensual to the deep learning uncertainty community, in which SGD usually has been adopted as an appropriate baseline rather than Adam. Furthermore, we do believe that variational inference has been a compelling direction for learning BNNs, and exploring efficient optimization strategies to it is very promising for future works.

K Changes in the camera-ready version compared to the submitted version

- We added one more contribution in Section 1 to further highlight the primary purpose of our work.
- In Section 3.2, we provided more discussions about the plausibility of the new variational objective, as well as gave concrete intuitions on the choice of the number of transformation steps T. Some minor suggestions from reviewers were also added.
- In Section 3.3, we included the detailed description hyper-distribution $p(z)$ and $q(z)$.
- In Appendix A.2, we updated the role of hierarchical parameterization in terms of enforcing a stronger regularization.
- In Appendix B, we provided discussion about the effect of Empirical Bayes in our method.
- In Appendix D, we analyzed more thoroughly the effect of T on VSD's practical runtime.
- In Appendix I.1, we provided more insights about the KL annealing/tempered posterior.
- In Appendix I.4, we made it clear why we only did Bayesian inference for fully connected layers of LeNet architecture.
- In Appendix I.5, we provided some additional comparisons between VSD and VOGN.
- In Appendix I.6, we added the standard deviations for empirical classification results in Tables 3-4-5.
- In Appendix J, we provided further discussions of VSD and SWAG, and also introduced some meaningful investigations.

IMPROVING RELATIONAL REGULARIZED AUTOENCODERS WITH SPHERICAL SLICED FUSED GROMOV WASSERSTEIN

Khai Nguyen
VinAI Research, Vietnam
v.khainb@vinai.io

Son Nguyen
VinAI Research, Vietnam
v.son3@vinai.io

Nhat Ho*
University of Texas, Austin
VinAI Research, Vietnam
minhnhat@utexas.edu

Tung Pham
VinAI Research, Vietnam
v.tungph4@vinai.io

Hung Bui
VinAI Research, Vietnam
v.hungbh1@vinai.io

ABSTRACT

Relational regularized autoencoder (RAE) is a framework to learn the distribution of data by minimizing a reconstruction loss together with a relational regularization on the latent space. A recent attempt to reduce the inner discrepancy between the prior and aggregated posterior distributions is to incorporate sliced fused Gromov-Wasserstein (SFG) between these distributions. That approach has a weakness since it treats every slicing direction similarly, meanwhile several directions are not useful for the discriminative task. To improve the discrepancy and consequently the relational regularization, we propose a new relational discrepancy, named *spherical sliced fused Gromov Wasserstein* (SSFG), that can find an important area of projections characterized by a von Mises-Fisher distribution. Then, we introduce two variants of SSFG to improve its performance. The first variant, named *mixture spherical sliced fused Gromov Wasserstein* (MSSFG), replaces the vMF distribution by a mixture of von Mises-Fisher distributions to capture multiple important areas of directions that are far from each other. The second variant, named *power spherical sliced fused Gromov Wasserstein* (PSSFG), replaces the vMF distribution by a power spherical distribution to improve the sampling time in high dimension settings. We then apply the new discrepancies to the RAE framework to achieve its new variants. Finally, we conduct extensive experiments to show that the new proposed autoencoders have favorable performance in learning latent manifold structure, image generation, and reconstruction.

1 INTRODUCTION

In recent years, autoencoders have been used widely as important frameworks in several machine learning and deep learning models, such as generative models (Kingma & Welling, 2013; Tolstikhin et al., 2018; Kolouri et al., 2018) and representation learning models (Tschannen et al., 2018). Formally, autoencoders consist of two components, namely, an encoder and a decoder. The encoder denoted by E_ϕ maps the data, which is presumably in a low dimensional manifold, to a latent space. Then the data could be generated by sampling points from the latent space via a prior distribution p, then decoding those points by the decoder G_θ. The decoder is formally a function from latent space to the data space and it induces a distribution p_{G_θ} on the data space. In generative modeling, the major task is to obtain a decoder G_{θ^*} such that its induced distribution $p_{G_{\theta^*}}$ and the data distribution are very close under some discrepancies. Two popular instances of autoencoders are the variational autoencoder (VAE) (Kingma & Welling, 2013), which uses KL divergence, and the Wasserstein autoencoder (WAE) (Tolstikhin et al., 2018), which chooses the Wasserstein distance (Villani, 2008) as the discrepancy between the induced distribution and the data distribution.

*The work was finished when Nhat Ho worked at VinAI Research in the summer of 2020.

In order to implement the WAE, a relaxed version was introduced by removing the constraint on the prior and the aggregated posterior (latent code distribution). In particular, a chosen discrepancy between these distributions is added to the objective function and plays a role as a regularization term. With that relaxation approach, the WAE becomes a flexible framework for customized choices of the discrepancies (Patrini et al., 2020; Kolouri et al., 2018). However, the WAE suffers either from the over-regularization problem when the prior distribution is too simple (Dai & Wipf, 2018; Ghosh et al., 2019), which is usually chosen to be isotropic Gaussian, or from the under-regularization problem when learning an expressive prior distribution jointly with the autoencoder without additional regularization, e.g., structural regularization (Xu et al., 2020). In order to circumvent these issues of WAE, relational regularized autoencoder (RAE) was proposed in (Xu et al., 2020) with two major changes. The first change is to use a mixture of Gaussian distributions as the prior while the second change is to set a regularization on the structural difference between the prior and the aggregated posterior distribution, which is called the relational regularization. The state-of-the-art version of RAE, deterministic relational regularized autoencoder (DRAE), utilizes the sliced fused Gromov Wasserstein (SFG) (Xu et al., 2020) as the relational regularization. Although DRAE performs well in practice and has good computational complexity (Xu et al., 2020), the SFG does not fully exploit the benefits of relational regularization due to its slicing drawbacks. Similar to sliced Wasserstein (SW) (Bonnotte, 2013; Bonneel et al., 2015) and sliced Gromov Wasserstein (SG) (Vayer et al., 2019), SFG uses the uniform distribution over the unit sphere to sample projecting directions. However, that leads to the underestimation of the discrepancy between two target distributions (Deshpande et al., 2019; Kolouri et al., 2019) since many unimportant directions are included in that estimation. A potential solution is by using only the best Dirac measure over the unit sphere to sample projecting directions in SFG, which was employed in max-sliced Wasserstein distance Deshpande et al. (2019). However, this approach focuses on the discrepancy of the target probability measures based on only one important direction while other important directions are not considered. As one alternative solution, authors in (Nguyen et al., 2021) proposed the distributional slicing approach which is a general technique to design a probabilistic way to select important directions.

Our contributions. To improve the effectiveness of the relational regularization in the autoencoder framework, we propose novel sliced relational discrepancies between the prior and the aggregated posterior. The new sliced discrepancies utilize von Mises-Fisher distribution and its variants instead of the uniform distribution as the distributions over slices. An advantage of the vMF distribution and its variants is that they could interpolate between the Dirac measure and uniform measure, thereby improving the quality of the projections sampled from these measures and overcoming the weaknesses of both the SFG and its max version—max-SFG. In summary, our major contributions are as follows:

1. First, we propose a novel discrepancy, named *spherical sliced fused Gromov Wassersetein* (SSFG). This discrepancy utilizes vMF distribution as the slicing distribution to focus on the area of directions that can separate the target probability measures on the projected space. Moreover, we show that SSFG is a well-defined pseudo-metric on the probability space and does not suffer from the curse of dimensionality for the inference purpose. With favorable theoretical properties of SSFG, we apply it to the RAE framework and obtain a variant of RAE, named *spherical deterministic RAE* (s-DRAE).
2. Second, we propose an extension of SSFG to *mixture SSFG* (MSSFG) where we utilize a mixture of vMF distributions as the slicing distribution (see Appendix C for the details). Comparing to the SSFG, the MSSFG is able to simultaneously search for multiple areas of important directions, thereby capturing more important directions that could be far from each other. Based on the MSSFG, we then propose another variant of RAE, named *mixture spherical deterministic RAE* (ms-DRAE).
3. Third, to improve the sampling time and stability of vMF distribution in high dimension settings, we introduce another variant of SSFG, named *power SSFG* (PSSFG), which uses power spherical distribution instead of the vMF distribution as the slicing distribution. Then, we apply the PSSFG to the RAE framework to obtain the *power spherical deterministic RAE* (ps-DRAE).
4. Finally, we carry out extensive experiments on standard datasets to show that proposed autoencoders achieve the best generative quality among well-known autoencoders, including the state-of-the-art RAE—DRAE. Furthermore, the experiments indicate that the s-DRAE, ms-DRAE, and ps-DRAE can learn a nice latent manifold structure, a good mixture of

Gaussian prior which can cover well the latent manifold, and provide more stable results in both generation and reconstruction than DRAE.

We note in passing that the proposed sliced-fused Gromov-Wasserstein divergences are not just applicable to the applications within the RAE framework. In Appendix E.6, we provide initial experiments to show that these divergences can be used to improve over other sliced-based distances in color transfer applications. We leave a thorough extension of the proposed sliced-fused discrepancies to other applications in the future work.

Organization. The remainder of the paper is organized as follows. In Section 2, we provide backgrounds for DRAE and vMF distribution. In Section 3, we propose the spherical sliced fused Gromov Wasserstein and its extension. We then apply these spherical discrepancies to the relational regularized autoencoder. Extensive experiment results are presented in Section 4 followed by conclusion in Section 5. Proofs of key results and extra materials are in the supplementary material.

Notation: Let $\mathbb{S}^{d-1}$ be the d-dimensional hypersphere and $\mathcal{U}(\mathbb{S}^{d-1})$ be the uniform distribution on $\mathbb{S}^{d-1}$. For a metric space $(\mathcal{X}, d_1)$, we denote by $\mathcal{P}(\mathcal{X})$ the space of probability distributions over $\mathcal{X}$ with finite moments. We say that d_1 is a pseudo-metric in space $\mathcal{X}$ if it is non-negative, symmetric, and satisfies the inequality: $d_1(x,z) \leq C\big[d_1(x,y) + d_1(y,z)\big]$ for a universal constant $C > 0$ and for all $x, y, z \in \mathcal{X}$. For any distribution μ and ν, $\Pi(\mu,\nu)$ is the set of all transport plans between μ and ν. For $x \in \mathbb{R}^d$, denote δ_x to be the Dirac measure at x. For any $\theta \in \mathbb{S}^{d-1}$ and any measure μ, $\theta\sharp\mu$ denotes the pushforward measure of μ through the mapping $\mathcal{R}_\theta$ where $\mathcal{R}_\theta(x) = \theta^\top x$ for all x.

2 Background

In this section, we provide backgrounds for the sliced fused Gromov Wasserstein and the relational regularized autoencoders. Then, we give backgrounds for the von Mises-Fisher distribution.

2.1 Deterministic relational regularized autoencoder and sliced Fused Gromov Wasserstein

First, we review the WAE framework (Tolstikhin et al., 2018), which is used to learn a generative model by minimizing a relaxed version of Wasserstein distance (Villani, 2008) between data distribution $p_d(x)$ and model distribution $p_\theta(x) := G_\theta\sharp p(z)$, where $p(z)$ is a noise distribution. The model aims to find the autoencoder which solves the following objective function:

$$\min_{\theta,\phi} \mathbb{E}_{p_d(x)}\mathbb{E}_{q_\phi(z|x)}[d(x, G_\theta(z))] + \lambda D(q_\phi(z)||p(z)), \tag{1}$$

where d is the ground metric of Wasserstein distance, D is a discrepancy between distributions, and $q_\phi(z|x)$ is a distribution for encoder $E_\phi : X \to Z$, parameterized by ϕ. Due to the efficiency in training generative models, several autoencoder models are derived from this framework. For example, WAE uses standard Gaussian distribution for $p(z)$ and chooses D to be either maximum mean discrepancy (MMD) or GAN. Later, by using sliced Wasserstein distance (Bonneel et al., 2015) for D, Kolouri et al. (2018) achieved another type of autoencoder, which is called SWAE.

Deterministic relational regularized autoencoder (DRAE): In DRAE, Xu et al. (2020) parametrizes the prior as a mixture of Gaussians ($p_{\mu_{1:k},\Sigma_{1:k}}(z)$) and makes it learnable. Additionally, they introduce the sliced fused Gromov Wasserstein as the discrepancy between the posterior and the prior distributions.

Definition 1. *(SFG) Let $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$ be two probability distributions, β be a constant in $[0,1]$, and $d_1 : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ be a pseudo-metric on $\mathbb{R}$. The **sliced fused Gromov Wasserstein** (SFG) between μ and ν is defined as:*

$$SFG(\mu,\nu;\beta) := \mathbb{E}_{\theta\sim\mathcal{U}(\mathbb{S}^{d-1})}[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta, d_1)], \tag{2}$$

*where the **fused Gromov Wasserstein** D_{fgw} is given by:*

$$D_{fgw}(\theta\sharp\mu,\theta\sharp\nu;\beta,d_1) := \min_{\pi\in\Pi(\theta\sharp\mu,\theta\sharp\nu)} \Big\{(1-\beta)\int_{\mathbb{R}^d\times\mathbb{R}^d} d_1(\theta^\top x, \theta^\top y)d\pi(x,y) + \beta\int_{(\mathbb{R}^d)^4} \big[d_1(\theta^\top x,\theta^\top x') - d_1(\theta^\top y,\theta^\top y')\big]^2 d\pi(x,y)d\pi(x',y')\Big\}. \tag{3}$$

Given the definition of SFG, the objective function of the deterministic relational regularized autoencoder (DRAE) takes the following form:

$$\min_{\theta,\phi,\mu_{1:k},\Sigma_{1:k}} \mathbb{E}_{p_d(x)}\mathbb{E}_{q_\phi(z|x)}\big[d\big(x, G_\theta(z)\big)\big] + \lambda\mathbb{E}_{q_\phi(z),p_{\mu_{1:k},\Sigma_{1:k}}(z)}SFG\big[\big(\hat{q}_N(z)||\hat{p}_N(z)\big)\big], \quad (4)$$

where $\hat{q}_N(z)$ and $\hat{p}_N(z)$ are the empirical distributions of $q_\phi(z)$ and $p_{\mu_{1:k},\Sigma_{1:k}}(z)$ respectively.

Properties of SFG: From equation (2), SFG is a linear combination of sliced Wasserstein (SW) and sliced Gromov Wasserstein (SG). In particular, SFG becomes SW and SG when $\beta = 0$ and $\beta = 1$, respectively. Hence SFG is able to take advantages of both of them. If μ and ν have n supports and uniform weights and $d_1(x, y) = (x - y)^2$, SFG has computational complexity of the order $\mathcal{O}(n \log n)$. It is because under d_1, both SW and SG have closed-form expressions (Vayer et al., 2019; Bonnotte, 2013) where the optimal transport map π in D_{fgw} (Vayer et al., 2018) can be obtained by sorting the projected supports of μ and ν.

Limitation of SFG: The major limitation of SFG is that the outer expectation with respect to $\theta \sim \mathcal{U}(\mathbb{S}^{d-1})$ in SFG is generally intractable. In practice, projections from the unit sphere are uniformly sampled and we then apply the Monte Carlo method to obtain an approximate of that expectation. However, the difference between two distributions is certainly not distributed uniformly, meaning that informative directions are mixed up with many non-informative ones. Hence, sampling blindly slices in high dimensional space not only is ineffective but also underestimates the discrepancy between the two distributions. The von Mises-Fisher (vMF) distribution provides a way to have concentrated weight on the most important directions and assigns less weight to further directions. Therefore, we gain a better representation of the discrepancy between probability measures.

2.2 VON MISES-FISHER DISTRIBUTION

Now, we review the definition of the von Mises-Fisher distribution.

Definition 2. *The von Mises–Fisher distribution (*vMF*) is a probability distribution on the unit sphere $\mathbb{S}^{d-1}$ where its density function is given by (Jupp et al., 1979):*

$$f(x|\epsilon, \kappa) := C_d(\kappa)\exp(\kappa\epsilon^\top x), \quad (5)$$

where $\kappa \geq 0$ is the concentration parameter, $\epsilon \in \mathbb{S}^{d-1}$ is the location vector, and $C_d(\kappa) := \frac{\kappa^{d/2-1}}{(2\pi)^{d/2} I_{d/2-1}(\kappa)}$ is the normalization constant. Here, $I_{d/2-1}$ is the modified Bessel function of the first kind at order $d/2 - 1$ (Temme, 2011).

The vMF concentrates around mode ϵ and its density decreases when x goes away from ϵ. When $\kappa \to 0$, vMF converges to the uniform distribution, and when $\kappa \to \infty$, vMF approaches to the Dirac distribution centered at ϵ (Sra, 2016). These properties are illustrated by a toy example in Figure 3 in Appendix B.1.

3 SPHERICAL SLICED FUSED GROMOV WASSERSTEIN AND ITS RELATIONAL REGULARIZED AUTOENCODER

In this section, we introduce a novel discrepancy, named *spherical sliced fused Gromov Wasserstein* (SSFG), that searches for the best vMF distribution which distributes more masses to the most important area of projections on the unit sphere $\mathbb{S}^{d-1}$. Then, we discuss an application of SSFG to the relational regularized autoencoder framework.

3.1 SPHERICAL SLICED FUSED GROMOV WASSERSTEIN

We first start with a definition of spherical sliced fused Gromov Wasserstein.

Definition 3. *(SSFG) Let $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$ be two probability distributions, $\kappa > 0$, $\beta \in [0, 1]$, $d_1 : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ be a pseudo-metric on $\mathbb{R}$. The **spherical sliced fused Gromov Wasserstein** (SSFG) between μ and ν is defined as follows:*

$$SSFG(\mu, \nu; \beta, \kappa) := \max_{\epsilon \in \mathbb{S}^{d-1}} \mathbb{E}_{\theta \sim \text{vMF}(\cdot|\epsilon,\kappa)}\big[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta, d_1)\big], \quad (6)$$

where the fused Gromov Wasserstein D_{fgw} is defined at equation (3).

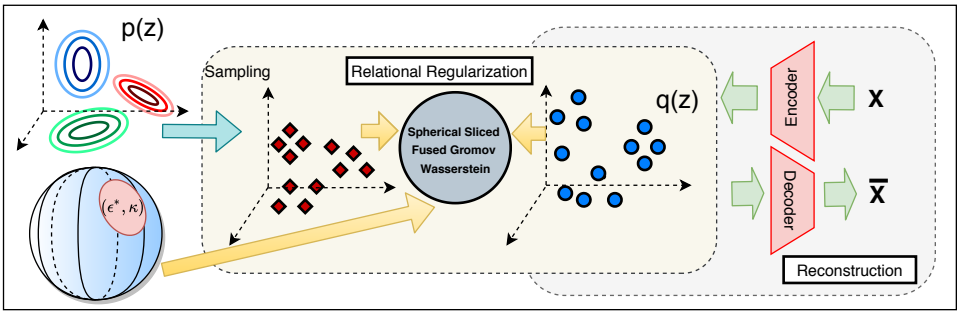


Figure 1: Visualization of the spherical deterministic relational regularized autoencoder (s-DRAE).

A few comments on the SSFG are in order. The family of vMF distributions is controlled by two parameters ϵ and κ, where ϵ is the mode of the vMF distribution while κ controls its concentration. By changing κ from 0 to infinity, the vMF family could interpolate from the uniform distribution to any Dirac distribution on the sphere. In other words, it allows us to control distributing weight to the most important direction and other directions based on the geodesic distance on the sphere. Optimizing over the family of vMF distributions helps us to identify where the best direction is as well as how much weight we need to put there in comparison with other less important directions.

Sampling procedure and reparameterization trick with the vMF: To generate samples from vMF, we follow the procedure in (Ulrich, 1984), which is described in Algorithm 1 in Appendix B. Note that this procedure does not suffer from the curse of dimensionality. Furthermore, to compute integral with respect to the vMF distribution, we use the reparametrization scheme in (Naesseth et al., 2017), which was extended for vMF in Lemma 2 in (Davidson et al., 2018). Finally, Davidson et al. (2018) proved that samples from Algorithm 1 (cf. Appendix B.2) can provide a differentiable estimator for the parameters of vMF distribution. More details of this scheme are given in Appendix B.3.

Complexity of computing SSFG: Let μ and ν be two discrete distributions that have n supports with uniform weights. For the general case of d_1, similar to SFG, the complexity of computing SSFG can be expensive (at least of the order $\mathcal{O}(n^4)$ as the fused Gromov Wasserstein D_{fgw} is a quadratic programming problem). However, the complexity of SSFG can be greatly improved under specific choices of d_1. For example, when $d_1(x, y) = (x - y)^2$, with a similar argument to the SFG case, the SSFG has computational complexity of the order $\mathcal{O}(n \log n)$.

Key properties of SSFG: We first prove that SSFG is a pseudo-metric on the probability space.

Theorem 1. *For any $\beta \in [0, 1]$ and $\kappa > 0$, SSFG$(., .; \beta, \kappa)$ is a pseudo-metric in the space of probability measures, namely, it is non-negative, symmetric, and satisfies the weak triangle inequality.*

The proof of Theorem 1 is in Appendix A.1. Our next result establishes relations between SSFG and SFG and the max version of SFG, named as max-SFG.

Theorem 2. *For any probability measures $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$, the following holds:*

(a)
$$\lim_{\kappa \to 0} \textit{SSFG}(\mu, \nu; \beta, \kappa) = \textit{SFG}(\mu, \nu; \beta),$$
$$\lim_{\kappa \to \infty} \textit{SSFG}(\mu, \nu; \beta, \kappa) = \max_{\theta \in \mathbb{S}^{d-1}} D_{fgw}(\theta \sharp \mu, \theta \sharp \nu; \beta) := \textit{max-SFG}(\mu, \nu; \beta).$$

(b) For any $\kappa > 0$, we find that
$$\exp(-\kappa) C_d(\kappa) \textit{SFG}(\mu, \nu; \beta) \leq \textit{SSFG}(\mu, \nu; \beta, \kappa) \leq \exp(\kappa) C_d(\kappa) \textit{SFG}(\mu, \nu; \beta),$$
$$\textit{SSFG}(\mu, \nu; \beta, \kappa) \leq \textit{max-SFG}(\mu, \nu; \beta).$$

The proof of Theorem 2 is in Appendix A.2. Theorem 2 shows that SSFG is an interpolation between SFG and max-SFG, namely, it combines the properties of both SFG and max-SFG. Furthermore, the result of part (b) of Theorem 2 indicates that SSFG is strongly equivalent to SFG.

Our next result shows that SSFG does not suffer from the curse of dimensionality for the inference purpose under certain choices of d_1. Therefore, it will be a statistically efficient discrepancy to compare the prior distribution to the encoder distribution in the DRAE framework.

Theorem 3. *Assume that μ is a probability measure supported on a compact subset $\Theta \subset \mathbb{R}^d$. Let $X_1, \ldots, X_n$ be i.i.d. data from P and $d_1(x, y) = |x - y|^r$ for a positive integer r. We denote by $\mu_n = \frac{1}{n}\sum_{i=1}^n \delta_{X_i}$ the empirical measure of the data points $X_1, \ldots, X_n$. Then, for any $\beta \in [0, 1]$ and $\kappa > 0$, there exists a constant c depending only on r and diameter of Θ such that*

$$\mathbb{E}\Big[\mathit{SSFG}(\mu_n, \mu; \beta, \kappa)\Big] \leq \frac{c}{n}.$$

Theorem 3 together with the earlier argument about the computational complexity of SSFG suggests that the choice of $d_1(x, y) = (x - y)^2$ is not only convenient for the computation but also statistically efficient. Therefore, we will specifically use this choice of d_1 in our experiments in Section 4.

Spherical deterministic relational regularized autoencoder: We replace SFG by SSFG in the deterministic relational regularized autoencoder framework in equation (4) to obtain a new variant of DRAE with a stronger relational regularization. The new autoencoder is named as *spherical deterministic relational regularized autoencoder* (s-DRAE). Intuitive visualization of s-DRAE is presented in Figure 1. The detailed training procedure for s-DRAE is left in Appendix B.5.

3.2 Extensions and Variants of SSFG

We first propose an extension of SSFG to its mixture variant.

Definition 4. *(MSSFG) Let $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$ be two probability distributions, $\beta \in [0, 1]$ be a constant, $\{\alpha_i\}_{i=1}^k$ be given mixture weights, and $\{\kappa_i\}_{i=1}^k$ be given mixture concentration parameters where $k \geq 1$. Furthermore, let $d_1 : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ be a pseudo-metric on $\mathbb{R}$. Then, the **mixture spherical sliced fused Gromov Wasserstein** (MSSFG) between μ and ν is defined as follows:*

$$\begin{aligned}&\mathit{MSSFG}(\mu, \nu; \beta, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k)\\ &\qquad := \max_{\epsilon_{1:k} \in \mathbb{S}^{d-1}} \mathbb{E}_{\theta \sim \mathit{MovMF}(\cdot|\epsilon_{1:k}, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k)}\big[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta, d_1)\big], \quad (7)\end{aligned}$$

where D_{fgw} is defined in equation (3) and the mixture of vMF distributions is defined as $\mathit{MovMF}(\cdot|\epsilon_{1:k}, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k) := \sum_{i=1}^k \alpha_i \text{vMF}(\cdot|\epsilon_i, \kappa_i)$.

Comparison between MSSFG and SSFG: When $k = 1$, the MSSFG becomes SSFG. Recall that SSFG tries to search for the best location parameter in the unit sphere $\mathbb{S}^{d-1}$ that maximizes the expected value of the fused Gromov Wasserstein between the projected probability measures. Intuitively, it places a large weight on the best projection and some weights on other important projections. However, if these important projections are far from the best projection, i.e., the center of the best von Mises-Fisher distribution, their weights will be very small, which can be undesirable. To account for this issue, the mixture of von Mises-Fisher distributions aims to find k best location parameters whose weights are guaranteed to be large enough. Furthermore, when k is chosen to be sufficiently large, mixture of von Mises-Fisher distributions will give a good coverage of the unit sphere; therefore, the important directions that MSSFG can find will be able to reflect more accurate differences between the target probability distributions than those from SSFG.

Properties of MSSFG and its DRAE version: As SSFG, MSSFG is a pseudo-metric in the probability space and does not suffer from the curse of dimensionality. Its computational complexity is of the order $\mathcal{O}(n \log n)$ when μ and ν are discrete measures with n atoms and uniform weights and $d_1(x, y) = (x - y)^2$. The detailed discussion of the properties of MSSFG is in Appendix C. An application of MSSFG to the DRAE framework leads to the *mixture spherical DRAE* (ms-DRAE).

Improving computational time of (M)SSFG: Drawing the samples from the vMF distribution and its mixtures can be slow in high dimension settings, which affects the computation of (M)SSFG. To account for this issue, we propose using *power spherical distribution* (De Cao & Aziz, 2020) instead of vMF and its mixtures as the slicing distribution to improve the computational time of (M)SSFG. It leads to a new discrepancy, named *power SSFG* (PSSFG), between the probability distributions (see Appendix D for the definition). In Section 4, we show that (M)PSSFG has better computational time than (M)SSFG while its DRAE version, named *(mixture) power spherical DRAE* ((m)ps-DRAE), has comparable performance to (m)s-DRAE.

Table 1: FID scores and reconstruction losses of different autoencoders. (*) denotes the results that are taken from (Xu et al., 2020) due to the reproducing failure. The results are taken from 5 different runs.

Method	MNIST		CelebA	
	FID	Reconstruction	FID	Reconstruction
VAE	71.55 ± 26.65	18.59 ± 2.22	59.99(*)	96.36(*)
GMVAE	75.68 ±11.95	18.19 ± 0.14	212.59 ± 18.15	97.77± 0.19
Vampprior	138.03 ± 34.09	29.98± 4.09	-	-
PRAE	100.25 ±41.72	16.20 ±3.14	52.20 (*)	**63.21(*)**
WAE	80.77 ± 11	11.53 ±0.33	52.07 (*)	63.83(*)
SWAE	80.28 ±19.22	14.12 ± 2.06	86.53 ± 2.49	89.71±2.15
DRAE	58.04 ± 20.74	14.07 ± 4.31	50.09 ±1.33	66.05 ± 2.56
m-DRAE (ours)	52.92 ±13.81	13.13 ± 0.33	49.05 ± 0.93	66.30 ± 0.22
s-DRAE (ours)	47.97 ± 13.83	11.17 ±1.73	46.63 ±0.83	66.62 ± 0.51
ps-DRAE (ours)	49.15 ± 12.93	11.71 ±1.21	48.21 ±1.02	66.31 ± 0.43
mps-DRAE (ours)	44.67 ±9.98	11.01 ±1.32	46.61 ± 1.01	66.23 ± 0.56
ms-DRAE (ours)	**43.57 ± 10.98**	**11.12 ±0.91**	**46.01 ±0.91**	65.91± 0.4

4 EXPERIMENTS

In this section, we conduct extensive experiments on MNIST (LeCun et al., 1998) and CelebA datasets (Liu et al., 2015) to evaluate the performance of s-DRAE, ps-DRAE and m(p)s-DRAE with various autoencoders, including DRAE (trained by SFG), PRAE (Xu et al., 2020), m-DRAE (trained by max-SFG—see its definition in Theorem 2), VAE (Kingma & Welling, 2013), WAE (Tolstikhin et al., 2018), SWAE (Kolouri et al., 2018), GMVAE (Dilokthanakul et al., 2016), and the VampPrior (Tomczak & Welling, 2018). We use two standard scores as evaluation metrics: (i) the Frechet Inception distance (FID) score (Heusel et al., 2017) is used to measure the generative ability; (ii) the reconstruction score is used to evaluate the reconstruction performance computed on the test set. For the computational details of the FID score, we compute the score between 10000 randomly generated samples and all samples from the test set of each dataset. To guarantee the fairness of the comparison, we use the same autoencoder architecture, Adam optimizer with learning rate = 0.001, $\beta_1 = 0.5$ and $\beta_2 = 0.999$; batch size = 100; latent size = 8 on MNIST and 64 on CelebA; coefficient λ=1; fused parameter β = 0.1. We set the number of components $K = 10$ for autoencoder with a mixture of Gaussian distribution as the prior. More detailed descriptions of these settings are in Appendix F. **Comparing with other autoencoders:** We first report the performances of autoencoders on MNIST (LeCun et al., 1998) and CelebA datasets (Liu et al., 2015). Table 1 presents the FID scores and reconstruction losses of trained autoencoders. All results are obtained from five different runs and reported with empirical mean and standard deviation. On the MNIST dataset, ms-DRAE achieves the lowest scores in both FID score and reconstruction loss among all the autoencoders. In addition, s-DRAE and (m)ps-DRAE also have better scores than DRAE. On the CelebA dataset, we cannot reproduce results from VAE, PRAE, and WAE; therefore, we use the results with these autoencoders from DRAE paper (Xu et al., 2020). Table 1 suggests that ms-DRAE also obtains the lowest mean and standard deviation in FID score than other autoencoders, meanwhile its reconstruction loss is almost the same as other DRAEs. The FID scores of s-DRAE and ps-DRAE are also better than those of DRAE. These results suggest that the proposed spherical discrepancies truly improve the performances of the DRAE framework. In these experiments, we set the number of projections $L = 50$ for every sliced-discrepancy. For s-DRAE, ps-DRAE and m(p)s-DRAE (10 vMF components with uniform weights and same concentration parameters), we search for $\kappa \in \{1, 5, 10, 50, 100\}$ which gives the best FID score on the validation set of the corresponding dataset. By tuning κ, we find that the performance of both s-DRAE and ps-DRAE is close to that of DRAE when $\kappa \in \{0.001, 0.01, 0.1, 1\}$, namely, the reconstruction loss and FID score are nearly equal to the scores of DRAE. On the other extreme, when $\kappa = 100$, s-DRAE and ps-DRAE behave like m-DRAE in both evaluation metrics. Further details are given in Figures 12 and 15 in Appendices E.1 and E.3 respectively.

Detailed results including generated images, reconstruction images and visualizing latent spaces are in Appendix E.1. These results indicate that s-DRAE, ps-DRAE, and ms-DRAE can learn nice latent

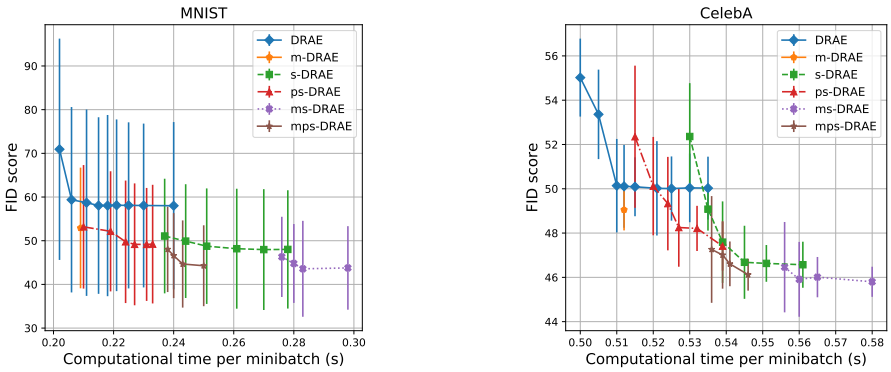


Figure 2: Each dot represents the computational time per minibatch and FID score. For DRAE, we vary the number of projections $L \in \{1, 5, 10, 20, 50, 100, 200, 500, 1000\}$; for s-DRAE we set $\kappa = 10$, $L \in \{1, 5, 10, 20, 50, 100\}$; for ps-DRAE we set $\kappa = 50, L \in \{1, 5, 10, 20, 50, 100\}$; and for m(p)s-DRAE we set $L = 50$, the number of vMF components $k \in \{2, 5, 10, 50\}$ (for each k, we find the best $\kappa \in \{1, 5, 10, 50, 100\}$).

structures and mixture Gaussian priors which can cover well these latent spaces. As a consequence, the spherical DRAEs can produce good generated images and reconstruct images correctly.

Detailed comparisons among deterministic RAEs: It is well-known that the quality and computational time of sliced-discrepancies depend on the number of projections (Kolouri et al., 2018; 2019; Deshpande et al., 2019). Therefore, we carry out experiments on MNIST and CelebA datasets to compare ps-DRAE, m(p)s-DRAE, s-DRAE to DRAE in a wide range the of number of projections. In detail, we set the number of projections $L \in \{1, 5, 10, 20, 50, 100, 200, 500, 1000\}$ in DRAE ; and $L \in \{1, 5, 10, 20, 50, 100\}$ for s-DRAE ($\kappa = 10$), and ps-DRAE ($\kappa = 50$). We then report the (minibatch) computation time and FID score of these autoencoders in Figure 2. In this figure, we also plot the time and FID scores of m-DRAE and m(p)s-DRAE (using $L = 50$ and the number of vMF components $k \in \{2, 5, 10, 50\}$. On the MNIST dataset, with only 1 projection, s-DRAE achieves lower FID score than all settings of DRAE; however, s-DRAE requires more time to train due to the sampling of vMF and its optimization problem. Having faster sampling procedure of PS distribution, ps-DRAE has better computational time than s-DRAE while it still has a comparable performance to s-DRAE. With the computational time greater than about 0.21(s), ps-DRAE always produces lower FID score than DRAE. We also observe the same phenomenon on CelebA dataset, namely, ps-DRAE and s-DRAE have lower FID score than DRAE with any value of L but $L = 1$. Between s-DRAE and ps-DRAE, s-DRAE gives better results but ps-DRAE is faster in training. On both the datasets, m-DRAE has faster speed than (p)s-DRAE but its FID score is higher. In terms of the FID score, ms-DRAE is the best autoencoder though it can be more expensive in training. Moreover, mps-DRAE has more efficient computational time than ms-DRAE while its FID scores are comparable. Finally, we observe that increasing the number of components in m(p)s-DRAE can enhance the FID score but also worsen the computational speed.

Additional experiments: We provide further comparisons between MSSFG, PSSFG, SSFG and SFG in ex-post density estimation of autoencoder (Ghosh et al., 2019) and GAN (Goodfellow et al., 2014) applications in Appendices E.4 and E.5. In the ex-post density estimation framework, we find that MSSFG, PSSFG and SSFG give better FID score than SFG. Like traditional training procedures, MSSFG achieves the best performance in this task. In the GAN application, we use a toy example, which is to learn a generator to produce 4 Gaussian modes. We observe that MSSFG, PSSFG and SSFG help the model distribution converge faster than SFG does. Finally, we also apply the proposed sliced-discrepancies to image color adaption (Rabin et al., 2014; Bonneel et al., 2015; Perrot et al., 2016) in Appendix E.6, where we find that using (M)vMF, (M)PS distributions to sample projecting directions can improve the performance of the sliced-based color adaption algorithms (Rabin et al., 2010; Bonneel et al., 2015; Muzellec & Cuturi, 2019).

5 Conclusion

In the paper, we first introduced a new spherical relational discrepancy, named spherical sliced fused Gromov Wasserstein (SFFG), between the probability measures. This discrepancy is obtained by

replacing the uniform distribution over slicing direction in sliced fused Gromov Wasserstein by a von Mises-Fisher distribution that can cover the most informative area of directions. To improve the performance and stability of SFFG, we then propose two variants of SSFG: (i) the first variant is mixture SSFG (MSSFG), obtained by using a mixture of vMF distributions instead of a single vMF distribution to capture more informative areas of directions; (ii) the second variant is power SSFG (PSSFG), obtained by replacing the vMF distribution by the power spherical distribution to improve the sampling time of vMF distribution in high dimension settings. An application of these discrepancies to the DRAE framework leads to several new variants of DRAE. Extensive experiments show that these new autoencoders are more stable and achieve better generative performance than the previous autoencoders, including DRAE, in comparable computational time.

REFERENCES

Arindam Banerjee, Inderjit S Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the unit hypersphere using von Mises-Fisher distributions. *Journal of Machine Learning Research*, 6(Sep): 1345–1382, 2005.

Sergey Bobkov and Michel Ledoux. One-dimensional empirical measures, order statistics, and Kantorovich transport distances. *Memoirs of the American Mathematical Society*, 2019.

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and Radon Wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51(1):22–45, 2015.

Nicolas Bonnotte. *Unidimensional and evolution methods for optimal transportation*. PhD thesis, Paris 11, 2013.

Charlotte Bunne, David Alvarez-Melis, Andreas Krause, and Stefanie Jegelka. Learning generative models across incomparable spaces. In *International Conference on Machine Learning*, 2019.

Bin Dai and David Wipf. Diagnosing and enhancing vae models. In *International Conference on Learning Representations*, 2018.

Tim R Davidson, Luca Falorsi, Nicola De Cao, and Thomas Kipf Jakub M Tomczak. Hyperspherical variational auto-encoders. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2018.

Nicola De Cao and Wilker Aziz. The power spherical distribution. *arXiv preprint arXiv:2006.04437*, 2020.

Ishan Deshpande, Yuan-Ting Hu, Ruoyu Sun, Ayis Pyrros, Nasir Siddiqui, Sanmi Koyejo, Zhizhen Zhao, David Forsyth, and Alexander G Schwing. Max-sliced Wasserstein distance and its use for GANs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10648–10656, 2019.

Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep unsupervised clustering with Gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

William Feller. *Introduction to Probability Theory and its Applications II*. Wiley, New York, 1966.

Mikhail Figurnov, Shakir Mohamed, and Andriy Mnih. Implicit reparameterization gradients. In *Advances in Neural Information Processing Systems*, pp. 441–452, 2018.

Partha Ghosh, Mehdi SM Sajjadi, Antonio Vergari, Michael Black, and Bernhard Scholkopf. From variational to deterministic autoencoders. In *International Conference on Learning Representations*, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pp. 6626–6637, 2017.

Peter E Jupp, Kanti V Mardia, et al. Maximum likelihood estimators for the matrix von Mises-Fisher and Bingham distributions. *The Annals of Statistics*, 7(3):599–606, 1979.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Soheil Kolouri, Phillip E Pope, Charles E Martin, and Gustavo K Rohde. Sliced Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018.

Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and Gustavo Rohde. Generalized sliced Wasserstein distances. In *Advances in Neural Information Processing Systems*, pp. 261–272, 2019.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Boris Muzellec and Marco Cuturi. Subspace detours: Building transport plans that are optimal on subspace projections. In *Advances in Neural Information Processing Systems*, pp. 6917–6928, 2019.

Christian Naesseth, Francisco Ruiz, Scott Linderman, and David Blei. Reparameterization gradients through acceptance-rejection sampling algorithms. In *Artificial Intelligence and Statistics*, pp. 489–498, 2017.

Khai Nguyen, Nhat Ho, Tung Pham, and Hung Bui. Distributional sliced-wasserstein and applications to generative modeling. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=QYjO70ACDK`.

Giorgio Patrini, Rianne van den Berg, Patrick Forre, Marcello Carioni, Samarth Bhargav, Max Welling, Tim Genewein, and Frank Nielsen. Sinkhorn autoencoders. In *Uncertainty in Artificial Intelligence*, pp. 733–743. PMLR, 2020.

Michaël Perrot, Nicolas Courty, Rémi Flamary, and Amaury Habrard. Mapping estimation for discrete optimal transport. *Advances in Neural Information Processing Systems*, 29:4197–4205, 2016.

Julien Rabin, Julie Delon, and Yann Gousseau. Regularization of transportation maps for color and contrast transfer. In *2010 IEEE International Conference on Image Processing*, pp. 1933–1936. IEEE, 2010.

Julien Rabin, Sira Ferradans, and Nicolas Papadakis. Adaptive color transfer with relaxed optimal transport. In *2014 IEEE International Conference on Image Processing (ICIP)*, pp. 4852–4856. IEEE, 2014.

Suvrit Sra. Directional statistics in machine learning: a brief review. *arXiv preprint arXiv:1605.00316*, 2016.

Nico M Temme. *Special functions: An introduction to the classical functions of mathematical physics*. John Wiley & Sons, 2011.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018.

Jakub Tomczak and Max Welling. Vae with a vampprior. In *International Conference on Artificial Intelligence and Statistics*, pp. 1214–1223, 2018.

Michael Tschannen, Olivier Bachem, and Mario Lucic. Recent advances in autoencoder-based representation learning. *arXiv preprint arXiv:1812.05069*, 2018.

Gary Ulrich. Computer generation of distributions on the m-sphere. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 33(2):158–163, 1984.

Titouan Vayer, Laetitia Chapel, Rémi Flamary, Romain Tavenard, and Nicolas Courty. Optimal transport for structured data with application on graphs. *arXiv preprint arXiv:1805.09114*, 2018.

Titouan Vayer, Rémi Flamary, Romain Tavenard, Laetitia Chapel, and Nicolas Courty. Sliced Gromov-Wasserstein. *Advances in Neural Information Processing Systems*, 2019.

C. Villani. *Optimal transport: Old and New*. Springer, 2008.

Cédric Villani. *Topics in Optimal Transportation*. American Mathematical Society, 2003.

Hongteng Xu, Dixin Luo, Ricardo Henao, Svati Shah, and Lawrence Carin. Learning autoencoders with relational regularization. *arXiv preprint arXiv:2002.02913*, 2020.

Infinite Dropout for training Bayesian models from data streams

Van-Son Nguyen[1], Duc-Tung Nguyen[1,2], Linh Ngo Van[1], Khoat Than[1,2]
[1]*Hanoi University of Science and Technology*, Hanoi, Vietnam
[2]*VinAI Research*, Hanoi, Vietnam
sonnguyenkstn@gmail.com, ductungnguyen1997@gmail.com, {linhnv, khoattq}@soict.hust.edu.vn

***Abstract*—The ability to continuously train Bayesian models in streaming environments is highly important in the era of big data. However, it has to face the famous stability-plasticity dilemma and the problem of noisy and sparse data. We propose a novel and easy-to-implement framework, called Infinite Dropout (iDropout), to address these challenges. iDropout has an easy mechanism to balance between old and new information, which allows models to trade off stability against plasticity. Thanks to the ability to reduce overfitting and the ensemble property of Dropout, our framework obtains better generalization, thus effectively handles undesirable effects of noise and sparsity. Further, iDropout is able to adapt quickly to abnormal changes in data streams. We theoretically analyze the equivalence of Dropout in iDropout to a regularizer, well applied to a much larger context than what was known before. Extensive experiments show that iDropout significantly outperforms the state-of-the-art baselines.**

***Index Terms*—Bayesian models, Data streams, Streaming learning, Dropout, Regularization**

I. Introduction

We are interested in how to efficiently train a Bayesian model in streaming conditions where the data comes continuously and infinitely. Learning from data streams requires to address the stability-plasticity dilemma [1], i.e., an algorithm should keep the learned knowledge stable while enabling the model to adapt well with sudden changes in the environments. Such a dilemma is present in most continual learning systems. Further, this learning process can be negatively affected by undesirable properties of data, including noise, which potentially causes overfitting, and sparsity, i.e., the situation when model does not have enough relevant information to make good predictions for unseen data. Our work focuses on these challenges.

Some recent studies [2]–[4] have provided excellent solutions for learning from data streams. Those methods enable Bayesian models, which are designed for static conditions, to work with data streams. However, those methods are limited when facing the above challenges. For example, we found that *streaming variational Bayes* (SVB) [2] becomes too stable after receiving a large enough amount of data. In other words, SVB makes models evolve slowly and have difficulties learning new information, thus fail to adapt sudden changes from the data stream. This is a serious problem and potentially happens in other methods, but has not been studied formally in any research before. Further, existing methods do not have any efficient way to deal with noisy and sparse data.

In this paper, we propose a novel framework called *Infinite Dropout* (iDropout) which enables a wide range of models to work in streaming environments. Our framework has several benefits. Firstly, iDropout has an easy mechanism to balance the information among old and new data throughout the data stream, which helps tackle the stability-plasticity dilemma. Secondly, we theoretically prove that Dropout in iDropout plays as a data-dependent regularizer, which allows our method to effectively overcome the overfitting issue. Moreover, with a fast approximation via a scaling factor, Dropout in our method works as an ensemble of an exponential number of learners, which is very useful in making good predictions for future data. These advantages help our method obtain better generalization. This is extremely important when data comes continuously with high uncertainty, which has the potential for sudden changes or undesirable properties such as noise and sparsity. Furthermore, our analysis about the role Dropout as regularization applies well to a large class of Bayesian models, extending existing works [5]–[10] to significantly wider contexts.

We did extensive experiments to compare iDropout with existing frameworks, using two base models: *latent Dirichlet allocation* (LDA) [11] for topic modeling and *Naive Bayes* (NB) [12] for classification. Empirical results show that our framework gives major improvements over existing state-of-the-art streaming methods on both learning tasks.

Roadmap: Section 2 briefly provides closely related work. We formally describe the iDropout framework and its applications in Section 3. Non-trivial findings about iDropout are described in Section 4. Finally, extensive evaluation appears in Section 5.

II. Related work

Recently, a lot of effort has been made to adapt Bayesian models from static conditions to streaming ones. Some work [2], [4], [13] propose recursive updating of the variational distribution. Streaming variational Bayes (SVB) [2] uses the variational parameter from preceding time step as the parameter for the prior distribution of current time step. However, this mechanism can be inappropriate in data streams, since the variational parameter learned from past data may not describe properly the property of current data. In particular, once given enough data, SVB becomes too stable and thus unable to learn new information from the data stream. To avoid this problem, HPP [4] is proposed to exponentially forget the

978-1-7281-0858-2/19/$31.00 ©2019 IEEE

variational parameters associated with old data, where the forgetting rate is considered as a hidden variable. Unfortunately, the introduction of this new latent variable makes the model no more conjugate, which requires non-trivial efforts to infer for the complicated Bayesian models (when the forgetting rate is considered a fixed hyperparameter, the method is called SVB-PP). The second direction is to cast the inference problem as a stochastic optimization problem. Stochastic variational inference (SVI) [14] is a typical example. However, SVI conditions on a fixed dataset to reveal the variational distribution, which isn't really appropriate in data streams. Population variational Bayes (PVB) [3], a closely related framework addresses this problem by assuming that the data stream is generated by sampling α data points from the population distribution F_α.

It is worth noting that none of these frameworks consider seriously the problem of noise and sparsity, which are pervasive in streaming environments. In order to address these problems, we consider using Dropout [15], which is a well-known stochastic regularization technique introduced in the context of feed forward neural networks. The idea of Dropout is to randomly omit a subset of features at each iteration of the training process. Dropout has two great advantages: it prevents models from overfitting by discouraging co-adaptation of features and more especially, Dropout provides an efficient way to approximately combine exponentially many models, working as a form of ensemble learning. Dropout works well for various machine learning methods, including neural networks [16], support vector machine [17], matrix factorization [18] and topic model [19]. The theory behind Dropout is considered by some recent researches [6], [8]–[10], [20]. Particularly, [6] shows that Dropout is equivalent to an L_2 regularizer when applied to generalized linear models.

III. Infinite Dropout

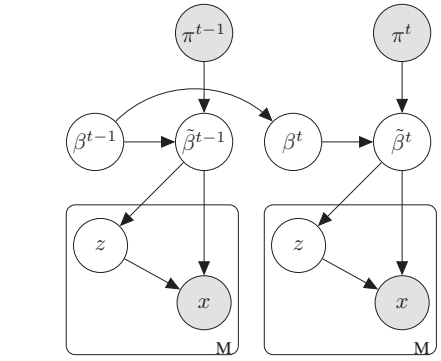
In this section, at first we present the framework for a general Bayesian model. After that, we explicitly describe applications to LDA and NB to clarify our framework.

A. The framework

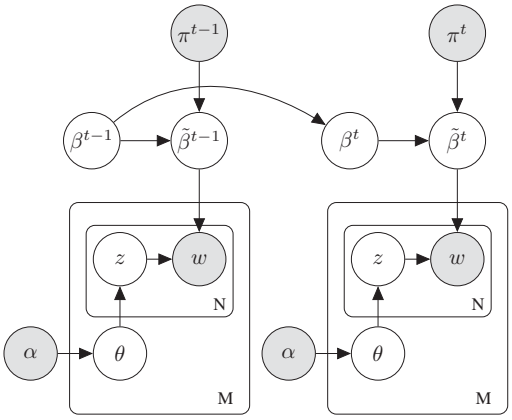
We consider a general model $B(\beta, z, x)$ [3], [14] involving observations, global variables and local variables. The global variable is matrix β which has size $K \times V$, shared among data points $x_{1:M}$, while local variable z_i only governs ith data point x_i. In traditional Bayesian methods, we condition on a fixed dataset to reveal the posterior distribution of hidden variables $p(\beta, z|x)$. Undoubtedly, this can not work with data streams where the data come in an infinite sequence of minibatches $C = \{D^1, D^2, \cdots, D^t, \cdots\}$ and each minibatch t consists of M observed data points: $D^t = \{x_1^t, x_2^t, \cdots, x_M^t\}$.

We need to extend the model to also describe the dynamics of the data stream. Here we assume that only the global variable β evolves over time, which we indicate with superscript t, i.e., β^t. We introduce a transition model $p(\beta^t|\beta^{t-1})$ to describe the transformation between two consecutive time steps:

$$p(\beta_k^t|\beta_k^{t-1}) = \mathcal{N}(.|\beta_k^{t-1}, \sigma^2 I) \quad (1)$$



(a) iDropout for $B(\beta, z, x)$



(b) iDropout for LDA

Fig. 1: Graphical representation for $iDropout$.

where k is the row index of β^{t-1} and I is the identity matrix of size V. The variance σ^2 is a hyperparameter, which describes our assumption about the fluctuation of β_k between two consecutive time steps.

Dropout is utilized in our framework as follows. In each time step t, we drop randomly a number of elements of matrix β^t. This is implemented by using a hyperparameter called drop matrix π^t to make *element-wise product* with β^t, then going through a transformation: $\tilde{\beta}^t = f(\beta^t \odot \pi^t)$. Transformation f should be chosen to assure that $\tilde{\beta}^t$ can replace β in model $B(\beta, z, x)$ at each time step t (in the later subsections, we use *softmax* to be the transformation). Given the new global variable $\tilde{\beta}^t$ at each minibatch t, the generative process of all data points is similar to the original model B (Fig. 1a). In order to keep the randomness of Dropout, we use a different drop matrix at each minibatch. Each element π_{ij}^t of π^t is generated using one of two options:

1) Bernouli dropout: $p(\pi_{ij}^t = 1) = 1 - dr$, $p(\pi_{ij}^t = 0) = dr$
2) Inverted dropout:

$$p(\pi_{ij}^t = 1/(1-dr)) = 1 - dr, p(\pi_{ij}^t = 0) = dr \quad (2)$$

in which dr is drop rate. Note that when β^t is used at test time, it has to be rescaled by $\mathbb{E}[\pi_{ij}^t]$. By doing this scaling, $2^{K \times V}$ models with shared parameters can be combined into a

Algorithm 1 Learning in iDropout

Require: Drop rate dr, variance σ^2, data sequence $\{D^1, D^2, \cdots\}$
Ensure: Global variable β
Initialize β^0 randomly.
for t^{th} minibatch with data D^t **do**
  Draw drop matrix π^t randomly
  Do inference w.r.t. the local variables z (e.g., by doing inference or sampling), given β^t and D^t
  Estimate β^t by using a gradient-based algorithm, given the statistics from z, D^t
end for

Algorithm 2 iDropout training for LDA

Require: Prior α, drop rate dr, variance σ^2, data sequence $\{D^1, D^2, \cdots\}$
Ensure: Global variable β
Initialize β^0 randomly.
for t^{th} minibatch with data D^t **do**
  Draw drop matrix π^t randomly
  for each document d in D^t **do**
    Infer (γ_d, ϕ_d) by alternatively updating (4) and (5)
  end for
  Find each β_k^t by maximizing (6)
end for

single model to be used at test time, which works as a form of ensemble learning.

Learning in iDropout is done at each minibatch t by estimating β^t through the posterior $p(\beta^t|\beta^{t-1}, \pi^t, D^t)$, where β^{t-1} is learned from the previous minibatch:

$$p(\beta^t|\beta^{t-1}, \pi^t, D^t) \propto p(\beta^t, D^t|\beta^{t-1}, \pi^t)$$
$$\propto p(\beta^t|\beta^{t-1})p(D^t|\pi^t, \beta^t) \propto p(\beta^t|\beta^{t-1})p(D^t|\tilde{\beta}^t)$$

In $\log$ form, we obtain:

$$F(\beta^t) = \log p(\beta^t|\beta^{t-1}, \pi^t, D^t)$$
$$= \log p(\beta^t|\beta^{t-1}) + \log p(D^t|\tilde{\beta}^t) + const \quad (3)$$

The learning process is composed of two phases: infer local variables and update global variables, respectively. While the inference of local variables z is inherited from the original model B (e.g., by using variational inference or sampling from $p(z|x, \tilde{\beta}^t)$), we focus on optimizing F with respect to β^t. We extract the component $G(\beta^t)$, which contains β^t, from log-likelihood $\log p(D^t|\tilde{\beta}^t)$. Then, we obtain the objective function: $F(\beta^t) = \log p(\beta^t|\beta^{t-1}) + G(\beta^t)$ and maximize it by using a gradient-based method. Algorithm 1 briefly describes the learning process.

B. Case study 1: when LDA is the base model

In this subsection, we show an application of iDropout to latent Dirichlet allocation [11], which is used for document analysis. Suppose that each minibatch t consists of M documents and each document d contains N_d words. Hyperparameter α is the parameter of Dirichlet distribution (Dir) for topic mixture θ, and the matrix β of size $K \times V$ is the topic distribution over V words in the vocabulary.

The generative process of documents in each minibatch t^{th} is as follows (Fig. 1b).

1) Draw the global variable β^t: $\beta_k^t \sim \mathcal{N}(\beta_k^{t-1}, \sigma^2 I)$
2) Calculate the topic distribution matrix:

$$\tilde{\beta}_{kj}^t = \text{softmax}(\beta_k^t \odot \pi_k^t)_j = \frac{\exp(\beta_{kj}^t \pi_{kj}^t)}{\sum_{i=1}^V \exp(\beta_{ki}^t \pi_{ki}^t)}$$

3) For each document d in minibatch t:
   a) Draw topic mixture: $\theta_d \sim Dirichlet(\alpha)$
   b) For n^{th} word in document d:
      i) Draw topic index: $z_{dn} \sim Multinomial(\theta_d)$
      ii) Draw word: $w_{dn} \sim Multinomial(\tilde{\beta}_{z_{dn}}^t)$

Learning process: As in Algorithm 1, we estimate β^t through the log-posterior:

$$\log p(\beta^t|\beta^{t-1}, \pi^t, \alpha, D^t)$$
$$= \log p(\beta^t|\beta^{t-1}) + \log p(D^t|\tilde{\beta}^t, \alpha) + const$$

As mentioned above, inference for local variables θ and z can be done by utilizing different inference methods, including variational inference and Gibbs sampling. In the experiments, we use mean-field variational inference as in the original paper [11]. For each document d: $q(\theta_d, z_d|\gamma_d, \phi_d) = q(\theta_d|\gamma_d) \prod_{n \in [N_d]} q(z_{dn}|\phi_{dn})$ with the variational distribution: $q(\theta_d|\gamma_d) = Dir(.|\gamma)$ and $q(z_{dn}|\phi_{dn}) = Mult(.|\phi_{dn})$, where γ_d and ϕ_d are variational parameters. According to [11], these parameters for each document d are updated until convergence as follow:

$$\gamma_{dk} \leftarrow \alpha_k + \sum_{n=1}^{N_d} \phi_{dnk} \text{ for } k = 1, \cdots, K \quad (4)$$

$$\phi_{dnk} \propto \exp(\mathbb{E}_q[\log \theta_{dk}] + \sum_{v=1}^{V} \mathbb{I}[w_{dn} = v] \log \beta_{kv}) \quad (5)$$

where $[V] = \{1, \cdots, V\}$, $\mathbb{I}[.]$ is the indicator function. Extracting $G(\beta^t)$ from $\log p(D^t|\beta^t, \alpha)$, we obtain the objective function. Since the topics are independent of each other, we only consider the objective function with respect to β_k^t:

$$F(\beta_k^t) = \log p(\beta_k^t|\beta_k^{t-1}) + \sum_{d=1}^{M} \sum_{n=1}^{N_d} \log p(w_{dn}|z_{dn}, \tilde{\beta}^t)$$
$$= -\frac{1}{2\sigma^2}||\beta_k^t - \beta_k^{t-1}||^2 + \sum_{d=1}^{M} \sum_{n,j=1}^{N_d,V} \phi_{dnk} \mathbb{I}[w_{dn} = j] \log \tilde{\beta}_{kj}^t$$
$$= -\frac{1}{2\sigma^2}||\beta_k^t - \beta_{k-1}^t||^2 + \sum_{d=1}^{M} \sum_{n,j=1}^{N_d,V} \phi_{dnk} I[w_{dn} = j] \beta_{kj}^t \pi_{kj}^t$$
$$- \sum_{d=1}^{M} \sum_{n,j=1}^{N_d,V} \phi_{dnk} I[w_{dn} = j] \log(\sum_{i=1}^{V} \exp(\beta_{ki}^t \pi_{ki}^t)) \quad (6)$$

The objective function F is guaranteed to be concave. In deed, $-\frac{1}{2\sigma^2}||\beta_k^t - \beta_k^{t-1}||^2$ and $\beta_{kj}^t \pi_{kj}^t$ are obviously concave with respect to β_k^t, while the log-sum-exp is also a well-known convex function. Therefore, $F(\beta_k^t)$ is concave with respect to β_k^t, and we can find its maximum by applying gradient ascent on F. We sum up the learning algorithm of iDropout for LDA in Algorithm 2.

C. Case study 2: when NB is the base model

We use Multinomial Naive Bayes (NB) [12] for document classification. Suppose that each minibatch consists of M documents, each document d contains N_d words and belongs to a class $c_d \in \{1, 2, \cdots C\}$. Each c_d is generated by: $c_d \sim Mult(\alpha)$ in which α is a fixed symmetric vector, and finally β of size $C \times V$ is the class distribution over V words in the vocabulary.

The generative process for each minibatch t is as follows. Firstly, draw the global variable β^t: $\beta_c^t \sim \mathcal{N}(\beta_c^{t-1}, \sigma^2 I)$ and calculate the class matrix: $\tilde{\beta}_{cj}^t = \text{softmax}(\beta_c^t \odot \pi_c^t)_j$. Each document d is drawn by first choosing the class label $c_d \sim Mult(\alpha)$ and then drawing n^{th} word $w_{dn} \sim Mult(\tilde{\beta}_{c_d}^t)$.

Learning process: From (3), we extract the term associated with β^t for each class c:

$$
\begin{aligned}
F(\beta_c^t) &= \log p(\beta_c^t | \beta_c^{t-1}) + \sum_{d \in D_c^t} \sum_{n=1}^{N_d} \log p(w_{dn} | c_d, \tilde{\beta}^t) \\
&= -\frac{1}{2\sigma^2} ||\beta_c^t - \beta_c^{t-1}||_2^2 + \sum_{d \in D_c^t} \sum_{n=1}^{N_d} \sum_{j=1}^{V} \mathbb{I}[w_{dn} = j] \log \tilde{\beta}_{cj}^t \\
&= -\frac{1}{2\sigma^2} ||\beta_c^t - \beta_c^{t-1}||_2^2 + \sum_{d \in D_c^t} \sum_{n=1}^{N_d} \sum_{j=1}^{V} \mathbb{I}[w_{dn} = j] \beta_{cj}^t \pi_{cj}^t \\
&\quad - N_c \log\big(\sum_{i \in [V]} \exp(\beta_{ci}^t \pi_{ci}^t)\big)
\end{aligned}
$$

where D_c^t includes all documents which belong to class c, N_c is the total number of words in all documents belonging to class c. Learning for NB is very simple. At each minibatch t, we use gradient ascent to maximize $F(\beta_c^t)$ with respect to β_c^t.

IV. Discussions about iDropout

This section shows our non-trivial findings about iDropout. We compare the behavior of iDropout and other frameworks for data streams, and also consider the theory behind the effect of Dropout.

A. The stability-plasticity dilemma

In this subsection, we investigate how different streaming learning frameworks trade off stability against plasticity in models similar to LDA[1], i.e., how they balance between old and new information from data streams. In particular, SVB [2] uses the variational parameter of the global variable β^t at time step t, which we denote by λ^t, as the parameter in the Dirichlet prior distribution at time step $t+1$. In other words, for each $k \in \{1, \cdots, K\}$, β_k^{t+1} has the prior distribution $Dir(\beta_k^{t+1} | \lambda_k^t)$. Then we have:

Theorem 1. *In SVB:* $\mathbb{E}_{Dir}[\beta_{kj}^{t+1}] = \beta_{kj}^t$ *and* $Var_{Dir}[\beta_{kj}^{t+1}] \to 0$ *as* $t \to \infty$.

Proof. SVB [2] proposes recursive updating of the variational distribution. For LDA (conjugate models, exponential family, i.i.d. data), the variational parameter λ^t of global variable β^t is updated by: $\lambda^t = \lambda^{t-1} + \tilde{\lambda}^t$, where λ^{t-1} is made available from the previous minibatch and $\tilde{\lambda}^t$ is the learned information from the current minibatch. In other words, λ^t is addition of the learned information from all previous steps:

$$\lambda^t = \tilde{\lambda}^t + \cdots + \tilde{\lambda}^1 + \lambda^0$$

where:

$$
\begin{aligned}
|||\tilde{\lambda}^t|||_1 &= \sum_{k=1}^{K} \sum_{j=1}^{V} \tilde{\lambda}_{kj}^t = \sum_{d \in D^t} \sum_{n=1}^{N_d} \sum_{k=1}^{K} \sum_{j=1}^{V} \phi_{dnk} \mathbb{I}[w_{dn} = j] \\
&= \sum_{d \in D^t} \sum_{n=1}^{N_d} \sum_{k=1}^{K} \phi_{dnk} = \sum_{d \in D^t} N_d \geq 1
\end{aligned}
$$

Therefore, $||\lambda^t||_1 = \sum_{i=1}^{T} \sum_{d \in D^t} N_d \geq t$, which approaches infinity as t goes to infinity. When a new minibatch $t+1$ arrives, λ^t will be used as the parameter of the prior: $p(\beta_k^{t+1} | \lambda_k^t) = Dir(.|\lambda_k^t)$. This distribution has the expectation:

$$\mathbb{E}_{Dir}[\beta_k^{t+1}] \propto \lambda_k^t = \beta_k^t$$

and the variance:

$$\text{Var}[\beta_{kj}^{t+1}] = \frac{\lambda_{kj}^t (\sum_{i=1}^{V} \lambda_{ki}^t - \lambda_{kj}^t)}{(\sum_{i=1}^{V} \lambda_{ki}^t)^2 (\sum_{i=1}^{V} \lambda_{ki}^t + 1)}$$

which varies inversely with size of λ_k^t. As $t \to \infty$, leading to $||\lambda^t||_1 \to \infty$, we have $\text{Var}[\beta_{kj}^{t+1}] \to 0$. □

This problem is potentially present in SVB-PP [4], albeit λ^t takes longer to accumulate: $\lambda^t = \rho \lambda^{t-1} + (1-\rho)\eta + \tilde{\lambda}^t$, where ρ is the forgetting factor ($0 < \rho < 1$) and η is the uninformative prior.

When this happens, SVB and SVB-PP expect the model at time $t+1$ to be nearly identical to the model at time t. This phenomenon essentially says that a model will evolve very slowly and have difficulties in learning new information, thus could not deal well with sudden changes in the environment.

iDropout does not encounter this problem. In iDropout, we have an easy mechanism to balance the information between old and new data. Indeed, to maximize the objective function $F(\beta_k^t) = -\frac{1}{2\sigma^2}||\beta_k^t - \beta_k^{t-1}||_2^2 + \log p(D^t | \tilde{\beta}_k^t)$ in (3), we need to consider both components. While the first term encourages new model β^t to fluctuate around the previously learned β^{t-1}, the latter allows model to accommodate information from new data D^t. In other words, iDropout helps model to flexibly learn new information, while retaining relevant information from historical observations to maintain the stability.

The balance ability of iDropout is easily controlled by the variance σ^2. The bigger σ^2 is, the more we focus on learning

[1]Such models require the global variable β to be in a simplex, e.g., NB.

new information, rather than keeping old information, and vice versa. This balance is unchanged throughout the learning process. Unlike iDropout, SVB and SVB-PP cannot control this balance. Particularly, in LDA, SVB and SVB-PP becomes too rigid and unable to learn new information after receiving a large amount of data, due to the reason mentioned above.

B. The role of Dropout in iDropout

In streaming environments, the problem of noisy and sparse data is unavoidable. Specifically, learning from noisy data potentially makes models become overfitting, while sparsity in data may not provide enough relevant information to make good predictions for unseen data, both leading to poor performance.

To overcome these challenges, we propose to utilize Dropout by omitting randomly a number of elements of the global variable β^t at each time step t. Dropout in our framework has two main roles. Firstly, we theoretically prove that it plays as a data-dependent regularizer, which makes iDropout more robust against overfitting. Moreover, in our framework, Dropout is used throughout the data stream, leading to a special effect, which is ensemble learning. Indeed, at each time step in training process, the use of Dropout is equivalent to sampling a single learner from a set of $2^{K\times V}$ possible learners. Then, by rescaling β^t with $\mathbb{E}[\pi^t]$, $2^{K\times V}$ learners with shared parameters can be combined into a single learner to be used at test time.

The ability to prevent overfitting and the ensemble property make iDropout have better generalization on future data, which is specially important in streaming learning, because data streams can be non-stationary and have high uncertainty.

C. Dropout as regularization

We examine the theory behind the effect of Dropout in iDropout for two models LDA and NB.

Theorem 2. *For LDA and NB, Dropout in iDropout is equivalent to a L2-regularization $R(\beta)$:*

$$R(\beta) = \frac{dr}{2(1-dr)} \sum_{k=1}^{K}\sum_{j=1}^{V}\left[\mu_{kj}(1-\mu_{kj})\sum_{i=1}^{V} u_{kj}\right]\beta_{kj}^2$$

in which μ_{kj} is the model probability and:

$$u_{kj} = \begin{cases} \sum_{d=1}^{M}\sum_{n=1}^{N_d}\phi_{dnk}\mathbb{I}[w_{dn}=j] \text{ in LDA} \\ \sum_{d\in D_c^t}\sum_{n=1}^{N_d}\mathbb{I}[w_{dn}=j] \text{ in NB} \end{cases}$$

Proof. The learning process at each minibatch in iDropout for LDA and NB reduces to maximizing the objective function of the following form:

$$F = -\frac{1}{2\sigma^2}||\beta_k - \beta_k^{prev}||_2^2 + \sum_{j=1}^{V} u_{kj}\log\left(\text{softmax}(\beta_k\odot\pi_k)_j\right)$$

where β_k^{prev} is made available from the previous minibatch (we omit superscript t for simplicity) and u_{kj} is defined as in the statement of the theorem.

Consider $x_1,\cdots,x_K$ as K-dimension one-hot vectors (x_k has only k^{th} element activated) and $\beta = [\beta_1\beta_2\cdots\beta_V]$ where β_j is j^{th} column of matrix β, then:

$$\text{softmax}(\beta_k)_j = \exp(s_{kj} - A(s_k))$$

with $s_{kj} = \beta_j^T x_k$ is a undropped score value and $A(s_k) = \log\sum_{i=1}^{V}\exp(s_{ki})$ is the log-partition function.

Assume π is drawn from the distribution ζ, corresponding the Inverted Dropout: $p(\pi_{ij} = 1/(1-dr)) = 1-dr, p(\pi_{ij} = 0) = dr$, then $\mathbb{E}_\zeta[\pi_{kj}] = 1$, and:

$$\text{softmax}(\beta_k\odot\pi_k)_j = \exp(\tilde{s}_{kj} - A(\tilde{s}_k))$$

with $\tilde{s}_{kj} = (\beta_i\odot\pi_i)^T x_k$, $A(\tilde{s}_k) = \log\sum_{i=1}^{V}\exp(\tilde{s}_{ki})$.

Using this notation, we can write F as:

$$F = -\frac{1}{2\sigma^2}||\beta_k - \beta_k^{prev}||_2^2 + \sum_{j=1}^{V} u_{kj}\mathbb{E}_\zeta[\tilde{s}_{kj} - A(\tilde{s}_k)]$$

Since $\mathbb{E}_\zeta[\pi_{kj}] = 1$ so the dropout technique preserves mean, leading to $\mathbb{E}_\zeta[\tilde{s}_{kj}] = s_{kj}$, then we have:

$$\begin{aligned}\mathbb{E}_\zeta[\tilde{s}_{kj} - A(\tilde{s}_k)] &= s_{kj} - A(s_k) - (\mathbb{E}_\zeta[A(\tilde{s}_k)] - A(s_k)) \\ &= \text{softmax}(\beta_k)_j - (\mathbb{E}_\zeta[A(\tilde{s}_k)] - A(s_k))\end{aligned}$$

Then we can write:

$$\begin{aligned}F = &-\frac{1}{2\sigma^2}||\beta_k - \beta_k^{prev}||_2^2 + \sum_{j=1}^{V} u_{kj}\log\left(\text{softmax}(\beta_k)_j\right) \\ &- (\mathbb{E}_\zeta[A(\tilde{s}_k)] - A(s_k))\sum_{j=1}^{V} u_{kj}\end{aligned}$$

Since the log-partition function $A(.)$ is convex, $(\mathbb{E}_\zeta[A(\tilde{s}_k)] - A(s_k))$ is always positive by Jensen's inequality and can therefore be interpreted as a regularizer. Indeed, applying second-order Taylor approximation to $A(\tilde{s}_k)$ around the undropped score vector s_k, we have means and covariances of the dropout features:

$$\begin{aligned}A(\tilde{s}_k) = A(s_k) &+ \nabla A(s_k)^T(\tilde{s}_k - s_k) \\ &+ \frac{1}{2}(\tilde{s}_k - s_k)^T\nabla^2 A(s_k)(\tilde{s}_k - s_k)\end{aligned}$$

then we obtain a following regularizer:

$$\begin{aligned}&\mathbb{E}_\zeta[A(\tilde{s}_k)] - A(s_k) = \frac{1}{2}\mathbb{E}_\zeta[(\tilde{s}_k - s_k)^T\nabla^2 A(s_k)(\tilde{s}_k - s_k)] \\ &= \frac{1}{2}\text{Tr}[\nabla^2 A(s_k)\text{Cov}_\zeta(\tilde{s}_k)] = \frac{1}{2}\sum_{j=1}^{V}\mu_{kj}(1-\mu_{kj})\text{Var}_\zeta[\tilde{s}_{kj}] \\ &= \frac{1}{2}\sum_{j=1}^{V}\mu_{kj}(1-\mu_{kj})\beta_j^T\text{Cov}_\zeta(x_k)\beta_j\end{aligned}$$

where $\mu_{kj} = \text{softmax}(s_k)_j$ is the model probability, the variance $\mu_{kj}(1-\mu_{kj})$ measures model uncertainty, and

$$\beta_j^T\text{Cov}_\zeta(x_k)\beta_j = \sum_{m=1}^{K}\frac{dr}{1-dr}x_{km}^2\beta_{mj}^2 = \frac{dr}{1-dr}\beta_{kj}^2$$

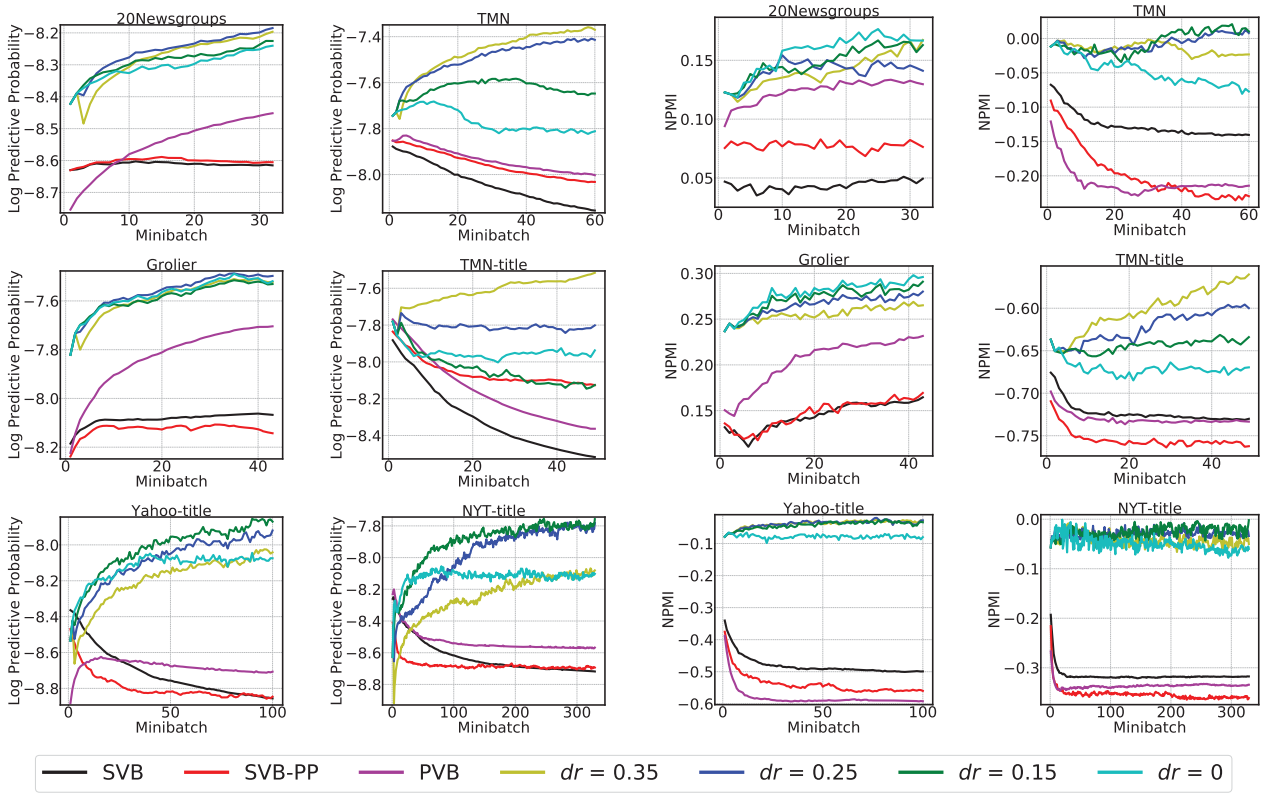


Fig. 2: Performance of 4 methods. iDropout uses drop rate $dr \in \{0.35, 0.25, 0.15, 0\}$ and $\sigma^2 = 100$. LDA is the base model. Higher is better.

Hence, $\mathbb{E}_\zeta[A(\tilde{s}_k)] - A(s_k) = \frac{dr}{2(1-dr)} \sum_{j=1}^{V} \mu_{kj}(1-\mu_{kj})\beta_{kj}^2$ has quadratic format w.r.t β_k. In other words, the effect of Dropout in iDropout is equivalent to a L2-regularization $R(\beta)$:

$$R(\beta) = (\mathbb{E}_\zeta[A(\tilde{s}_k)] - A(s_k)) \sum_{j=1}^{V} u_{kj}$$
$$= \frac{dr}{2(1-dr)} \sum_{j=1}^{V} \left[\mu_{kj}(1-\mu_{kj}) \sum_{j=1}^{V} u_{kj} \right] \beta_{kj}^2.$$

□

This is a theoretical interpretation on the ability of iDropout to reduce overfitting. Unlike other regularization techniques, each β_{kj} in iDropout has a different regularization parameter $\frac{dr}{2(1-dr)} \mu_{kj}(1-\mu_{kj}) \sum_{j=1}^{V} u_{kj}$, depending on the input data. This is interesting, since this data-dependent regularization allows each β_{kj} to have its own search space to catch the geometric property of data. With this property, dropout in our method is more effective than other standard computationally inexpensive regularizers, such as weight decay, filter norm constraints and sparse activity regularization [21].

V. Empirical Evaluation

In this section, we conduct various experiments to evaluate the performance of iDropout. Firstly, we simulate the streaming environment using 6 non-chronologically ordered datasets to thoroughly investigate the behavior of different methods on two aspects: how they balance between old and new information from data streams, as well as their ability to deal with noise and sparsity. Additionally, we examine how these methods adapt with sudden changes from the data stream in two settings: (1) using a dataset with time stamp; (2) simulating concept drift.

A. Baselines

We compare iDropout with three state-of-the-art frameworks: **SVB** [2], **SVB-PP** [4][2] and **PVB** [3]. We use grid search to select the best version of each framework for each dataset. The range of each parameter is as follows: the forgetting factor $\rho \in \{0.5, 0.6, 0.7, 0.8, 0.9, 0.99\}$ for SVB-PP, the population size $\alpha \in \{10^3, 10^4, 10^5, 10^6, 5.10^3, 5.10^4, 5.10^5, 5.10^6\}$ and dimming factor $\kappa \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$ for PVB, the variance $\sigma^2 \in \{0.01, 0.1, 1, 10, 100\}$ and the drop rate $dr \in \{0, 0.15, 0.25, 0.35\}$ for iDropout.

B. Experiments on datasets without time stamp

Base model and datasets: We use LDA to analyze 6 popular corporas including 2 regular text datasets (20News-

[2]SVB-HPP is not included since its application requires non-trivial efforts. Further, as observed by [4], SVB-HPP is often comparable to the best SVB-PP.

TABLE I: 6 datasets without time stamp

Dataset	Vocab size	Training size	Testing size	words/doc
20Newsgroups	24905	17846	1000	88.2
Grolier	15269	23044	1000	79.9
TMN	11599	31604	1000	24.3
TMN-title	2823	26251	1000	4.6
Yahoo-title	21439	517770	10000	4.6
NYT-title	46854	1664127	10000	5.0

Groups, Grolier[3]) and 4 short text ones (TagMyNews (TMN)[4], TagMyNews-title (TMN-title), Yahoo-title, NYT-title [5]) with some statistics in Table I.

Settings: Since all 6 datasets are not chronologically ordered, we simulate the streaming environment by dividing each dataset into a sequence of minibatches with batchsize: 500 for {Grolier, 20Newsgroups, TMN, TMN-title}, 5000 for {NYT-title, Yahoo-title}. We set prior of topic mixture $\alpha = 0.01$, the number of topic $K = 50$ for {Grolier, 20Newsgroups, TMN, TMN-title}, $K = 100$ for {NYT-title, Yahoo-title}.

Evaluation metric: Log predictive probability [14] (LPP) and Normalized pointwise mutual information (NPMI) [22] are used. While LPP measures the generalization of a model on unseen data, NPMI is used to examine the coherence and interpretability of the learned topics. The details of computing two metrics are given in the Appendix.

Result: The result is shown in Fig. 2. Overall, iDropout outperforms the baselines on all datasets, even when $dr = 0$. This observation essentially shows that iDropout has a more effective mechanism to balance information than other methods. More specifically, we figure out that methods have different behaviors on different types of datasets:

1) For two long text datasets 20NewsGroups and Grolier: $||\lambda^t||_1 = \sum_{i=1}^{T} \sum_{d \in D^t} N_d$ accumulates very fast over time, making SVB and SVB-PP become too stable with a very high rate, following Theorem 1. As a result, models have difficulties learning new information, explaining why the performance of these methods is roughly unchanged. PVB does not encounter this problem since it assumes the data stream is generated from a population distribution F_α, where the sample space is controlled by population size α, thus has a considerable evolution over time. It is also worth noting that training on these two datasets does not encounter seriously the problem of noise and sparsity, which explains why iDropout with different drop rate dr does not make a noticeable performance improvement.

2) In 4 short text datasets, there are two typical properties present in the data stream: noise and sparsity. While statistical noise potentially causes overfitting, sparsity leads to the lack of relevant information to make good predictions. Since the three baseline methods and iDropout with $dr = 0$ do not have an efficient way to tackle these serious problems, they all have poor performance over time. By contrast, iDropout with $dr \neq 0$ has a superior performance. This is achieved by two advantages of

[3]https://cs.nyu.edu/~roweis/data.html

[4]http://acube.di.unipi.it/tmn-dataset/

[5]http://archive.ics.uci.edu/ml/datasets/Bag+of+Words

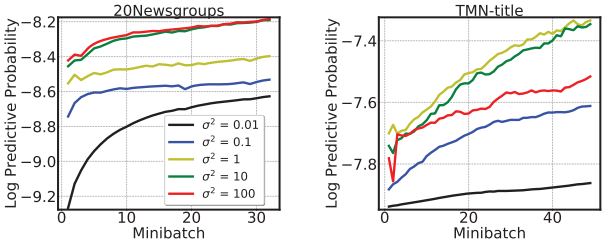


Fig. 3: Sensitivity of iDropout w.r.t variance σ^2.

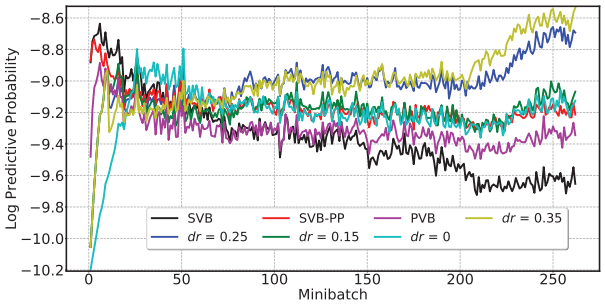


Fig. 4: Log predictive probability on The Irish Times dataset

Dropout. Firstly, it plays as a data-dependent regularizer, which makes iDropout more robust against overfitting. Moreover, Dropout in our method can be regarded as an ensemble of exponentially many learners, which is a well-known solution to make better predictions for sparse data.

Fig. 3 shows the sensitivity of iDropout w.r.t σ^2. 20Newsgroups (long text) and TMN-title (short text) are used for this evaluation, and the settings are the same as above. We can see that the variance affects the performance of iDropout significantly. Depending on the characteristics of data, there will be a trade-off between learning new information and keeping old information at early steps, corresponding to whether big or small values of σ^2 give better initial performance. In general, this hyperparameter needs to be tuned carefully in different datasets to obtain the best result. However, we suggest that $\sigma^2 = 100$ can be a good starting point, since this value gives fairly good performance on almost all datasets in our experiments.

C. Experiments on datasets with time stamp

Base model and datasets: We use the popular The Irish Times dataset[6], which is chronologically ordered to perform two different tasks: topic modeling using LDA and classification using Naive Bayes. In the LDA experiment, we simply throw away labels and use $K = 100$ and $\alpha = 0.01$. The Irish Times corpus contains 1376099 data instances from 02/01/1996 to 31/12/2017. There are 6 classes and vocab size is 25328.

Settings: Since the dataset is chronologically ordered, we divide the whole dataset into minibatches such that each minibatch D^t contains data of month t. We use documents

[6]https://www.kaggle.com/therohk/ireland-historical-news/

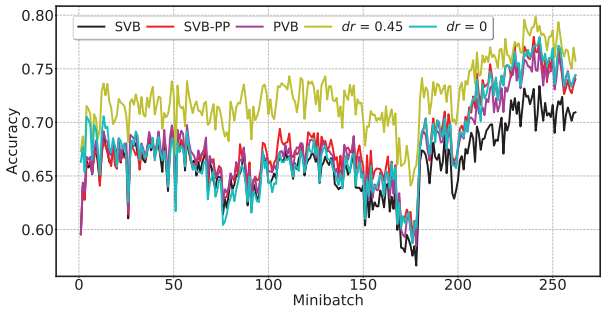


Fig. 5: Classification accuracy on The Irish Times

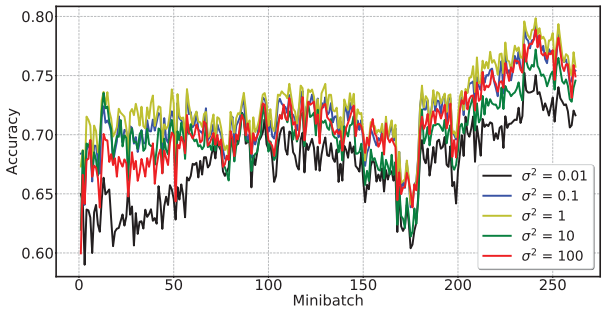


Fig. 6: Sensitivity of iDropout w.r.t variance σ^2 on the classification task

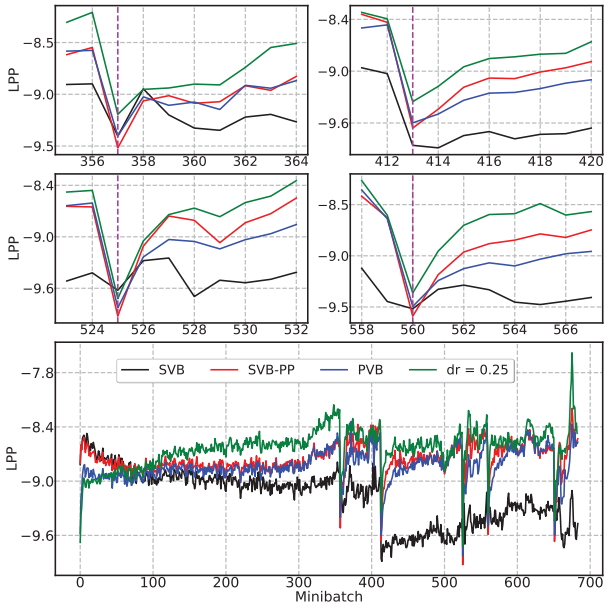


Fig. 7: Behavior on concept drift

of the next minibatch (month) to evaluate the model at any minibatch.

Evaluation metric: We use LPP to evaluate the learned topic model in LDA and accuracy to evaluate the classification performance.

Result on LDA: The result is shown in Fig. 4, in which iDropout uses $\sigma^2 = 100$. It is easy to find that our framework has a significantly better performance in comparison to other streaming Bayesian learning methods. With $dr = 0$, iDropout is still slightly better than the baselines, which again demonstrates the effectiveness of the balance mechanism of iDropout. We can also see that SVB suffers from the serious overfitting problem and has the severe decline in performance later. This is explained by Theorem 1, SVB becomes too stable after receiving a large enough amount of data, which makes model not able to learn new information, therefore fail to adapt to the change of data. The Irish Times is a short-text dataset, which contains undesirable properties, esepecially noise and sparsity. Same as in the previous experiment, the three baseline methods and iDropout with $dr = 0$ encounter the overfitting problem and have a decrease in performance over time. Then, thanks to the ability to prevent overfitting and the ensemble property, Dropout helps our method to obtain better generalization and thus effectively handles negative effects of noisy and sparse data. More specific, iDropout with $dr = 0.25$ and $dr = 0.35$ has a significant improvement over the baselines. This result strengthens our argument about the efficiency of Dropout in our method.

Result on NB: Fig. 5 shows the classification performance of four methods. While iDropout with $\sigma^2 = 1$, $dr = 0.25$ achieves the highest result on nearly the whole data stream, better about $6-8\%$ than SVB and about $3-4\%$ than SVB-PP and PVB, iDropout with $dr = 0$ only has a similar performance compared to SVB-PP and PVB. This continues to strengthen our argument that Dropout plays an important role in our method. Furthermore, there is a period of time when the performance of all methods drops (about 175th minibatch) due to sudden changes in the data stream. Thanks to the balance ability and the effect of Dropout, iDropout (even with $dr = 0$) does not fall too deep and can recover quickly to keep leading on the remaining minibatches.

Fig. 6 shows the impact of variance σ^2 on the performance of iDropout. More specific, we show the classification results of iDropout on The Irish Times dataset with different values of σ^2. Looking at Figure 6, we can see that $\sigma^2 = 1$ gives the best performance, better about 5% compared to $\sigma^2 = 0.01$. This a significant difference, suggesting that we need to tune this hyperparameter carefully.

D. Evaluation on concept drift

Concept drift [23] is the phenomenon when the underlying distribution of data changes suddenly. We conduct this experiment to examine our argument about the stability and plasticity dilemma in IV-A, especially the ability to adapt abrupt changes in the data stream.

Setting up: We simulate concept drift by using The Irish Times dataset as follow: data is divided into minibatches, each minibatch contains 2000 documents of a particular class and all minibatches of the same class are placed adjacent to each other. Therefore, the concept drift happens significantly when data transfers from one class to another. We then use LDA with $K = 100$ and $\alpha = 0.01$ to analyze these documents without

information from labels. After learning on each minibatch, the model is evaluated by computing LPP on the next minibatch.

Result: The result is illustrated in Fig. 7, in which top four figures zoom in the first four drift points, i.e., where the data stream transfers from a class to another. SVB performs poorly when facing concept drift. The performance of SVB plunges after each drift point and recovers slowly due to its too much stability discussed in Theorem 1. SVB-PP can delay this problem by exponentially forgetting the information of old data, which allows it to adapt better new information from new data. PVB can also adapt to concept drift, since the variance of the variational posterior never decreases below a given threshold indirectly controlled by population size α. Finally, iDropout provides the best result (we continue to use variance $\sigma^2 = 100$). The ability to reduce overfitting and the ensemble property of Dropout allows iDropout to obtain better generalization, thus prevent the performance from falling too deeply when facing concept drift. Moreover, the balance mechanism enables iDropout to easily learn new underlying distribution of data, which incorporates with the ensemble learning to help our method adapt quickly to these new changes in data.

VI. Conclusion

We presented iDropout, a novel and straightforward framework to address many challenges of learning in streaming conditions. In particular, iDropout helps Bayesian models to tackle the stability-plasticity dilemma and handle noisy and sparse data. Further, iDropout is able to adapt quickly to abnormal changes in data streams. iDropout can be used for a wide range of models.

Acknowledgements

This research is supported by Vingroup Innovation Foundation (VINIF) in project code VINIF.2019.DA18. The first author is supported by Domestic Master Scholarship Programme code VINIF.2019.ThS.23.

Appendix

Evaluation metrics for the unsupervised task

Log Predictive Probability [14]: Predictive Probability measures the predictiveness and generalization of a model on new data. Assume that after learning from training data D_{train}, we obtain the model parameter β. For each document in testing D_{test} with more than or equal to 5 words, we divide randomly into two disjoint parts $\mathbf{w_{obs}}$ and $\mathbf{w_{ho}}$ with a ratio of 80:20. We next do inference for $\mathbf{w_{obs}}$ to estimate θ^{obs}. Then, we approximate the predictive probability $\mathbf{w_{ho}}$ as:

$$\begin{aligned} p(\mathbf{w_{ho}} \mid \mathbf{w_{obs}}, \beta) &= \prod_{w \in \mathbf{w_{ho}}} p(w \mid \mathbf{w_{obs}}, \beta) \\ &\approx \prod_{w \in \mathbf{w_{ho}}} p(w \mid \theta^{obs}, \beta) \\ &= \prod_{w \in \mathbf{w_{ho}}} \sum_{k=1}^{K} p(w \mid z = k, \beta) p(z = k \mid \theta^{obs}) \\ &= \prod_{w \in \mathbf{w_{ho}}} \sum_{k=1}^{K} \theta_k^{obs} \beta_{kw} \end{aligned}$$

Then Log Predictive Probability of each document d is:

$$LPP_d = \frac{\log p(\mathbf{w_{ho}} \mid \mathbf{w_{obs}}, \beta)}{|\mathbf{w_{ho}}|} \tag{7}$$

(with $|\mathbf{w_{ho}}|$ is the length of d in $\mathbf{w_{ho}}$) and on the whole testing D_{test} is:

$$\text{Log Predictive Probability} = \frac{\sum_{d \in D_{test}} LPP_d}{|D_{test}|} \tag{8}$$

Log Predictive Probability was averaged from 5 random splits, each was on 1000 documents.

Normalized Pointwise Mutual Information [22]: NPMI is the measure to help us see the coherence or semantic quality of individual topics. For each topic k, we pick a set $\mathbf{w^k} = \{w_1^k, w_2^k, ..., w_t^k\}$, including t words with the highest probabilities in topic distribution β_k. NPMI of one topic k is computed as follows:

$$\begin{aligned} \text{NPMI}(k, \mathbf{w^k}) &= \frac{2}{t(t-1)} \sum_{i=2}^{t} \sum_{j=1}^{i-1} \frac{\log \frac{p(w_i^k, w_j^k)}{p(w_i^k) p(w_j^k)}}{-\log p(w_i^k, w_j^k)} \\ &\approx \frac{2}{t(t-1)} \sum_{i=2}^{t} \sum_{j=1}^{i-1} \frac{\log \frac{D(w_i^k, w_j^k) + 10^{-2}}{D} - \log \frac{D(w_i^k) D(w_j^k)}{D^2}}{-\log \frac{D(w_i^k, w_j^k) + 10^{-2}}{D}} \\ &= \frac{2}{t(t-1)} \sum_{i=2}^{t} \sum_{j=1}^{i-1} -1 + \frac{2 \log D - \log D(w_i^k) - \log D(w_j^k)}{\log D - \log(D(w_i^k, w_j^k) + 10^{-2})} \end{aligned}$$

where D is the total number of documents, $D(w_i^k)$ is the number of docs containing w_i^k, $D(w_i^k, w_j^k)$ is the number of docs containing pair (w_i^k, w_j^k).

Overall, NPMI of a model with all K topics is:

$$NPMI = \frac{1}{K} \sum_{k=1}^{K} NPMI(k, t) \tag{9}$$

In the experiments, we choose $t = 20$ for each topic.

References

[1] M. Mermillod, A. Bugaiska, and P. Bonin, "The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects," *Frontiers in psychology*, vol. 4, p. 504, 2013.

[2] T. Broderick, N. Boyd, A. Wibisono, A. C. Wilson, and M. I. Jordan, "Streaming variational bayes," in *Advances in Neural Information Processing Systems*, pp. 1727–1735, 2013.

[3] J. McInerney, R. Ranganath, and D. Blei, "The population posterior and bayesian modeling on streams," in *Advances in Neural Information Processing Systems*, pp. 1153–1161, 2015.

[4] A. Masegosa, T. D. Nielsen, H. Langseth, D. Ramos-López, A. Salmerón, and A. L. Madsen, "Bayesian models of data streams with hierarchical power priors," in *International Conference on Machine Learning*, pp. 2334–2343, 2017.

[5] S. Rifai, X. Glorot, Y. Bengio, and P. Vincent, "Adding noise to the input of a model trained with a regularized objective," *arXiv preprint arXiv:1104.3250*, 2011.

[6] S. Wager, S. Wang, and P. S. Liang, "Dropout training as adaptive regularization," in *Advances in Neural Information Processing Systems*, pp. 351–359, 2013.

[7] S. Wang, M. Wang, S. Wager, P. Liang, and C. D. Manning, "Feature noising for log-linear structured prediction," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1170–1179, 2013.

[8] P. Baldi and P. Sadowski, "The dropout learning algorithm," *Artificial intelligence*, vol. 210, pp. 78–122, 2014.
[9] D. P. Helmbold and P. M. Long, "On the inductive bias of dropout," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 3403–3454, 2015.
[10] P. Mianjy, R. Arora, and R. Vidal, "On the implicit bias of dropout," in *International Conference on Machine Learning*, pp. 3537–3545, 2018.
[11] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.
[12] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.
[13] Z. Ghahramani and H. Attias, "Online variational bayesian learning," in *Slides from talk presented at NIPS workshop on Online Learning*, 2000.
[14] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.
[15] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.
[16] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
[17] N. Chen, J. Zhu, J. Chen, and B. Zhang, "Dropout training for support vector machines.," in *AAAI*, pp. 1752–1759, 2014.
[18] S. Zhai and Z. Zhang, "Dropout training of matrix factorization and autoencoder for link prediction in sparse graphs," in *Proceedings of the 2015 SIAM International Conference on Data Mining*, pp. 451–459, SIAM, 2015.
[19] C. Ha, V.-D. Tran, L. N. Van, and K. Than, "Eliminating overfitting of probabilistic topic models on short and noisy text: The role of dropout," *International Journal of Approximate Reasoning*, vol. 112, pp. 85 – 104, 2019.
[20] D. P. Helmbold and P. M. Long, "Surprising properties of dropout in deep networks," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 7284–7311, 2017.
[21] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
[22] G. Bouma, "Normalized (pointwise) mutual information in collocation extraction," *Proceedings of GSCL*, pp. 31–40, 2009.
[23] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM computing surveys (CSUR)*, vol. 46, no. 4, p. 44, 2014.
[24] L. Maaten, M. Chen, S. Tyree, and K. Weinberger, "Learning with marginalized corrupted features," in *International Conference on Machine Learning*, pp. 410–418, 2013.
[25] H. Noh, T. You, J. Mun, and B. Han, "Regularizing deep neural networks by noise: Its interpretation and optimization," in *Advances in Neural Information Processing Systems*, pp. 5109–5118, 2017.
[26] P. Baldi and P. J. Sadowski, "Understanding dropout," in *Advances in neural information processing systems*, pp. 2814–2822, 2013.

Noname manuscript No.
(will be inserted by the editor)

Adaptive Infinite Dropout for Noisy and Sparse Data Streams*

Ha Nguyen · Hoang Pham · Son Nguyen · Ngo Van Linh · Khoat Than

Received: / Accepted:

Abstract The ability to analyze data streams, which arrive sequentially and possibly infinitely, is increasingly vital in various online applications. However, data streams pose various challenges, including *sparse and noisy data* as well as *concept drifts*, which easily mislead a learning method. This paper proposes a simple yet robust framework, called *Adaptive Infinite Dropout* (aiDropout), to effectively tackle these problems. Our framework uses a dropout technique in a recursive Bayesian approach in order to create a flexible mechanism for balancing between old and new information. In detail, the recursive Bayesian approach imposes a constraint on the model parameters to make a regularization term between the current and previous mini-batches. Then, dropout whose drop rate is autonomously learned can adjust the constraint to new data. Thanks to the ability to reduce overfitting and the ensemble property of Dropout, our framework obtains better generalization, thus it effectively handles undesirable effects of noise and sparsity. In particular, theoretical analyses show that aiDropout imposes a data-dependent regularization, therefore, it can adapt quickly to sudden changes from data streams. Extensive experiments show that aiDropout significantly outperforms the state-of-the-art baselines on a variety of tasks such as supervised and unsupervised learning.

Keywords Bayesian models · Data streams · Streaming learning · Dropout · Data-dependent Regularization

* A part of this work appears in [46]

Ha Nguyen [1,a] E-mail: hant.hanguyen@gmail.com ·
Hoang Pham[2,a] E-mail: pvh16021998@gmail.com ·
Son Nguyen[3,a] E-mail: sonnguyenkstn@gmail.com ·
Ngo Van Linh[4] E-mail: linhnv@soict.hust.edu.vn ·
Khoat Than [5] E-mail: khoattq@soict.hust.edu.vn ·
[1,2,3,4,5] School of Information & Communication Technology, Hanoi University of Science and Technology, No. 1, Dai Co Viet road, Hanoi, Vietnam
[a] Equal contribution

1 Introduction

Bayesian modelling has become a powerful tool in machine learning and has been utilized in a wide range of applications such as text mining [4,56], recommendation systems [33,57], social network [41,18], computer vision [12], bioinformatics [48], etc. Based on assumptions about data, we can straightforwardly build a model with hidden variables and observations. The inferred posteriors of the hidden variables expose data characteristics that can used in applications.

There are numerous inference methods [35,63] to work well in a static environment in which there is no change in data in the entire training process. However, in modern applications such as social networks, and E-commerce, data is generated continually and is collected in infinitely many mini-batches (known as the streaming environment). The prevailing characteristics of data are big, noisy and sparse. Moreover, the various kinds of concept drifts (such as sudden, incremental, and recurring concept drifts [14,31]), in which the statistical characteristics of new data change, can happen. Therefore, developing an effective learning method poses challenging problems in the streaming environment. Firstly, traditional inference methods which implement an iterative procedure on all data are impossible to work on data streams. It is necessary for a method to adapt to new data quickly without revisiting the past data. As a result, it must deal with the stability-plasticity dilemma [42]. The dilemma requires a learning method to be stable to effectively exploit acquired knowledge when working on new data whose characteristics are similar to those of the past data. Simultaneously, it should be plastic when concept drift happens. Secondly, noisy and sparse data [54,2,21,37] makes a lot of difficulties for learning methods. While sparse data does not provide an unclear context, noisy data can mislead the methods. Consequently, the generalization ability of a learned model can be limited. In this paper, we focus on these challenges.

Some recent studies [6,40,38,10,53,55] have provided solutions to learning from data streams. Those methods enable Bayesian models, which are designed for static conditions, to work with data streams. The recursive Bayesian approach [6,40,38, 10,45,1] has emerged as an effective solution and has been paid a great deal of attention by researchers. The main idea is that the learned posterior from a mini-batch is used as the prior in the next one. Therefore, this approach provides a flexible mechanism to exploit acquired knowledge in the current mini-batch without revisiting past data. However, the existing studies are still limited when facing the above challenges. We found that *streaming variational Bayes* (SVB) [6] could suffer from the phenomenon of overconfident posterior after receiving a large enough amount of data. Concretely, the posterior variance would become arbitrarily small leading to point-mass posterior concentration. This arguably cause several critical issues in the online Bayesian updates including poor uncertainty representation of the underlying data-generating distribution, and lack of the flexibility to adapt to the sudden changes in data streams. Hierarchical power prior (HPP) [38,39] is more plastic to learn a new concept, however, it does not have any efficient way to work on noisy and sparse data. Other recursive Bayesian-based studies [2,10, 53,55] require external knowledge to deal with the challenges.

In this paper, we propose a novel framework called *Adaptive Infinite Dropout* (aiDropout) which enables a wide range of models to work in streaming environments. Our framework is based on the recursive Bayesian approach and dropout technique [51] to create an effective solution to learning from data streams. It has

several benefits. Firstly, aiDropout has an easy mechanism to balance the information between old and new data throughout the data stream, which helps tackle the stability-plasticity dilemma. Secondly, we theoretically prove that Dropout in aiDropout induces a data-dependent regularization, which allows each parameter component to have its own search space to capture geometric properties, especially highly discriminative characteristics, of the data features. This is extremely important when data comes continuously with high uncertainty, which has the possibility of concept drifts or undesirable properties such as noise and sparsity. Thirdly, Dropout in our method works as an ensemble of an exponential number of learners, which is very useful in making good predictions for future data. These advantages help our method obtain better generalization. Finally, because the data inevitably changes over time, the drop rate should be adapted according to the data. Our method provides a mechanism to automatically tune the drop rate and therefore obtains better generalization and more practicality.

We empirically evaluate the performance of aiDropout compared to the existing state-of-the-art streaming methods by using two base models: 1) *latent Dirichlet allocation* (LDA) [4] for topic modelling and 2) *Naïve Bayes* (NB) [49] for classification. The extensively experimental results on both learning tasks show the superior effectiveness of aiDropout.

ROADMAP: Section 2 briefly provides closely related work. We formally describe the aiDropout framework in Section 3 and its applications in Section 4. Non-trivial findings of aiDropout are described in Section 5. Section 6 presents extensive experiments and a conclusion is made in Section 7.

2 Related work

Several studies have addressed the inference problem on data streams. They are divided into two notable approaches: Stochastic optimization and recursive Bayesian update. The first one [24, 40, 52, 25, 27] considers the inference problem as a stochastic optimization problem in which the objective function is the expectation of the likelihood. In particular, stochastic variational inference (SVI) [24] aims to optimize the empirical expectation by sampling data instances from the uniform distribution on a fixed dataset, which is impractical in the streaming environment. To address this limit, population variational Bayes (PVB) [40] assumes that data streams are generated by consecutively sampling ν (population size) data instances from the population distribution F_ν instead of the uniform distribution. However, ν must be manually adjusted to achieve better performances. Meanwhile, the second approach assumes that the learned knowledge from a mini-batch is considered as prior knowledge in the next one [6, 38, 32, 8, 45, 30, 1, 61]. Streaming variational Bayes (SVB) [6] uses the variational distribution learned in the previous mini-batch as the prior distribution for the current one. However, we find that SVB can become too stable in many cases and therefore can be unable to learn new information once trained from large enough data. To address the drawback of SVB, hierarchical power prior (HPP) [38, 39] uses a forgetting factor relating to the degree of forgetting the old knowledge at the current mini-batch. Unfortunately, because this forgetting factor is considered as a hidden variable, the model in HPP

is non-conjugate, leading to difficulties in inferring complicated Bayesian models. In addition, a lot of studies [45,30,1,61] apply the recursive Bayesian approach to deal with multiple tasks. In this paper, we only concentrate on addressing the problem in data streams without changing tasks.

In terms of dealing with sparse and noisy data, none of these mentioned methods pays careful attention to this problem. Following the recursive Bayesian approach, some recent studies [10,2,53,55] have proposed methods based on exploiting external/prior knowledge which is derived from a pre-trained model or human knowledge. Albeit those methods show promising results in coping with extremely short texts, their performances depend heavily on the quality of prior knowledge. Moreover, it is difficult to find external knowledge which is suitable for data streams. In contrast, by adopting the dropout technique in the streaming environment, our proposed method can tackle the problem of noise and sparsity efficiently without using external knowledge.

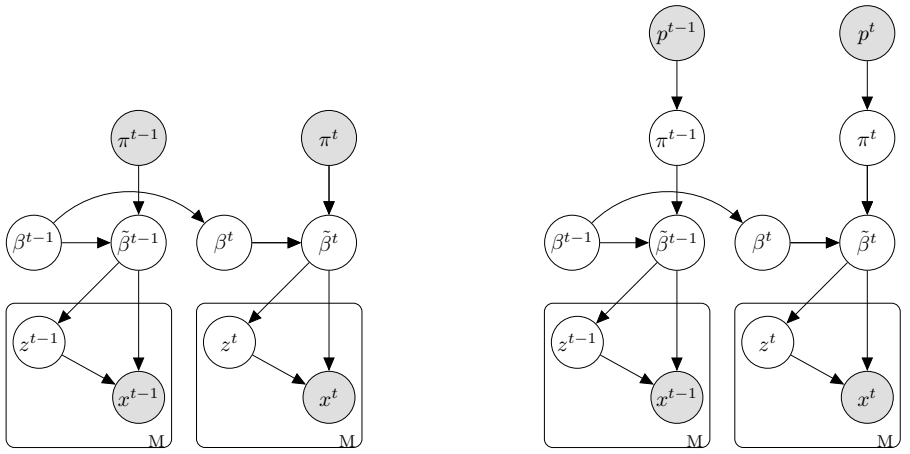
Dropout [23] is well-known as a powerful regularization technique [44] for preventing overfitting by discouraging the co-adaptation of features. Moreover, dropout provides an efficient way to approximately combine an exponential number of learners, working as a form of ensemble learning. The idea of dropout is to randomly drop a subset of features at each iteration during training. At first, the drop rate is manually tuned, which demands to use grid-search, a prohibitive operation with large network models. To address this problem, some methods [28, 34,13] have been proposed to automatically determine the drop rate. Dropout has achieved great success in various machine learning models, e.g., neural networks [51], SVM [7], matrix factorization [62], topic models [21]. However, the applications of Dropout are still limited to a static data environment. In parallel to our work, Guzy et.al [20] showed that dropout helps deal with recurring concept drift because dropout leads to using submodels generated for each concept. However, this work only focused on deterministic neural networks and lacked an adaptive mechanism. Moreover, the random selection of features in adaptive Random Forest [15] achieves good performance in streaming environments. Therefore, exploiting adaptive dropout for learning Bayesian models on data streams is hopeful. In this paper, we explore Dropout for training Bayesian models to address the challenges in the streaming data. Our analysis about the role of Dropout as regularization applies well to a large class of Bayesian models, extending existing works [47,58, 59,3,22,43] to significantly wider contexts.

3 Adaptive Infinite Dropout for Bayesian models

In this section, we first introduce how to apply dropout to a general Bayesian model to work on data streams. Then, we present a strategy to learn drop rate.

3.1 Infinite Dropout

We consider a general model $B(\beta, z, x)$ [40,24] which consists of global variable β, the set of observations $x = x_{1:M}$ and the set of hidden variables $z = z_{1:M}$. More explicitly, each data instance (observation) x_i has a hidden variable z_i to encode



(a) iDropout for $B(\beta, z, x)$ (b) aiDropout for $B(\beta, z, x)$

Fig. 1: Graphical representation for iDropout and aiDropout.

the hidden feature of x_i. Meanwhile, the global variable β is shared among all of the data instance $x_{1:M}$. Bayesian methods aim to infer the posterior distribution of hidden variables $p(\beta, z|x)$ when given a fixed dataset. Undoubtedly, this can not work with data streams where the data comes in an infinite sequence of mini-batches $C = \{D^1, D^2, \cdots, D^t, \cdots\}$ and each mini-batch t consists of M observed data points: $D^t = \{x_1^t, x_2^t, \cdots, x_M^t\}$.

We need to extend the model to also describe the dynamics of data streams. Here we assume that only the global variable β evolves over time, which we indicate with superscript t, i.e., β^t. We introduce a transition model $p(\beta^t|\beta^{t-1})$ to describe the transformation between two consecutive mini-batches:

$$p(\beta_k^t|\beta_k^{t-1}) = \mathcal{N}(.|\beta_k^{t-1}, \sigma^2 I) \tag{1}$$

where k is the row index of β^{t-1} and I is the identity matrix of size V. The variance σ^2 is a hyperparameter, which describes our assumption about the fluctuation of β_k between two consecutive mini-batches.

We assume that β is represented by a $K \times V$ matrix. Dropout is utilized in our framework as follows. In each mini-batch t, we drop randomly some elements of matrix β^t. This is implemented by using a hyperparameter called drop matrix π^t to make the *element-wise product* with β^t, then going through a transformation: $\tilde{\beta}^t = f(\beta^t \odot \pi^t)$. Transformation f should be chosen to assure that $\tilde{\beta}^t$ can replace β in model $B(\beta, z, x)$ at each mini-batch t (in the later subsections, we use *softmax* to be the transformation). Given the new global variable $\tilde{\beta}^t$ at each mini-batch t, the generative process of all data points is similar to the original model B (Fig. 1a). In order to keep the randomness of Dropout, we use a different drop matrix at each mini-batch. Each element π_{ij}^t of π^t is generated using one of two options:

1. Bernouli dropout: $p(\pi_{ij}^t = 1) = 1 - p$, $p(\pi_{ij}^t = 0) = p$

2. Inverted dropout:

$$p(\pi_{ij}^t = 1/(1-p)) = 1-p, p(\pi_{ij}^t = 0) = p \tag{2}$$

in which p is drop rate. Note that when β^t is used at test time, it has to be rescaled by $\mathbb{E}[\pi_{ij}^t]$. By doing this scaling, $2^{K\times V}$ models with shared parameters can be combined into a single model to be used at test time, which works as a form of ensemble learning.

Learning: At each mini-batch t, we make a point estimate for β^t by maximizing $\log p(\beta^t|\beta^{t-1}, \pi^t, D^t)$, where β^{t-1} is learned from the previous mini-batch:

$$\begin{aligned}\hat{\beta}^t &= \arg\max_{\beta^t} \Big\{\log p(\beta^t|\beta^{t-1}, \pi^t, D^t)\Big\} \\ &= \arg\max_{\beta^t} \Big\{\log p(\beta^t, D^t|\beta^{t-1}, \pi^t)\Big\} \\ &= \arg\max_{\beta^t} \Big\{\log p(\beta^t|\beta^{t-1}) + \log p(D^t|\pi^t, \beta^t)\Big\}\end{aligned} \tag{3}$$

While the direct optimization of $p(D^t|\pi^t, \beta^t)$ is intractable, it is significantly easier to optimize than the complete data likelihood $\int_{z^t} p(D^t, z^t|\pi^t, \beta^t)$. By introducing a variational distribution $q(z^t|\phi^t)$ defined over the local variables z^t, we then have:

$$\begin{aligned}\log p(D^t|\pi^t, \beta^t) &= \log \int_{z^t} p(D^t, z^t|\pi^t, \beta^t) dz^t = \log \int_{z^t} q(z^t|\phi^t) \frac{p(D^t, z^t|\pi^t, \beta^t)}{q(z^t|\phi^t)} dz^t \\ &\geq \int_{z^t} q(z^t|\phi^t) \log \frac{p(D^t, z^t|\pi^t, \beta^t)}{q(z^t|\phi^t)} dz^t = E_{q(z^t|\phi^t)}\left[\log \frac{p(D^t, z^t|\pi^t, \beta^t)}{q(z^t|\phi^t)}\right]\end{aligned} \tag{4}$$

By substituting (4) into (3), our objective function can be rewritten as:

$$\{\hat{\beta}^t, \hat{\phi}^t\} = \arg\max_{\beta^t, \phi^t} \left\{\log p(\beta^t|\beta^{t-1}) + E_{q(z^t|\phi^t)}\left[\log \frac{p(D^t, z^t|\pi^t, \beta^t)}{q(z^t|\phi^t)}\right]\right\} \tag{5}$$

The learning process is composed of two phases. We first infer the local variables by inheriting from the original model B, and then update the global variable. Algorithm 1 briefly describes the learning process.

Algorithm 1 Learning in iDropout

Input: Data sequence $\{D^1, D^2, ...\}$, variance σ^2, drop rate p
Output: Global variable β

Initialize β^0 randomly
for t^{th} *mini-batch with data* D^t **do**
  repeat
    Draw dropout matrix π^t randomly
    Do inference with respect to ϕ^t in (5) given β^t and D^t
    Estimate β^t in (5) by using a gradient-based algorithm given the statistics from ϕ^t, D^t
  until *convergence*;

3.2 Learning drop rate

Difference from the previous subsection in which π^t is sampled from a fixed Bernoulli distribution, we will infer the posterior of π^t. The prior distribution for π^t is a Bernoulli distribution parameterized by p^t (Fig. 1b). Our goal here is to maximize $\log p(\beta^t|\beta^{t-1}, p^t, D^t)$ at each mini-batch t:

$$\begin{aligned}\hat{\beta}^t &= \arg\max_{\beta^t} \Big\{\log p(\beta^t|\beta^{t-1}, p^t, D^t)\Big\} \\ &= \arg\max_{\beta^t} \Big\{\log p(\beta^t, D^t|\beta^{t-1}, p^t)\Big\} \\ &= \arg\max_{\beta^t} \Big\{\log p(\beta^t|\beta^{t-1}) + \log p(D^t|p^t, \beta^t)\Big\} \end{aligned} \tag{6}$$

To automatically determine the posterior of π^t and hyperparameter p^t, we use the empirical Bayesian method and introduce the variational distribution $q(\pi^t|\lambda^t) = Bernoulli(\lambda^t)$ where λ^t is the variational hyper-parameter of Bernoulli distribution. Therefore, we have a lower bound on $\log p(D^t|p^t, \beta^t)$:

$$\begin{aligned}&\log p(D^t|p^t, \beta^t) = \log \sum_{\pi^t} p(D^t, \pi^t|p^t, \beta^t) = \log \sum_{\pi^t} q(\pi^t|\lambda^t) \frac{p(D^t, \pi^t|p^t, \beta^t)}{q(\pi^t|\lambda^t)} \\ &\geq \sum_{\pi^t} q(\pi^t|\lambda^t) \log \frac{p(D^t, \pi^t|p^t, \beta^t)}{q(\pi^t|\lambda^t)} = \sum_{\pi^t} q(\pi^t|\lambda^t) \log \frac{p(D^t|\pi^t, \beta^t)p(\pi^t|p^t)}{q(\pi^t|\lambda^t)} \\ &= E_{q(\pi^t|\lambda^t)}\left[\log p(D^t|\pi^t, \beta^t)\right] - KL\left[q(\pi^t|\lambda^t)||p(\pi^t|p^t)\right] \end{aligned} \tag{7}$$

By introducing (7) into (6), the objective function can be written as:

$$\begin{aligned}\{\hat{\beta}^t, \hat{\lambda}^t, \hat{p}^t\} = \arg\max_{\beta^t, \lambda^t, p^t} \Big\{&\log p(\beta^t|\beta^{t-1}) + E_{q(\pi^t|\lambda^t)}\left[\log p(D^t|\pi^t, \beta^t)\right] \\ &-KL\left[q(\pi^t|\lambda^t)||p(\pi^t|p^t)\right]\Big\} \end{aligned} \tag{8}$$

While the KL term has a closed-form expression, it is not straightforward to estimate the expected log-likelihood $\mathbb{E}_{q(\pi^t)} \log p(D^t|\beta^t, \pi^t)$, and more importantly the derivative of this second term with respect to the variational distribution parameter λ^t due to the difficulty in applying reparameterization trick [29] to discrete random variables. There are some studies [26, 36, 19, 60] to handle the learning problem from discrete latent variables. We select a simple solution that exploits the Gumbel-Softmax distribution [26], a continuous distribution, which helps us to do reparameterization for discrete variables. The original Gumbel-Softmax trick is intended to approximate samples from a categorical distribution that depends on a temperature parameter τ, but here we concentrate on the Bernoulli distribution case. It turns out that we now have a simple formula for the continuous relaxation $\tilde{\pi}^t$ of π^t:

$$\begin{aligned}\tilde{\pi}^t = \frac{\exp\left(\frac{\log(\lambda^t)+g_1}{\tau}\right)}{\exp\left(\frac{\log(\lambda^t)+g_1}{\tau}\right) + \exp\left(\frac{\log(1-\lambda^t)+g_2}{\tau}\right)} \\ \text{with } g_1, g_2 \sim Gumbel(0,1)^1 \end{aligned} \tag{9}$$

In each iteration, we draw L samples $(\{\tilde{\pi}_l^t\}_{l=1}^L)$ of π^t to calculate $E_{q(\pi^t|\lambda^t)}\left[\log p(D^t|\pi^t,\beta^t)\right]$ as follows:

$$E_{q(\pi^t|\lambda^t)}\left[\log p(D^t|\pi^t,\beta^t)\right] = \frac{1}{L}\sum_{l=1}^{L}\log p(D^t|\tilde{\pi}_l^t,\beta^t)$$

The previous studies [29, 26] showed that using the reparameterization trick with $L = 1$ also achieves good performance in terms of both computation and quality. The objective function (8) can then be rewritten in the form of:

$$\{\hat{\beta}^t,\hat{\lambda}^t,\hat{p}^t\} = \underset{\beta^t,\lambda^t,p^t}{\arg\max}\left\{\log p(\beta^t|\beta^{t-1}) + \log p(D^t|\tilde{\pi}^t,\beta^t) - KL\left[q(\pi^t|\lambda^t)||p(\pi^t|p^t)\right]\right\} \tag{10}$$

Similar to the previous subsection, instead of directly optimizing $\log p(D^t|\tilde{\pi}^t,\beta^t)$, we try to optimize the complete-data log likelihood $\int_{z^t} p(D^t,z^t|\tilde{\pi}^t,\beta^t)$. We then rewrite (10) in the form of:

$$\begin{aligned}\{\hat{\beta}^t,\hat{\phi}^t,\hat{\lambda}^t,\hat{p}^t\} = \underset{\beta^t,\phi^t,\lambda^t,p^t}{\arg\max}\Bigg\{&\log p(\beta^t|\beta^{t-1}) + E_{q(z^t|\phi^t)}\left[\log\frac{p(D^t,z^t|\tilde{\pi}^t,\beta^t)}{q(z^t|\phi^t)}\right]\\ &-KL\left[q(\pi^t|\lambda^t)||p(\pi^t|p^t)\right]\Bigg\}\end{aligned} \tag{11}$$

It is worth observing that the parts of objective functions w.r.t β and ϕ in iDropout (5) and aiDropout (11) are in the same form. They are merely different in the random dropout variable π. While in iDropout π is sampled from a Bernoulli distribution with a fixed drop rate, aiDropout provides a mechanism to autonomously learn drop rate for adapting to new data. In the special case that samples of π in both iDropout and aiDropout are the same, aiDropout will degenerate to iDropout. Algorithm 2 briefly describes the learning process of aiDropout.

Algorithm 2 Learning in aiDropout

Input: Data sequence $\{D^1, D^2, ...\}$, variance σ^2, τ
Output: Global variable β, λ, p

Initialize β^0 randomly
for t^{th} *mini-batch with data* D^t **do**
  Set an initial estimate for λ^t
  repeat
    Draw dropout matrix $\tilde{\pi}^t$ randomly
    Do inference with respect to ϕ^t in (11) given β^t and D^t
    Find (β^t,λ^t,p^t) in (11) by using a gradient-based algorithm given the statistics from ϕ^t, D^t
  until *convergence*;

[1] We can sample realizations from the $Gumbel(0,1)$ distribution by firstly drawing $u \sim Uniform(0,1)$, and then computing $g = -\log(-\log(u))$.

4 Case study

We will show the application of aiDropout to latent Dirichlet allocation (LDA) [4] for document analysis and Multinomial Naïve Bayes (NB) [49] for document classification (see Appendix A for iDropout).

4.1 Case study 1: when LDA is the base model

LDA is one of the most popular unsupervised models and provides an effective and interpretable solution in a wide range of applications such as text mining, recommendation system, etc. Therefore, some recent studies considered LDA as a base model to develop learning methods in the streaming environment. We merely explore how aiDropout works on LDA.

Suppose that each mini-batch t consists of M documents and each document d contains N_d words. LDA aims to learn hidden topics in a text dataset as well as the topic proportion of each document. Hyper-parameter α is the parameter of Dirichlet distribution for topic mixture θ, the matrix $\tilde{\beta}$ of size $K \times V$ is the topic distribution over V words in the vocabulary.

The generative process for documents in each mini-batch t^{th} is as follows:

1. Draw $\beta^t : \beta_k^t \sim \mathcal{N}(\beta_k^{t-1}, \sigma^2 I)$
2. Draw $\pi^t : \pi_{kj}^t \sim \text{Bernoulli}(p_{kj}^t)$
3. Calculate topic distribution $\tilde{\beta}^t$:

$$\tilde{\beta}_{kj}^t = \text{softmax}(\beta_k^t \odot \pi_k^t)_j = \frac{\exp(\beta_{kj}^t \pi_{kj}^t)}{\sum_{i=1}^{V} \exp(\beta_{ki}^t \pi_{ki}^t)}$$

4. For each document d in mini-batch t:
   (a) Draw topic mixture: $\theta_d \sim \text{Dirichlet}(\alpha)$
   (b) For n^{th} word in document d:
      i. Draw a topic index: $z_{dn} \sim \text{Multinomial}(\theta_d)$
      ii. Draw a word: $w_{dn} \sim \text{Multinomial}(\tilde{\beta}_{z_{dn}}^t)$

Learning process: At each mini-batch t, we maximize $\log p(\beta^t | \beta^{t-1}, p^t, \alpha, D^t)$:

$$\begin{aligned}
\hat{\beta}^t &= \arg\max_{\beta^t} \Big\{ \log p(\beta^t | \beta^{t-1}, p^t, \alpha, D^t) \Big\} \\
&= \arg\max_{\beta^t} \Big\{ \log p(\beta^t, D^t | \beta^{t-1}, p^t, \alpha) \Big\} \\
&= \arg\max_{\beta^t} \Big\{ \log p(\beta^t | \beta^{t-1}) + \log p(D^t | p^t, \beta^t, \alpha) \Big\} \qquad (12)
\end{aligned}$$

We have a lower bound on $\log p(D^t|p^t,\beta^t,\alpha)$, which is the same as (7):

$$\begin{aligned}
&\log p(D^t|p^t,\beta^t,\alpha) = \log \sum_{\pi^t} p(D^t,\pi^t|p^t,\beta^t,\alpha) = \log \sum_{\pi^t} q(\pi^t|\lambda^t)\frac{p(D^t,\pi^t|p^t,\beta^t,\alpha)}{q(\pi^t|\lambda^t)} \\
&\geq \sum_{\pi^t} q(\pi^t|\lambda^t)\log\frac{p(D^t,\pi^t|p^t,\beta^t,\alpha)}{q(\pi^t|\lambda^t)} = \sum_{\pi^t} q(\pi^t|\lambda^t)\log\frac{p(D^t|\pi^t,\beta^t,\alpha)p(\pi^t|p^t)}{q(\pi^t|\lambda^t)} \\
&= E_{q(\pi^t|\lambda^t)}\left[\log p(D^t|\pi^t,\beta^t,\alpha)\right] - KL\left[q(\pi^t|\lambda^t)||p(\pi^t|p^t)\right] \\
&\simeq \log p(D^t|\tilde{\pi}^t,\beta^t,\alpha) - KL\left[q(\pi^t|\lambda^t)||p(\pi^t|p^t)\right]
\end{aligned} \tag{13}$$

As mentioned above, due to the difficulties in directly optimizing $p(D^t|\tilde{\pi}^t,\beta^t,\alpha)$, inference for local variables θ and z can be done by using Mean-field variational inference as in the original paper [4]. In particular, for each document d: $q(\theta_d, z_d|\gamma_d,\phi_d) = q(\theta_d|\gamma_d)\prod_{n\in[N_d]} q(z_{dn}|\phi_{dn})$ with the variational distributions: $q(\theta_d|\gamma_d) = Dirichlet(.|\gamma)$ and $q(z_{dn}|\phi_{dn}) = Multinomial(.|\phi_{dn})$ where γ_d and ϕ_d are variational parameters. According to [4], these parameters for each document d are updated until convergence as follow:

$$\gamma_{dk} \leftarrow \alpha_k + \sum_{n=1}^{N_d}\phi_{dnk} \text{ for } k = 1,...,K \tag{14}$$

$$\phi_{dnk} \propto \exp(\mathbb{E}_q[\log\theta_{dk}] + \sum_{v=1}^{V}\mathbb{I}[w_{dn}=v]\log\tilde{\beta}_{kv}) \tag{15}$$

where $[V] = \{1,...,V\}$, $\mathbb{I}[.]$ is an indicator function that equals 1 if the condition is true.

As the topics are independent, we consider the objective function with respect to β_k^t, λ_k^t and p^t:

$$\begin{aligned}
&\{\hat{\beta}_k^t, \hat{\phi}_k^t, \hat{\lambda}_k^t, \hat{p}_k^t\} \\
&= \underset{\beta_k^t,\phi_k^t,\lambda_k^t,p_k^t}{\arg\max}\left\{\log p(\beta_k^t|\beta_k^{t-1}) + \sum_{d=1}^{M}\sum_{n=1}^{N_d}\log p(w_{dn}|z_{dn},\beta_k^t,\tilde{\pi}_k^t) - KL[q(\pi_k^t|\lambda_k^t)||p(\pi_k^t|p_k^t)]\right\} \\
&= \underset{\beta_k^t,\phi_k^t,\lambda_k^t,p_k^t}{\arg\max}\left\{-\frac{1}{2\sigma^2}\|\beta_k^t - \beta_k^{t-1}\|_2^2 + \sum_{d=1}^{M}\sum_{n=1}^{N_d}\sum_{j=1}^{V}\phi_{dnk}\mathbb{I}[w_{dn}=j]\beta_{kj}^t\tilde{\pi}_{kj}^t\right. \\
&\left. - \sum_{d=1}^{M}\sum_{n=1}^{N_d}\sum_{j=1}^{V}\phi_{dnk}\mathbb{I}[w_{dn}=j]\log(\sum_{i=1}^{V}\exp(\beta_{ki}^t\tilde{\pi}_{ki}^t)) - KL[q(\pi_k^t|\lambda_k^t)||p(\pi_k^t|p_k^t)]\right\}
\end{aligned} \tag{16}$$

Algorithm 3 briefly describes the learning process of aiDropout for LDA.

Algorithm 3 aiDropout training for LDA

Input: Data sequence $\{D^1, D^2, ...\}$, variance σ^2, τ
Output: Global variable β, λ, p

Initialize β^0 randomly
for t^{th} *mini-batch with data* D^t **do**
  Set an initial estimate for λ^t
  repeat
    Draw dropout matrix π^t randomly
    for *each document* d *in* D^t **do**
      Infer (γ_d, ϕ_d) by alternatively updating (14) and (15)
    Find $(\beta_k^t, \lambda_k^t, p^t)$ by maximizing (16)
  until *convergence*;

4.2 Case study 2: when NB is the base model

NB is a popular supervised model for text classification. We will use NB as a base model to evaluate how our framework works in the streaming environment.

Suppose that each mini-batch consists of M documents, each document d contains N_d words and belongs to a class $c_d \in \{1, 2, ..., C\}$. Each c_d is generated by: $c_d \sim Multinomial(\alpha)$ in which α is a fixed symmetric vector, and finally β of size $C \times V$ is the class distribution over V words in the vocabulary.

The generative process for each mini-batch t is as follows:

1. Draw β^t: $\beta_c^t \sim N(\beta_c^{t-1}, \sigma^2 I)$
2. Draw π^t: $\pi_{cj}^t \sim \text{Bernoulli}(p_{cj}^t)$
3. Calculate the class matrix: $\tilde{\beta}_{cj}^t = \text{softmax}(\beta_c^t \odot \pi_c^t)_j$
4. Each document d is drawn by:
   (a) Choose the class label $c_d \sim \text{Multinomial}(\alpha)$
   (b) Draw n^{th} word $w_{dn} \sim \text{Multinominal}(\tilde{\beta}_{cd}^t)$

Learning process:

At each mini-batch t, our goal is to maximize $\log p(\beta^t | \beta^{t-1}, p^t, c, D^t)$:

$$\begin{aligned}
\hat{\beta}^t &= \arg\max_{\beta^t} \Big\{ \log p(\beta^t | \beta^{t-1}, p^t, c, D^t) \Big\} \\
&= \arg\max_{\beta^t} \Big\{ \log p(\beta^t, D^t | \beta^{t-1}, p^t, c) \Big\} \\
&= \arg\max_{\beta^t} \Big\{ \log p(\beta^t | \beta^{t-1}) + \log p(D^t | p^t, \beta^t, c) \Big\} \qquad (17)
\end{aligned}$$

Same as (7), we have a lower bound on $\log p(D^t|p^t, \beta^t, c)$:

$$\begin{aligned}
&\log p(D^t|p^t, \beta^t, c) \\
&= \log \sum_{\pi^t} p(D^t, \pi^t|p^t, \beta^t, c) = \log \sum_{\pi^t} q(\pi^t|\lambda^t) \frac{p(D^t, \pi^t|p^t, \beta^t, c)}{q(\pi^t|\lambda^t)} \\
&\geq \sum_{\pi^t} q(\pi^t|\lambda^t) \log \frac{p(D^t, \pi^t|p^t, \beta^t, c)}{q(\pi^t|\lambda^t)} = \sum_{\pi^t} q(\pi^t|\lambda^t) \log \frac{p(D^t|\pi^t, \beta^t, c) p(\pi^t|p^t)}{q(\pi^t|\lambda^t)} \\
&= E_{q(\pi^t|\lambda^t)} \left[\log p(D^t|\pi^t, \beta^t, c)\right] - KL\left[q(\pi^t|\lambda^t)||p(\pi^t|p^t)\right] \\
&\simeq \log p(D^t|\tilde{\pi}^t, \beta^t, c) - KL\left[q(\pi^t|\lambda^t)||p(\pi^t|p^t)\right]
\end{aligned} \tag{18}$$

For each class c, the objective function with respect to β_c^t and λ_c^t is:

$$\begin{aligned}
\{\hat{\beta}_c^t, \hat{\lambda}_c^t, \hat{p}_c^t\} = \underset{\beta_c^t, \lambda_c^t, p_c^t}{\arg\max} \Bigg\{ & \log p(\beta_c^t|\beta_c^{t-1}) + \left[\sum_{d=1}^{D_c^t} \sum_{n=1}^{N_d} \log p(w_{dn}|c_d, \beta_c^t, \tilde{\pi}_c^t)\right] \\
& - KL[q(\pi_c^t|\lambda_c^t)||p(\pi_c^t|p_c^t)] \Bigg\} \\
= \underset{\beta_c^t, \lambda_c^t, p_c^t}{\arg\max} \Bigg\{ & -\frac{1}{2\sigma^2} \|\beta_c^t - \beta_c^{t-1}\|_2^2 + \sum_{d=1}^{D_c^t} \sum_{n=1}^{N_d} \sum_{j=1}^{V} \mathbb{I}[w_{dn} = j] \beta_{cj}^t \tilde{\pi}_{cj}^t \\
& - N_c \log(\sum_{i=1}^{V} \exp(\beta_{ci}^t \tilde{\pi}_{cj}^t)) - KL[q(\pi_c^t|\lambda_c^t)||p(\pi_c^t|p_c^t)] \Bigg\}
\end{aligned} \tag{19}$$

where D_c^t includes all documents which belong to class c, N_c is the total number of words in all documents belonging to class c. We use a gradient-based algorithm to maximize $F(\beta_c^t, \lambda_c^t, p_c^t)$ with respect to $\beta_c^t, \lambda_c^t, p_c^t$.

5 Discussions

In this section, we discuss some aspects of aiDropout. First, we analyse how existing frameworks and aiDropout deal with the stability-plasticity dilemma. Then, we present the role of Dropout in our framework and prove that aiDropout provides a data-dependent regularization.

5.1 The stability-plasticity dilemma

In this subsection, we investigate how different streaming learning frameworks trade off stability against plasticity in models similar to LDA[2], i.e., how they balance between old and new information from data streams. In particular, SVB [6] uses the variational parameter of the global variable β^t at mini-batch t, which we denote by λ^t, as the parameter of the Dirichlet prior distribution at mini-batch $t+1$. In other words, for each $k \in \{1, \cdots, K\}$, β_k^{t+1} has the prior distribution $Dir(\beta_k^{t+1}|\lambda_k^t)$ (Dir is Dirichlet distribution). Then we have:

[2] Such models require the global variable β to be in a simplex, e.g., NB.

Theorem 1 *In SVB:* $\mathbb{E}[\beta_{kj}^{t+1}] = \beta_{kj}^{t}$ *and* $Var[\beta_{kj}^{t+1}] \to 0$ *as* $t \to \infty$.

Proof SVB [6] proposes recursive updating of the variational distribution. For LDA (conjugate models, exponential family, i.i.d. data), the variational parameter λ^t of global variable β^t is updated by: $\lambda^t = \lambda^{t-1} + \tilde{\lambda}^t$, where λ^{t-1} is made available from the previous mini-batch and $\tilde{\lambda}^t$ is the learned information from the current mini-batch. In other words, λ^t is the addition of the learned information from all previous steps:

$$\lambda^t = \tilde{\lambda}^t + \cdots + \tilde{\lambda}^1 + \lambda^0$$

where:

$$\begin{aligned}\|\tilde{\lambda}^t\|_1 &= \sum_{k=1}^{K}\sum_{j=1}^{V}\tilde{\lambda}_{kj}^t = \sum_{d\in D^t}\sum_{n=1}^{N_d}\sum_{k=1}^{K}\sum_{j=1}^{V}\phi_{dnk}\mathbb{I}[w_{dn}=j] \\ &= \sum_{d\in D^t}\sum_{n=1}^{N_d}\sum_{k=1}^{K}\phi_{dnk} = \sum_{d\in D^t} N_d \geq 1\end{aligned}$$

Therefore, $||\lambda^t||_1 = \sum_{t=1}^{T}\sum_{d\in D^t} N_d \geq t$, which approaches infinity as t goes to infinity. When a new mini-batch $t+1$ arrives, λ^t will be used as the parameter of the prior: $p(\beta_k^{t+1}|\lambda_k^t) = Dir(.|\lambda_k^t)$. This distribution has the expectation:

$$\mathbb{E}[\beta_k^{t+1}] \propto \lambda_k^t = \beta_k^t$$

and the variance:

$$\mathrm{Var}[\beta_{kj}^{t+1}] = \frac{\lambda_{kj}^t(\sum_{i=1}^{V}\lambda_{ki}^t - \lambda_{kj}^t)}{(\sum_{i=1}^{V}\lambda_{ki}^t)^2(\sum_{i=1}^{V}\lambda_{ki}^t + 1)}$$

which varies inversely with the size of λ_k^t. As $t \to \infty$, leading to $||\lambda^t||_1 \to \infty$, we have $\mathrm{Var}[\beta_{kj}^{t+1}] \to 0$ □.

This problem is potentially present in SVB-PP [38], albeit λ^t takes longer to accumulate: $\lambda^t = \rho\lambda^{t-1} + (1-\rho)\eta + \tilde{\lambda}^t$, where ρ is the forgetting factor ($0 < \rho < 1$) and η is the uninformative prior.

When this happens, SVB and SVB-PP expect the model at time $t+1$ to be nearly identical to the model at time t. This phenomenon essentially says that a model will evolve very slowly and have difficulties in learning new information, thus could not deal well with sudden changes in the environment.

aiDropout does not encounter this problem. In aiDropout, we have an easy mechanism to balance the information between old and new data. Indeed, to maximize the objective function $F(\beta_k^t) = -\frac{1}{2\sigma^2}||\beta_k^t - \beta_k^{t-1}||_2^2 + \log p(D^t|\tilde{\pi}_k^t, \beta_k^t)$, we need to consider both components. While the first term encourages new model β^t to fluctuate around the previously learned β^{t-1}, the latter allows model to accommodate information from new data D^t. In other words, aiDropout helps model to flexibly learn new information, while retaining relevant information from historical observations to maintain the stability.

The balancing ability of aiDropout is easily controlled by the variance σ^2. The bigger σ^2 is, the more we focus on learning new information, rather than keeping old information, and vice versa. This balance is unchanged throughout the learning process. Unlike aiDropout, SVB and SVB-PP cannot control this balance. Particularly, in LDA, SVB and SVB-PP become too rigid and unable to learn new information after receiving a large amount of data, due to the reason mentioned above.

5.2 The role of Dropout in aiDropout

In streaming environments, the problem of noisy and sparse data is unavoidable. Specifically, learning from noisy data can potentially overfit models, while sparsity in data may not provide enough relevant information to make good predictions for unseen data, both leading to poor performance.

To overcome these challenges, we propose to utilize Dropout by omitting randomly a number of elements of the global variable β^t at each mini-batch t. Dropout in our framework has two main roles. Firstly, we theoretically prove that it plays as a data-dependent regularizer, which makes aiDropout more robust against overfitting. Moreover, in our framework, Dropout is used throughout the data stream, leading to a special effect, which is ensemble learning. Indeed, at each mini-batch in the training process, the use of Dropout is equivalent to sampling a single learner from a set of $2^{K\times V}$ possible learners. Then, by rescaling β^t with $\mathbb{E}[\pi^t]$, $2^{K\times V}$ learners with shared parameters can be combined into a single learner to be used at test time. Therefore, methodically, we would like to remark that there is not much difference between the dropout technique in aiDropout and the original one used widely in deep learning. However, we also agree that there would be certain differences in terms of the ensembling efficiency. Concretely, in deep neural nets, the desirable effect of Dropout ensemble could be interpreted well via the functional behaviors (such as diversity, mutual explanation) of the predictive distribution. Whilst the similar effect in our method needs further investigation for better understanding.

The ability to prevent overfitting and the ensemble property make iDropout have better generalization on future data, which is specially important in streaming learning, because data streams can be non-stationary and have high uncertainty.

5.3 Dropout in aiDropout as adaptive data-dependent regularization

The learning process at each mini-batch in aiDropout for LDA and NB can be reduced to maximizing the objective function of the following form:

$$F = -\frac{1}{2\sigma^2}||\beta_k - \beta_k^{prev}||_2^2 + \sum_{j=1}^{V} u_{kj}\mathbb{E}_{q(\pi_k|\lambda_k)} \log\left(\text{softmax}(\beta_k \odot \pi_k)_j\right) - KL[q(\pi_k|\lambda_k)||p(\pi_k|p_k)] \quad (20)$$

where β_k^{prev} is made available from the previous mini-batch, λ_k is the drop rate that needs to be learned on the current mini-batch (we omit superscript t for simplicity) and u_{kj} is defined as follows:

$$u_{kj} = \begin{cases} \sum_{d=1}^{M} \sum_{n=1}^{N_d} \phi_{dnk} \mathbb{I}[w_{dn} = j] \text{ in LDA} \\ \sum_{d \in D_c^t} \sum_{n=1}^{N_d} \mathbb{I}[w_{dn} = j] \text{ in NB} \end{cases}$$

Consider $x_1, \cdots, x_K$ as K-dimension one-hot vectors (x_k has only k^{th} element activated) and $\beta = [\beta_1 \beta_2 \cdots \beta_V]$ where β_j is j^{th} column of matrix β, then:

$$\text{softmax}(\beta_k)_j = \exp(s_{kj} - A(s_k))$$

with $s_{kj} = \beta_j^T x_k$ is a undropped score value and $A(s_k) = \log \sum_{i=1}^{V} \exp(s_{ki})$ is the log-partition function.
Assume π_k is drawn from $q(\pi_k|\lambda_k)$ which is a Bernoulli distribution parameterized by λ_k, corresponding to the Inverted Dropout:

$$q(\pi_{ij} = 1/(1-\lambda_k)|\lambda_k) = 1 - \lambda_k, q(\pi_{ij} = 0|\lambda_k) = \lambda_k$$

then $\mathbb{E}_{q(\pi_k|\lambda_k)}[\pi_{kj}] = 1$, and:

$$\text{softmax}(\beta_k \odot \pi_k)_j = \exp(\tilde{s}_{kj} - A(\tilde{s}_k))$$

with $\tilde{s}_{kj} = (\beta_i \odot \pi_i)^T x_k$, $A(\tilde{s}_k) = \log \sum_{i=1}^{V} \exp(\tilde{s}_{ki})$. Using this notation, we can write F as:

$$F = -\frac{1}{2\sigma^2} ||\beta_k - \beta_k^{prev}||_2^2 + \sum_{j=1}^{V} u_{kj} \mathbb{E}_{q(\pi_k|\lambda_k)}[\tilde{s}_{kj} - A(\tilde{s}_k)] - KL[q(\pi_k|\lambda_k)||p(\pi_k|p_k)]$$

Since $\mathbb{E}_{q(\pi_k|\lambda_k)}[\pi_{kj}] = 1$ so the dropout technique preserves mean, leading to $\mathbb{E}_{q(\pi_k|\lambda_k)}[\tilde{s}_{kj}] = s_{kj}$, we have:

$$\begin{aligned} \mathbb{E}_{q(\pi_k|\lambda_k)}[\tilde{s}_{kj} - A(\tilde{s}_k)] &= s_{kj} - A(s_k) - (\mathbb{E}_{q(\pi_k|\lambda_k)}[A(\tilde{s}_k)] - A(s_k)) \\ &= \text{softmax}(\beta_k)_j - (\mathbb{E}_{q(\pi_k|\lambda_k)}[A(\tilde{s}_k)] - A(s_k)) \end{aligned}$$

Then we can write:

$$\begin{aligned} F = &- \frac{1}{2\sigma^2} ||\beta_k - \beta_k^{prev}||_2^2 + \sum_{j=1}^{V} u_{kj} \log \left(\text{softmax}(\beta_k)_j\right) \\ &- (\mathbb{E}_{q(\pi_k|\lambda_k)}[A(\tilde{s}_k)] - A(s_k)) \sum_{j=1}^{V} u_{kj} - KL[q(\pi_k|\lambda_k)||p(\pi_k|p_k)] \end{aligned}$$

Since the log-partition function $A(.)$ is convex, $(\mathbb{E}_{q(\pi_k|\lambda_k)}[A(\tilde{s}_k)] - A(s_k))$ is always positive by Jensen's inequality and can therefore be interpreted as a regularizer. Indeed, applying second-order Taylor approximation to $A(\tilde{s}_k)$ around the undropped score vector s_k, we have means and covariances of the dropout features:

$$A(\tilde{s}_k) = A(s_k) + \nabla A(s_k)^T(\tilde{s}_k - s_k) + \frac{1}{2}(\tilde{s}_k - s_k)^T\nabla^2 A(s_k)(\tilde{s}_k - s_k)$$

then we obtain a following regularizer:

$$\mathbb{E}_{q(\pi_k|\lambda_k)}[A(\tilde{s}_k)] - A(s_k) = \frac{1}{2}\mathbb{E}_{q(\pi_k|\lambda_k)}[(\tilde{s}_k - s_k)^T\nabla^2 A(s_k)(\tilde{s}_k - s_k)]$$
$$= \frac{1}{2}\mathrm{Tr}[\nabla^2 A(s_k)\mathrm{Cov}_{q(\pi_k|\lambda_k)}(\tilde{s}_k)] = \frac{1}{2}\sum_{j=1}^{V}\mu_{kj}(1-\mu_{kj})\mathrm{Var}_{q(\pi_k|\lambda_k)}[\tilde{s}_{kj}]$$
$$= \frac{1}{2}\sum_{j=1}^{V}\mu_{kj}(1-\mu_{kj})\beta_j^T\mathrm{Cov}_{q(\pi_k|\lambda_k)}(x_k)\beta_j$$

where $\mu_{kj} = \mathrm{softmax}(s_k)_j$ is the model probability, the variance $\mu_{kj}(1-\mu_{kj})$ measures model uncertainty, and

$$\beta_j^T\mathrm{Cov}_{q(\pi_k|\lambda_k)}(x_k)\beta_j = \sum_{m=1}^{K}\frac{\lambda_k}{1-\lambda_k}x_{km}^2\beta_{mj}^2 = \frac{\lambda_k}{1-\lambda_k}\beta_{kj}^2$$

Hence, $\mathbb{E}_{q(\pi_k|\lambda_k)}[A(\tilde{s}_k)] - A(s_k) = \frac{\lambda_k}{2(1-\lambda_k)}\sum_{j=1}^{V}\mu_{kj}(1-\mu_{kj})\beta_{kj}^2$ has quadratic format w.r.t β_k. In other words, the effect of Dropout in aiDropout is equivalent to a L2-regularization $R(\beta)$:

$$R(\beta) = (\mathbb{E}_{q(\pi_k|\lambda_k)}[A(\tilde{s}_k)] - A(s_k))\sum_{j=1}^{V}u_{kj}$$
$$= \frac{\lambda_k}{2(1-\lambda_k)}\sum_{j=1}^{V}\left[\mu_{kj}(1-\mu_{kj})\sum_{j=1}^{V}u_{kj}\right]\beta_{kj}^2.$$

This is a theoretical interpretation on the ability of iDropout to reduce overfitting. Unlike other regularization techniques, each β_{kj} in aiDropout has a different regularization parameter $\frac{\lambda_k}{2(1-\lambda_k)}\mu_{kj}(1-\mu_{kj})\sum_{j=1}^{V}u_{kj}$, depending on the input data. This is interesting, since this data-dependent regularization allows each β_{kj} to have its own search space to catch the geometric property of data. With this property, dropout in our method is more effective than other standard computationally inexpensive regularizers, such as weight decay, filter norm constraints and sparse activity regularization [16].

6 Empirical evaluation

This section will present extensive experiments to evaluate how our methods (iDropout and aiDropout) and baselines deal with two challenges: Short and noisy texts and stability-plasticity dilemma. In terms of short and noisy texts, we use two popular scenarios: Evaluating on a hold-out test set and evaluating on consecutive arriving mini-batches. The former scenario [6,40] is often used to examine the performance of the methods in simulated streaming environments on datasets without time stamp. Meanwhile, the latter scenario [38,39] helps evaluate them

Table 1: 6 datasets without time stamp

Dataset	Vocab size	Training size	Testing size	words/doc
20Newsgroups	24905	17846	1000	88.2
Grolier	15269	23044	1000	79.9
TMN	11599	31604	1000	24.3
TMN-title	2823	26251	1000	4.6
Yahoo-title	21439	517770	10000	4.6
NYT-title	46854	1664127	10000	5.0

on actual streaming environments. Regarding stability-plasticity dilemma, we investigate how the methods deal with concept drift and forgetting the knowledge learned from past data.

6.1 Baselines

We compare aiDropout[3] to iDropout [46], SVB [6], SVB-PP [38][4], PVB [40]. We select the best result of each method by using grid search. The range of each parameter is as follows:

- the forgetting factor $\rho \in \{0.5, 0.6, 0.7, 0.8, 0.9, 0.99\}$ for SVB-PP.
- the population size $\nu \in \{10^3, 10^4, 10^5, 10^6, 5.10^3, 5.10^4, 5.10^5, 5.10^6\}$ and dimming factor $\kappa \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$ for PVB.
- the variance $\sigma^2 \in \{0.01, 0.1, 1, 10, 100\}$ and the drop rate $dr \in \{0, 0.15, 0.25, 0.35\}$ for iDropout.
- the variance $\sigma^2 \in \{0.01, 0.1, 1, 10, 100\}$, the hyperparameter of the Gumbel-Softmax distribution $\tau = 0.01$ for aiDropout.

Moreover, for iDropout and aiDropout, the gradient-based algorithm is Adagrad with learning rate 0.01, the maximum number of Adagrad iterations 100, and the number of iterations between the updating phases of local and global parameters in each mini-batch is set to 10.

6.2 Experiments on noisy and sparse data

To evaluate how the methods deal with noisy and sparse data, we conduct extensive experiments with both chronological and non-chronological datasets. While the two chronological datasets (The Irish Times and News Aggregator) have available published time for each document, the six non-chronological datasets do not have this information. On the non-chronological datasets, we follow the experimental scenarios of prior studies [6, 40, 10, 2, 55] to create a sequence of mini-batches for training and a hold-out set for evaluating after having finished traininig each mini-batch. The experiments not only show how the methods deal with sparse and noisy data but also consider the stability of the methods when they are evaluated on the same hold-out test set. Meanwhile, we follow the experimental scenarios of recent

[3] The implementation of aiDropout and iDropout is available at https://github.com/pvh1602/aiDropout

[4] SVB-HPP is not included since its application requires non-trivial efforts. Further, as observed by [38], SVB-HPP is often comparable to the best SVB-PP

studies [38,39,55] on the two chronological datasets to examine how the methods work on actual noisy and sparse data streams.

6.2.1 Experiments on datasets without time stamp

Base model and datasets: In this subsection, we use LDA as our base model. As mentioned in Section 4, LDA, a popular Bayesian model, is widely applied to uncover hidden topics. We analyze how the methods deal with sparse and noisy data by using six non-chronological datasets, including two long text datasets (20Newsgroups [5], Grolier [6]) and four short text ones (TagMyNews (TMN)[7], TagMyNews-title (TMN-title), Yahoo-title, NYT-title[8] with some statistics in Table 1.

Settings: To simulate streaming data, we randomly shuffle and then divide each dataset into a sequence of mini-batches with batchsize: 500 for Grolier, 20Newsgroups, TMN, TMN-title; 5000 for NYT-title, Yahoo-title. We set prior of topic mixture $\alpha = 0.01$; the number of topic $K = 50$ for Grolier, 20Newsgroups, TMN, TMN-title; $K = 100$ for NYT-title, Yahoo-title. We note that batchzise and the number of topics are selected based on the sizes of datasets. We can consider the stability of our methods (aiDropout and iDropout) compared to the others when they are evaluated on the same hold-out test set after having finished training each mini-batch. Moreover, we conduct experiments with different dropout rates ($dr \in \{0; 0.15; 0.25; 0.35\}$) for iDropout to show the sensitivity of iDropout w.r.t dropout rate.

Evaluation metric: Log Predictive Probability (LPP) [24] and Normalized Pointwise Mutual Information (NPMI) [5] are used. While LPP measures the generalization of a model on unseen data, NPMI is used to examine the coherence and interpretability of the learned topics. LPP calculates the probabilities of a part of a test document given the remaining part and trained model's parameters. NPMI is based on the co-occurrence of pairs of top words of learned topics. The details on the calculation of these two metrics are given in the Appendix B.

Experimental results:

Fig 2 shows the performance of all methods. The results of aiDropout roughly approximate that of iDropout and are better than the others. For short text datasets (NYT-title, Yahoo-title,TMN, and TMN-title), it is quite likely to encounter the unwanted properties of data such as noise and sparsity in these datasets. While noisy data may lead to overfitting, sparse data causes the model to make wrong predictions due to the lack of information. Thanks to the benefits of dropout, our methods can address this problem effectively. In contrast, the other methods do not have efficient ways to deal with this issue, hence give poor performance. The LPPs and NPMIs of baselines reduce significantly although more mini-batches arrive. It means that the baselines suffer from overfitting on short

[5] http://qwone.com/ jason/20Newsgroups/
[6] https://cs.nyu.edu/ roweis/data.html
[7] http://acube.di.unipi.it/tmn-dataset/
[8] http://archive.ics.uci.edu/ml/datasets/Bag+of+Words

and noisy datasets. We also consider the performances of the methods on long (regular) text datasets (20Newsgropus and Grolier). It is obvious that the baselines do not suffer from decreasing both LPP and NPMI when more data comes. SVB and SVB-PP become too stable once received large enough data. This may explain why the results of these two methods are roughly unchanged. It can be seen that PVB does not encounter this problem and has a considerable evolution over time. Our proposed methods also overcome this issue and have superior results.

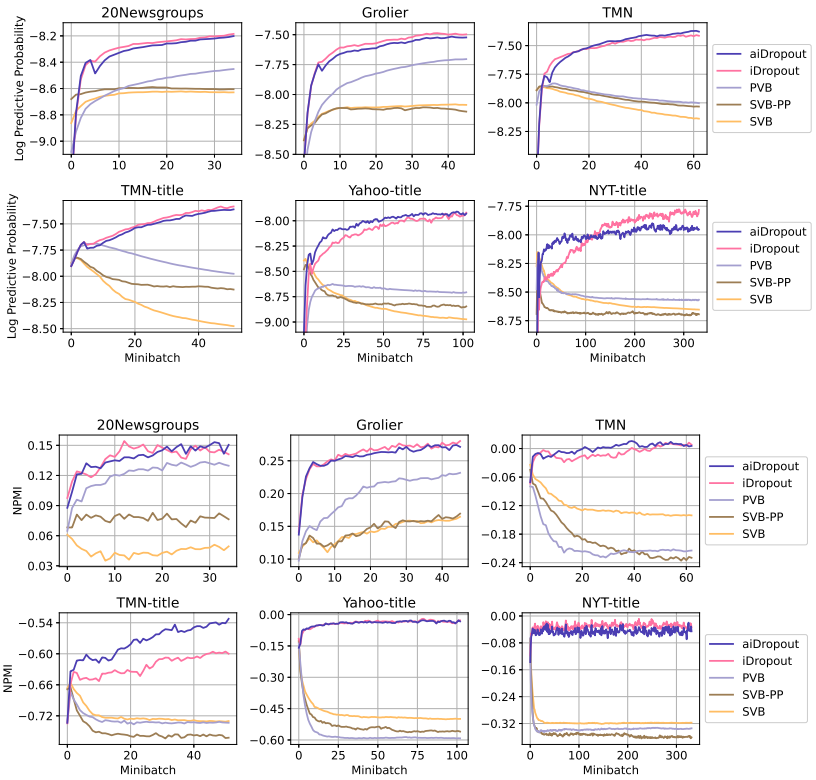


Fig. 2: Performance of the 5 methods on datasets without time stamp. LDA is the base model. Higher is better.

Fig 3 shows the results of LPP measurement on aiDropout and iDropout with different drop rates($dr \in \{0, 0.15, 0.25, 0.35\}$). It can be seen that aiDropout with adaptive drop rate has different effects on different datasets. It is remarkable to see that with four short text datasets (NYT-title, Yahoo-title,TMN, and TMN-title) that have two typical properties in streaming data, i.e., noise and sparsity, the results of aiDropout outperform that of iDropout with most drop rates. This may be explained that while iDropout with fixed drop rate is not flexible in handling noisy and sparse data, aiDropout enables the drop rate to be automatically

adapted along the change in data. However, with two long text datasets (20Newsgropus and Grolier), aiDropout seems not to work as well as itself with the short datasets. This results could be clarified that as training on these long datasets does not severely meet the problem of noise and sparsity, learning the drop rate does not give a significant improvement.

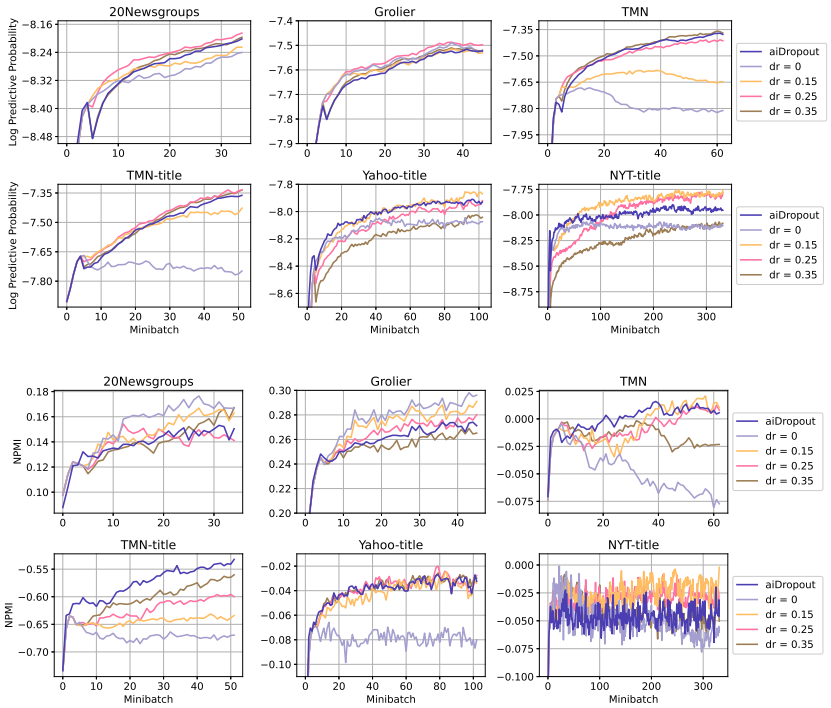


Fig. 3: Performance of aiDropout compared to iDropout with different drop rates. LDA is the base model. Higher is better.

6.2.2 Experiments on datasets with time stamp

Base model and datasets: In this subsection, LDA is used as a base model for topic modeling and NB for classification. We will study how the methods work on actual data streams on two chronological datasets which are The Irish Times dataset [9] and News Aggregator dataset [10]. Particularly, The Irish Times corpus contains 1376099 data instances from 02/01/1996 to 31/12/2017 with 6 classes and its vocab size is 25328. News Aggregator dataset includes 422937 news stories between 10/03/2014 and 10/08/2014 with 4 classes and its vocab size is 25509.

[9] https://www.kaggle.com/therohk/ireland-historical-news/
[10] https://www.kaggle.com/uciml/news-aggregator-dataset

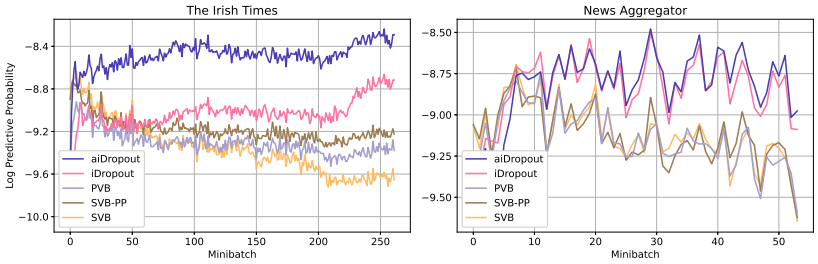


Fig. 4: Performance of the 5 methods on datasets with time stamp. LDA is the base model. Higher is better.

Settings: When evaluating using LDA, we simply throw away labels and use $K = 100$ and $\alpha = 0.01$. We also divide the whole datasets into mini-batches in which each mini-batch corresponds to a month in The Irish Times and two consecutive days in News Aggregator. To find out how our proposed frameworks act in streaming environments compared with other methods, we use documents of the next mini-batch to evaluate the model at any mini-batch. Additionally, we also study on how sensitive iDropout is when dropout rate is tuned, particularly we set $dr \in \{0; 0.1; 0.3; 0.5\}$ with The Irish Times dataset and $dr \in \{0; 0.2; 0.4; 0.6; 0.8\}$ with News Aggregator dataset when NB is base model.

Evaluation metric: We use LPP to evaluate the learned topic model in LDA and accuracy to evaluate the classification performance in NB.

Experimental results on LDA: The results are shown in Fig. 4. It is clear that our methods with dropout outperform the others. Comparing the results of aiDropout and iDropout on both of the two datasets, we see that while the performance of aiDropout on The Irish Times is much better than that of iDropout, the two frameworks give similar results on News Aggregator. This could be due to the small number of mini-batches, and the fairly large number of data points per mini-batch on News Aggregator, hence the aiDropout cannot enhance significantly compared to the one with fixed drop rate. It can also be easily seen that the methods with no dropout suffer from overfitting and decline in performance as learning from more data, especially SVB. In contrast, our proposed methods with dropout demonstrate the effectiveness of handling overfitting. This may be explained that because both two datasets are short-text and contain unwanted properties such as noise and sparsity, the use of dropout helps our methods reduce overfitting, and hence obtains better generalization.

Experimental results on NB:

Fig. 5 shows the performance of five methods on classfication task. In particular, the results of aiDropout are slightly better than that of iDropout. Compared to the others with no dropout, the performance of aiDropout with adaptive drop rate gets about 6-8% better than SVB, and about 3-4% better than SVB-PP and PVB on The Irish Times, about 5-6% better than SVB and SVB-PP, and about 1-2% better than PVB on News Aggregator. It is clear that dropout plays a crucial

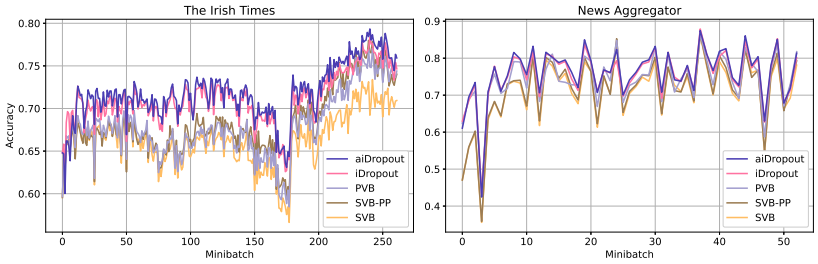


Fig. 5: Performance of the 5 methods on datasets with time stamp. NB is the base model. Higher is better.

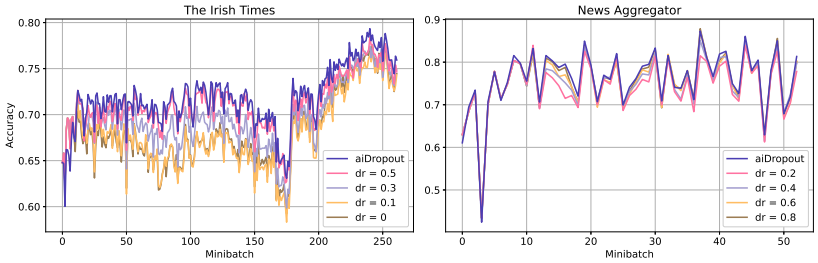


Fig. 6: Performance of aiDropout compared to iDropout with different drop rates. NB is the base model. Higher is better.

role in helping our framework work effectively in data streams. We also notice that about the 175th mini-batch on The Irish Times dataset, the results of all methods drop due to abnormal changes. Again, thanks to the benefits of dropout, our proposed methods do not fall too deeply and then recover quickly to keep leading on the successive mini-batches.

Fig. 6 shows the accuracy for aiDropout and iDropout with different settings of drop rate ($dr \in \{0, 0.1, 0.3, 0.5\}$ with The Irish Times, and $dr \in \{0.2, 0.4, 0.6, 0.8\}$ with News Aggregator). We observe that the performance of aiDropout is slightly higher than that of iDropout with the best drop rate setting (dr = 0.5 with The Irish Times, and dr = 0.2 with News Aggregator), and outperforms the others. Specifically, the datasets used in this experiment are short-text, and hence contain significant undesirable properties such as noise and sparsity. Our framework with adaptive drop rate tends to work more flexibly than the fixed drop rate based method when dealing with this problem. It can also be seen that there is not much difference between the performance of aiDropout and various settings of iDropout on News Aggregator compared to The Irish Times. It seems that when the number of mini-batches on News Aggregator is small and the number of data points per mini-batch is quite large, aiDropout and iDropout can obtain similar results.

6.3 Balancing stability and plasticity

In this subsection, we consider how the methods balance stability and plasticity when training on data streams. We design experimental scenarios with various kinds of concept drifts to evaluate the plasticity of the methods for adapting to new concepts. Meanwhile, we examine the forgetting phenomenon of the methods to evaluate their stability.

6.3.1 Evaluation on sudden concept drift

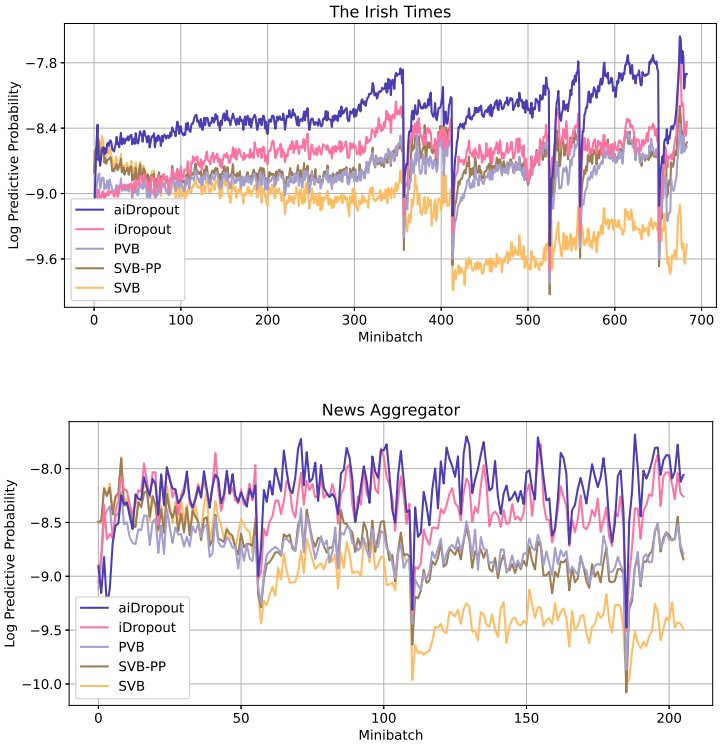


Fig. 7: Performance of the methods when facing with sudden concept drift on The Irish Times and News Aggregator datasets. LDA is the base model. Higher is better.

The problem in which the underlying relationships in the data change suddenly is referred to as sudden concept drift [14,31]. This issue is very likely to be encountered in streaming data. We evaluate how well our frameworks and other methods deal with abrupt changes in data streams. In appendix C, we present how the methods face incremental and recurring concept drifts [31].

Base model and settings: We use LDA with $K = 100$ and $\alpha = 0.01$ as our base model and two datasets (News Aggregator and The Irish Times) for this experiment. Each dataset is split into mini-batches, each mini-batch contains 2000 documents of a particular class, and all mini-batches of the same class are placed adjacent to each other. Therefore, concept drift happens noticeably when data transfers from one class to another. After learning on each mini-batch, the model is evaluated by computing LPP on the next mini-batch.

Experimental results: The result is illustrated in Fig. 7. It is clear that after each drift point, SVB recovers slowly and gives poor performance when encountering concept drift. SVB-PP and PVB seem to adapt better to concept drift. While SVB-PP uses a forgetting factor which allows it to learn new information from new data, the variance of the variational posterior in PVB never decreases below a given threshold indirectly controlled by population size α that helps adapt to concept drift. iDropout gets better result compared to the mentioned methods. This may be easily explained that iDropout has the balance mechanism which enables it to learn new underlying distribution of data. In addition, thanks to the ability to reduce overfitting and the ensemble property of dropout, the fixed drop rate based method can obtain better generalization, and hence prevent the performance from falling too deeply when facing the concept drift problem. Finally, aiDropout outperforms the others significantly. This result may be due to the drop rate adaptation over time. Particularly, The Irish Times and News Aggregator datasets are short-text datasets that can contain undesirable properties, such as noise and sparsity. aiDropout enables the drop rate to be adaptively learned corresponding to the changes in arriving data, thereby it addresses the problem of noisy and sparse data in streaming data better than the method with a fixed drop rate.

Next, we examine the methods' behaviors more thoroughly when dealing with concept drift. Following by [50, 20], we evaluate the five methods in terms of the lowest LPP achieved in a drift area, the median LPP in each concept, and restoration time for each new concept. While the lowest LPP shows the performance of methods when a new concept has just appeared, the median LPP illustrates the ability to learn this new concept from all data. Meanwhile, restoration time shows the number of required mini-batches for a method to achieve back good performance after appearing a new concept. For each new concept, restoration time T_R is calculated as follow:

$$T_R = \frac{t_2 - t_1}{T} \tag{21}$$

where t_1 is a mini-batch where the LPP drops below 95% of the median LPP of an old concept, t_2 is a mini-batch where the LPP achieves 95% of the mean LPP of the next concept, and T is the total number of mini-batches. We use again the available source code[11] from [20] to compute the mentioned measures.

Table 2 shows the performance of the five methods in terms of the average of lowest LPP, median LPP, and restoration time on all times that concepts happen. In terms of the lowest LPP, both aiDropout and iDropout achieve significantly better results than PVB, SVB-PP, and SVB. It means that aiDropout and iDropout can work better than the remaining methods on data from a new concept without

[11] https://gitlab.com/filipmg/ds-dropout-submodels/-/blob/master/evaluators/DriftEvaluator.py

Table 2: The lowest LPP, median LPP, and restoration time of the five methods when dealing with sudden concept drifts. For the lowest LPP and median LPP, higher is better. For restoration time, lower is better.

		aiDropout	iDropout	PVB	SVB-PP	SVB
The Irish Times	Lowest LPP	-9.18982	-9.18982	-9.58466	-9.6715	-9.61944
	Median LPP	-8.27008	-8.61166	-8.8322	-8.72202	-9.45928
	Restoration time	0.17194	0.25264	0.22924	0.26344	0.18918
News Aggregator	Lowest LPP	-8.99844	-9.2935	-9.52333	-9.66889	-9.80323
	Median LPP	-8.1325	-8.36248	-8.77829	-8.81617	-9.26231
	Restoration time	0.025767	0.025733	0.074067	0.069233	0.0338

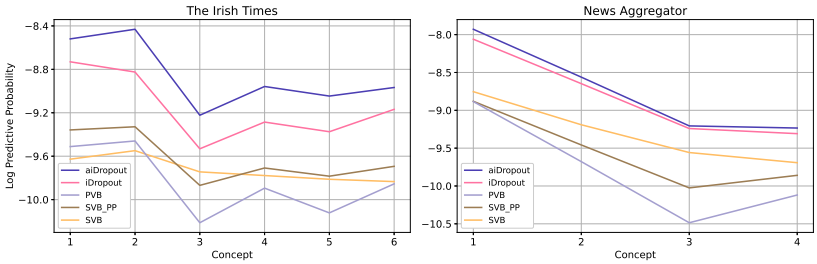


Fig. 8: Catastrophic forgetting phenomenon when training LDA on data streams. LPP is averagely calculated on hold-out test sets of past classes. Higher is better.

any training on this concept. After that, many mini-batches arrive, the LPPs of all methods increase considerably 7. Therefore, their median LPPs are noticeably higher than their lowest LPPs, respectively. Because aiDropout and iDropout have high uncertainty, they can learn new concepts well. Thus, their LPPs are higher than PVB, SVB-PP, SVB. Moreover, aiDropout uses adaptive droprate, and it obtains a better median LPP than iDropout. It is obvious that online Bayesian updating (such as SVB [6]) often suffers from the phenomenon of overconfident posterior after receiving a large enough amount of data. Consequently, it cannot learn new concepts well. After the LPPs of SVB drop significantly (the lowest LPP is low) when concept drift happens, its median LPP does not increase significantly in comparison with the lowest LPP. SVB-PP and PVB alleviate this issue, however, they do not achieve good results as aiDropout and iDropout. Regarding restoration time, aiDropout achieves the smallest value among the five methods. It means that aiDropout requires the smallest number of mini-batches to work well on a new concept in comparison with the remaining methods. Meanwhile, iDropout restore more quickly than PVB, SVB-PP, and SVB on the News Aggregator dataset, but it needs more mini-batchs on the Irish Times dataset. It is acceptable when iDropout achieves significantly better median LPP than PVB, SVB-PP, and SVB.

6.3.2 Catastrophic forgetting phenomenon when training LDA on data streams

In this subsection, we examine the stability of the methods in streaming environments. The methods often deal with the problem of forgetting knowledge acquired from past data, known as catastrophic forgetting phenomenon [45, 30, 1, 11], when

training on new data. This phenomenon is studied carefully in the continual learning field where a method must learn multiple tasks consecutively. We follow the experimental scenarios of continual learning to evaluate the catastrophic forgetting phenomenon of the methods. In detail, learning hidden topics in each class is considered as a task and tasks are learned consecutively as in the experimental scenarios of the sudden concept drift on News Aggregator and The Irish Times datasets. However, we create a hold-out test set (2000 short texts) for each class. The average LPP on the hold-out test sets of past classes is calculated after finishing training each class. The predictive ability of a method on past data shows how it deals with the forgetting problem.

Fig. 8 presents the average LPP of the methods. It is obvious that forgetting phenomenon is unavoidable for artificial intelligence as well as human beings. The average LPPs of the methods decrease when training new tasks. Albeit both iDropout and aiDropout suffer from catastrophic forgetting, their average LPPs are superior to these of remaining methods. Because our methods learn each task well and outperform the others with significant magnitudes. Moreover, some studies [17,9] practically investigated dropout in continual learning and they showed that dropout can reduce the catastrophic forgetting phenomenon. Meanwhile, according to our theoretical analyses, SVB is the most stable, therefore, the average LLPs of SVB reduce the least on both of the two datasets. While SVB-PP and PVB deal better with concept drift than SVB, they forget previous tasks more considerably than SVB. These results demonstrate stability-plasticity dilemma that all the methods must face with.

7 Conclusion

In this paper, we aim to develop a framework which helps learn a wide range of Bayesian models on data streams. We focus on two popular challenges: Noisy and sparse data and stability-plasticity dilemma to build an effective streaming method. We propose aiDropout, a novel and straightforward framework, which is based on the transition model and adaptive dropout technique to address these challenges. The transition model creates a simple mechanism to balance knowledge learned from past data and current data. In spite of simplicity, aiDropout avoids being too stable to learn new concepts. Meanwhile, the adaptive dropout brings the properties of data-dependent regularization and ensemble learning to tackle the stability-plasticity dilemma as well as handle noisy and sparse data. The extensively experimental results shows that aiDropout prevents overfitting which prior methods suffer when training in noisy and sparse data. Although the performance of aiDropout decreases dramatically when new concepts happen, aiDropout is still significantly better than other methods. Then, the performance of aiDropout increases more quickly as well as significantly than those of other methods. Moreover, our framework achieves better performances than the baselines in facing catastrophic forgetting phenomenon.

In the future, we will focus on three topics to make aiDropout more effective and impactful. Firstly, our framework needs to manually tune the parameter of the transition model σ. Therefore, a solution to automatically learn this parameter will make aiDropout more practical in actual streaming environments. Secondly, in this paper, aiDropout is merely applied to the two Bayesian models: LDA and

Multinomial naive Bayes for text mining. We will aim to exploit aiDropout for recommender systems that also deals with the noise and sparse rating matrix. Finally, this work only focuses on learning one task on data streams. In the next work, we will consider how our framework deals with multiple tasks in online continual learning.

8 Declarations

8.1 Funding

This work was funded by Gia Lam Urban Development and Investment Company Limited, Vingroup and supported by Vingroup Innovation Foundation (VINIF) under project code VINIF.2019.DA18.

8.2 Conflicts of Interest

The authors declare that they have no competing interests.

8.3 Availability of data and material

Not applicable

8.4 Code availability

The implementation is available at https://github.com/pvh1602/aiDropout.

8.5 Authors' contributions

The contributions of each author are presented as follows:

Ha Nguyen: Methodology, Software, Validation, Formal analysis, Writing - original draft, Investigation.

Hoang Pham: Methodology, Software, Validation, Formal analysis, Visualization, Investigation

Son Nguyen: Methodology, Software, Validation, Formal analysis, Writing - original draft, Visualization

Linh Ngo Van: Conceptualization, Methodology, Validation, Formal analysis, Writing - review, Visualization, Investigation.

Khoat Than: Methodology, Validation, Formal analysis, Writing - review, Supervision, Project administration, Funding acquisition.

8.6 Ethics approval

Not applicable

8.7 Consent to participate

Not applicable

8.8 Consent for publication

Not applicable

References

1. Ahn, H., Cha, S., Lee, D., Moon, T.: Uncertainty-based continual learning with adaptive regularization. In: Advances in Neural Information Processing Systems, pp. 4392–4402 (2019)
2. Anh Nguyen; Linh Ngo Van Kim Anh Nguyen, C.H.N., Than, K.: Boosting prior knowledge in streaming variational bayes. Neurocomputing **424**, 143–159 (2021)
3. Baldi, P., Sadowski, P.: The dropout learning algorithm. Artificial intelligence **210**, 78–122 (2014)
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research **3**(Jan), 993–1022 (2003)
5. Bouma, G.: Normalized (pointwise) mutual information in collocation extraction. Proceedings of GSCL pp. 31–40 (2009)
6. Broderick, T., Boyd, N., Wibisono, A., Wilson, A.C., Jordan, M.I.: Streaming variational bayes. In: Advances in Neural Information Processing Systems, pp. 1727–1735 (2013)
7. Chen, N., Zhu, J., Chen, J., Zhang, B.: Dropout training for support vector machines. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, pp. 1752–1759. AAAI Press (2014)
8. Chérief-Abdellatif, B.E., Alquier, P., Khan, M.E.: A generalization bound for online variational inference. In: Asian Conference on Machine Learning (2019)
9. De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., Tuytelaars, T.: A continual learning survey: Defying forgetting in classification tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)
10. Duc, A.N., Van Linh, N., Kim, A.N., Than, K.: Keeping priors in streaming bayesian learning. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 247–258. Springer (2017)
11. Ebrahimi, S., Elhoseiny, M., Darrell, T., Rohrbach, M.: Uncertainty-guided continual learning with bayesian neural networks. In: 8th International Conference on Learning Representations, ICLR (2020)
12. Fei-Fei, L., Perona, P.: A bayesian hierarchical model for learning natural scene categories. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 2, pp. 524–531. IEEE (2005)
13. Gal, Y., Hron, J., Kendall, A.: Concrete dropout. In: Advances in neural information processing systems, pp. 3581–3590 (2017)
14. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM computing surveys (CSUR) **46**(4), 44 (2014)
15. Gomes, H.M., Bifet, A., Read, J., Barddal, J.P., Enembreck, F., Pfharinger, B., Holmes, G., Abdessalem, T.: Adaptive random forests for evolving data stream classification. Machine Learning **106**(9), 1469–1495 (2017)
16. Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press (2016)
17. Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An empirical investigation of catastrophic forgetting in gradient-based neural networks. arXiv preprint arXiv:1312.6211 (2013)
18. Gopalan, P.K., Wang, C., Blei, D.: Modeling overlapping communities with node popularities. Advances in neural information processing systems **26**, 2850–2858 (2013)
19. Grathwohl, W., Choi, D., Wu, Y., Roeder, G., Duvenaud, D.: Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In: 6th International Conference on Learning Representations, ICLR 2018. OpenReview.net (2018). URL `https://openreview.net/forum?id=SyzKd1bCW`

20. Guzy, F., Woźniak, M.: Employing dropout regularization to classify recurring drifted data streams. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–7. IEEE (2020)
21. Ha, C., Tran, V.D., Van, L.N., Than, K.: Eliminating overfitting of probabilistic topic models on short and noisy text: The role of dropout. International Journal of Approximate Reasoning **112**, 85–104 (2019)
22. Helmbold, D.P., Long, P.M.: On the inductive bias of dropout. The Journal of Machine Learning Research **16**(1), 3403–3454 (2015)
23. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580 (2012)
24. Hoffman, M.D., Blei, D.M., Wang, C., Paisley, J.: Stochastic variational inference. The Journal of Machine Learning Research **14**(1), 1303–1347 (2013)
25. Hughes, M.C., Sudderth, E.B.: Memoized online variational inference for dirichlet process mixture models. In: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1, NIPS'13, p. 1133–1141. Curran Associates Inc., Red Hook, NY, USA (2013)
26. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. In: International Conference on Learning Representation (2017)
27. Kim, G.H., Jang, Y., Lee, J., Jeon, W., Yang, H., Kim, K.E.: Trust region sequential variational inference. In: Asian Conference on Machine Learning, pp. 1033–1048 (2019)
28. Kingma, D.P., Salimans, T., Welling, M.: Variational dropout and the local reparameterization trick. In: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, NIPS'15, p. 2575–2583. MIT Press, Cambridge, MA, USA (2015)
29. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: The International Conference on Learning Representations (ICLR) (2014)
30. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences **114**(13), 3521–3526 (2017)
31. Krawczyk, B., Cano, A.: Online ensemble learning with abstaining classifiers for drifting and noisy data streams. Applied Soft Computing **68**, 677–692 (2018)
32. Kurle, R., Cseke, B., Klushyn, A., van der Smagt, P., Günnemann, S.: Continual learning with bayesian neural networks for non-stationary data. In: 8th International Conference on Learning Representations, ICLR (2020)
33. Le, H.M., Cong, S.T., The, Q.P., Van Linh, N., Than, K.: Collaborative topic model for poisson distributed ratings. International Journal of Approximate Reasoning **95**, 62–76 (2018)
34. Liu, Y., Dong, W., Zhang, L., Gong, D., Shi, Q.: Variational bayesian dropout with a hierarchical prior. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7124–7133 (2019)
35. MacKay, D.J., Mac Kay, D.J.: Information theory, inference and learning algorithms. Cambridge university press (2003)
36. Maddison, C.J., Mnih, A., Teh, Y.W.: The concrete distribution: A continuous relaxation of discrete random variables. In: 5th International Conference on Learning Representations, ICLR 2017 (2017)
37. Mai, K., Mai, S., Nguyen, A., Linh, N.V., Than, K.: Enabling hierarchical dirichlet processes to work better for short texts at large scale. In: Advances in Knowledge Discovery and Data Mining - 20th Pacific-Asia Conference, PAKDD, *Lecture Notes in Computer Science*, vol. 9652, pp. 431–442. Springer (2016)
38. Masegosa, A., Nielsen, T.D., Langseth, H., Ramos-López, D., Salmerón, A., Madsen, A.L.: Bayesian models of data streams with hierarchical power priors. In: International Conference on Machine Learning, pp. 2334–2343 (2017)
39. Masegosa, A.R., Ramos-López, D., Salmerón, A., Langseth, H., Nielsen, T.D.: Variational inference over nonstationary data streams for exponential family models. Mathematics **8**(11), 1942 (2020)
40. McInerney, J., Ranganath, R., Blei, D.: The population posterior and bayesian modeling on streams. In: Advances in Neural Information Processing Systems, pp. 1153–1161 (2015)
41. Mehrotra, R., Sanner, S., Buntine, W., Xie, L.: 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, pp. 889–892 (2013)
42. Mermillod, M., Bugaiska, A., Bonin, P.: The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects. Frontiers in psychology **4**, 504 (2013)
43. Mianjy, P., Arora, R., Vidal, R.: On the implicit bias of dropout. In: International Conference on Machine Learning, pp. 3537–3545 (2018)
44. Mou, W., Zhou, Y., Gao, J., Wang, L.: Dropout training, data-dependent regularization, and generalization bounds. In: International Conference on Machine Learning, pp. 3645–3653 (2018)
45. Nguyen, C.V., Li, Y., Bui, T.D., Turner, R.E.: Variational continual learning. In: The International Conference on Learning Representations (ICLR) (2018)
46. Nguyen, V., Nguyen, D., Van, L.N., Than, K.: Infinite dropout for training bayesian models from data streams. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 125–134 (2019)
47. Rifai, S., Glorot, X., Bengio, Y., Vincent, P.: Adding noise to the input of a model trained with a regularized objective. arXiv preprint arXiv:1104.3250 (2011)
48. Rogers, S., Girolami, M., Campbell, C., Breitling, R.: The latent process decomposition of cdna microarray data sets. IEEE/ACM Transactions on Computational Biology and Bioinformatics **2**(2), 143–156 (2005)
49. Russell, S.J., Norvig, P.: Artificial intelligence: a modern approach. Malaysia; Pearson Education Limited, (2016)
50. Shaker, A., Hüllermeier, E.: Recovery analysis for adaptive learning from non-stationary data streams: Experimental design and case study. Neurocomputing **150**, 250–264 (2015)
51. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research **15**(1), 1929–1958 (2014)
52. Theis, L., Hoffman, M.D.: A trust-region method for stochastic variational inference with applications to streaming data. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, ICML'15, p. 2503–2511. JMLR.org (2015)
53. Tran, B., Nguyen, A.D., Van, L.N., Than, K.: Dynamic transformation of prior knowledge into bayesian models for data streams. IEEE Transactions on Knowledge and Data Engineering (2021)
54. Tuan, A.P., Bach, T.X., Nguyen, T.H., Linh, N.V., Than, K.: Bag of biterms modeling for short texts. Knowl. Inf. Syst. **62**(10), 4055–4090 (2020)
55. Van, L.N., Tran, B., Than, K.: Graph convolutional topic model for data streams. Neurocomputing **468**, 345–359 (2022). DOI https://doi.org/10.1016/j.neucom.2021.10.047
56. Van Linh, N., Anh, N.K., Than, K., Dang, C.N.: An effective and interpretable method for document classification. Knowledge and Information Systems **50**(3), 763–793 (2017)
57. Van Linh, N., Nguyen, D.A., Nguyen, T.B., Than, K.: Neural poisson factorization. IEEE Access **8**, 106395–106407 (2020)
58. Wager, S., Wang, S., Liang, P.S.: Dropout training as adaptive regularization. In: Advances in Neural Information Processing Systems, pp. 351–359 (2013)
59. Wang, S., Wang, M., Wager, S., Liang, P., Manning, C.D.: Feature noising for log-linear structured prediction. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1170–1179 (2013)
60. Yin, M., Yue, Y., Zhou, M.: ARSM: augment-reinforce-swap-merge estimator for gradient backpropagation through categorical variables. In: K. Chaudhuri, R. Salakhutdinov (eds.) Proceedings of the 36th International Conference on Machine Learning, (ICML), *Proceedings of Machine Learning Research*, vol. 97, pp. 7095–7104. PMLR (2019)
61. Zenke, F., Poole, B., Ganguli, S.: Continual learning through synaptic intelligence. Proceedings of machine learning research **70**, 3987 (2017)
62. Zhai, S., Zhang, Z.M.: Dropout training of matrix factorization and autoencoder for link prediction in sparse graphs. In: Proceedings of the 2015 SIAM International Conference on Data Mining, pp. 451–459 (2015). DOI 10.1137/1.9781611974010.51
63. Zhang, C., Bütepage, J., Kjellström, H., Mandt, S.: Advances in variational inference. IEEE transactions on pattern analysis and machine intelligence **41**(8), 2008–2026 (2018)

A iDropout for LDA and NB

A.1 When LDA is the base model

Learning process: Inference for local variables θ and z can be done as in aiDropout. We only consider the objective function with respect to β_k^t:

$$\begin{aligned} F(\beta_k^t) &= \log p(\beta_k^t|\beta_k^{t-1}) + \sum_{d=1}^{M}\sum_{n=1}^{N_d} \log p(w_{dn}|z_{dn},\tilde{\beta}^t) \\ &= -\frac{1}{2\sigma^2}||\beta_k^t - \beta_k^{t-1}||^2 + \sum_{d=1}^{M}\sum_{n,j=1}^{N_d,V} \phi_{dnk}\mathbb{I}[w_{dn}=j]\log\tilde{\beta}_{kj}^t \\ &= -\frac{1}{2\sigma^2}||\beta_k^t - \beta_{k-1}^t||^2 + \sum_{d=1}^{M}\sum_{n,j=1}^{N_d,V} \phi_{dnk} I[w_{dn}=j]\beta_{kj}^t\pi_{kj}^t \\ &\quad - \sum_{d=1}^{M}\sum_{n,j=1}^{N_d,V} \phi_{dnk} I[w_{dn}=j]\log(\sum_{i=1}^{V}\exp(\beta_{ki}^t\pi_{ki}^t)) \end{aligned} \tag{22}$$

The objective function F is guaranteed to be concave. In deed, $-\frac{1}{2\sigma^2}||\beta_k^t - \beta_k^{t-1}||^2$ and $\beta_{kj}^t\pi_{kj}^t$ are obviously concave with respect to β_k^t, while the log-sum-exp is also a well-known convex function. Therefore, $F(\beta_k^t)$ is concave with respect to β_k^t, and we can find its maximum by applying gradient ascent on F. We sum up the learning algorithm of iDropout for LDA in Algorithm 4.

Algorithm 4 iDropout training for LDA

Input: Data sequence $\{D^1, D^2, ...\}$, variance σ^2, prior α
Output: Global variable β

Initialize β^0 randomly
for t^{th} *mini-batch with data* D^t **do**
  Draw dropout matrix π^t randomly
  for *each document* d *in* D^t **do**
    Infer (γ_d, ϕ_d) as in the original paper [4]
  Find β_k^t by maximizing (22)

A.2 When NB is the base model

The generative process for each mini-batch t is as follows. Firstly, draw the global variable β^t: $\beta_c^t \sim \mathcal{N}(\beta_c^{t-1}, \sigma^2 I)$ and calculate the class matrix: $\tilde{\beta}_{cj}^t = \text{softmax}(\beta_c^t \odot \pi_c^t)_j$. Each document d is drawn by first choosing the class label $c_d \sim Mult(\alpha)$ and then drawing n^{th} word $w_{dn} \sim Mult(\tilde{\beta}_{c_d}^t)$.

Learning process: For each class c, the objective function with respect to β_c^t is:

$$\begin{aligned} F(\beta_c^t) &= \log p(\beta_c^t|\beta_c^{t-1}) + \sum_{d\in D_c^t}\sum_{n=1}^{N_d} \log p(w_{dn}|c_d, \tilde{\beta}^t) \\ &= -\frac{1}{2\sigma^2}||\beta_c^t - \beta_c^{t-1}||_2^2 + \sum_{d\in D_c^t}\sum_{n=1}^{N_d}\sum_{j=1}^{V} \mathbb{I}[w_{dn}=j]\log\tilde{\beta}_{cj}^t \\ &= -\frac{1}{2\sigma^2}||\beta_c^t - \beta_c^{t-1}||_2^2 + \sum_{d\in D_c^t}\sum_{n=1}^{N_d}\sum_{j=1}^{V} \mathbb{I}[w_{dn}=j]\beta_{cj}^t\pi_{cj}^t - N_c\log(\sum_{i\in[V]}\exp(\beta_{ci}^t\pi_{ci}^t)) \end{aligned}$$

where D_c^t includes all documents which belong to class c, N_c is the total number of words in all documents belonging to class c. Learning for NB is very simple. At each mini-batch t, we use gradient ascent to maximize $F(\beta_c^t)$ with respect to β_c^t.

B Evaluation metrics for the unsupervised task

Log Predictive Probability [24]: Predictive Probability measures the predictiveness and generalization of a model on new data. Assume that after learning from training data D_{train}, we obtain the model parameter β. For each document in testing D_{test} with more than or equal to 5 words, we divide randomly into two disjoint parts $\mathbf{w_{obs}}$ and $\mathbf{w_{ho}}$ with a ratio of 80:20. We next do inference for $\mathbf{w_{obs}}$ to estimate θ^{obs}. Then, we approximate the predictive probability $\mathbf{w_{ho}}$ as:

$$\begin{aligned} p(\mathbf{w_{ho}} \mid \mathbf{w_{obs}}, \beta) &= \prod_{w\in\mathbf{w_{ho}}} p(w \mid \mathbf{w_{obs}}, \beta) \\ &\approx \prod_{w\in\mathbf{w_{ho}}} p(w \mid \theta^{obs}, \beta) \\ &= \prod_{w\in\mathbf{w_{ho}}}\sum_{k=1}^{K} p(w \mid z=k, \beta)p(z=k \mid \theta^{obs}) \\ &= \prod_{w\in\mathbf{w_{ho}}}\sum_{k=1}^{K} \theta_k^{obs}\beta_{kw} \end{aligned}$$

Then Log Predictive Probability of each document d is:

$$LPP_d = \frac{\log p(\mathbf{w_{ho}} \mid \mathbf{w_{obs}}, \beta)}{|\mathbf{w_{ho}}|} \tag{23}$$

(with $|\mathbf{w_{ho}}|$ is the length of d in $\mathbf{w_{ho}}$) and on the whole testing D_{test} is:

$$\text{Log Predictive Probability} = \frac{\sum_{d\in D_{test}} LPP_d}{|D_{test}|} \tag{24}$$

Log Predictive Probability was averaged from 5 random splits, each was on 1000 documents.

Normalized Pointwise Mutual Information [5]: NPMI is the measure to help us see the coherence or semantic quality of individual topics. For each topic k, we pick a set $\mathbf{w^k} = \{w_1^k, w_2^k, ..., w_t^k\}$, including t words with the highest probabilities in topic distribution

Table 3: Data streams with incremental and recurring concept drifts on News Aggregator. LDA is trained consecutively on the mini-batches of classes. While class order shows the class sequence that is trained consecutively, final mini-batch index is the final mini-batch of the corresponding class in the mini-batch sequence. Regarding incremental concept drift, at the change point between two classes, we add 5 mixed mini-batches which include data from both of the two classes. In terms of recurring concept drift, the mini-batches of each class are divided into three parts to simulate the recurring concept drift with three iterations.

Incremental	Class order	1	2	3	4	-	-	-	-	-	-	-	-
	Final mini-batch index	54	108	183	208	-	-	-	-	-	-	-	-
Recurring	Class order	1	2	3	4	1	2	3	4	1	2	3	4
	Final mini-batch index	20	40	65	72	92	112	137	144	161	174	199	207

Table 4: The lowest LPP, median LPP and restoration time of the 5 methods when dealing with gradual and recurring concept drift on News Agrregator dataset. For the lowest LPP and median LPP, higher is better. For restoration time, lower is better.

		aiDropout	iDropout	PVB	SVB-PP	SVB
Recurring concept drift	Lowest LPP	-9.16939	-9.21681	-9.44007	-9.53854	-9.50774
	Median LPP	-8.23875	-8.49794	-8.90354	-8.92836	-9.17998
	Restoration time	0.157218	0.242427	0.414582	0.414582	0.359682
Gradual concept drift	Lowest LPP	-8.8894	-8.96813	-9.23823	-9.36473	-9.60927
	Median LPP	-8.20903	-8.3581	-8.77503	-8.838	-9.25923
	Restoration time	0.0048	0.0129	0.404433	0.4106	0.0483

β_k. NPMI of one topic k is computed as follows:

$$\begin{aligned}
\text{NPMI}(k, \mathbf{w^k}) &= \frac{2}{t(t-1)} \sum_{i=2}^{t} \sum_{j=1}^{i-1} \frac{\log \frac{p(w_i^k, w_j^k)}{p(w_i^k)p(w_j^k)}}{-\log p(w_i^k, w_j^k)} \\
&\approx \frac{2}{t(t-1)} \sum_{i=2}^{t} \sum_{j=1}^{i-1} \frac{\log \frac{D(w_i^k, w_j^k)+10^{-2}}{D} - \log \frac{D(w_i^k)D(w_j^k)}{D^2}}{-\log \frac{D(w_i^k, w_j^k)+10^{-2}}{D}} \\
&= \frac{2}{t(t-1)} \sum_{i=2}^{t} \sum_{j=1}^{i-1} -1 + \frac{2\log D - \log D(w_i^k) - \log D(w_j^k)}{\log D - \log(D(w_i^k, w_j^k) + 10^{-2})}
\end{aligned}$$

where D is the total number of documents, $D(w_i^k)$ is the number of docs containing w_i^k, $D(w_i^k, w_j^k)$ is the number of docs containing pair (w_i^k, w_j^k).

Overall, NPMI of a model with all K topics is:

$$NPMI = \frac{1}{K} \sum_{k=1}^{K} NPMI(k, t) \tag{25}$$

In the experiments, we choose $t = 20$ for each topic.

C Evaluation on incremental and recurring concept drifts

We conduct more experiments on News Aggregator dataset to investigate how our methods (iDropout and aiDropout) and the baselines deal with incremental and recurring concept drifts.

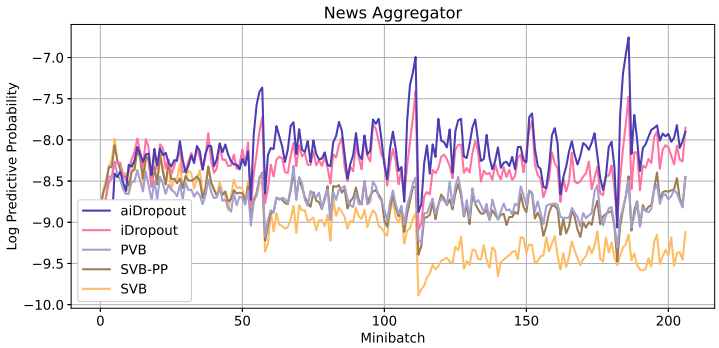


Fig. 9: Performance of the methods when facing with incremental concept drift on News Aggregator dataset. LDA is the base model. Higher is better.

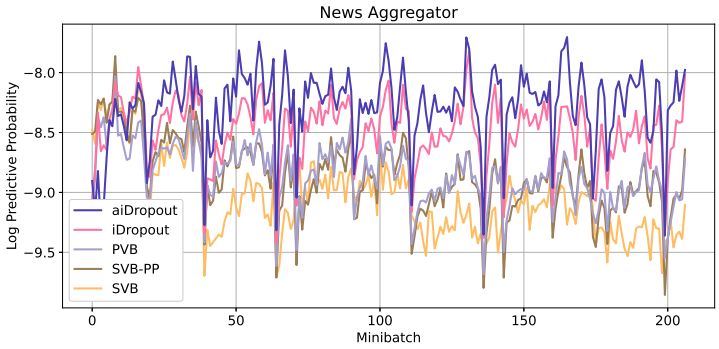


Fig. 10: Performance of the methods when facing with recurring concept drift on News Aggregator dataset. LDA is the base model. Higher is better.

We use LDA with $K = 100$ and $\alpha = 0.01$ as our base model for these experiments. Each dataset is split into mini-batches, each mini-batch contains 2000 documents. Regarding incremental concept drift, we randomly shuffle data instances in each class and then divide them into mini-batches. We train LDA on the mini-batches of consecutive classes. At the change point between two classes, we add 5 mixed mini-batches which include data from both of the two classes. In terms of recurring concept drift, the mini-batches of each class are divided into three parts to simulate the recurring concept drift with three iterations. Table 3 shows mini-batch sequence in incremental and recurring concept drifts. After finishing training each mini-batch, we measure LPP on the next mini-batch to evaluate how the methods adapt to new concept drift.

Fig. 9 and Fig. 10 illustrate the performance of the methods when dealing with the incremental and recurring concept drifts respectively. Overall, both iDropout and aiDropout achieve better performance than the baselines. Although their LPPs decrease dramatically when new concept drift happens, they adapt quickly to the new concept after a few next mini-batches. It is obvious that the decrease in the performances of all methods in the incremental concept drift is less than this in both the sudden and recurring concept drifts. In particular, aiDropout outperforms significantly iDropout in the recurring concept drift where several new concepts occur. This phenomenon can be because the adaptive mechanism helps learn dropout rate automatically to adapt to new data effectively. Moreover, table 4 shows the lowest LPP, median LPP and restoration time of the 5 methods when dealing with gradual and recurring concept

drift on News Agrregator dataset. Overall, the behaviours of the 5 methods on recurring and gradual concept drifts are similar to them on sudden one. Both aiDropout and iDropout drop LPP least when a new concept happens, then they restore most quickly to achieve better performance than the remaining methods.

D Discussion on Computational and Memory complexity

This section discusses the computational and memory complexity of iDropout and aiDropout. We make an intuitive comparison with the baselines (SVB, PVB, and SVB-PP) albeit it is difficult to theoretically analyze the computational and memory complexity of the methods for a general model. Then, we make an empirical comparison.

We consider a general model $B(\beta, z, x)$ (as in section 3) where β is the global variable, $x = x_{1:M}$ is the set of observations, and $z = z_{1:M}$ is the set of hidden variables of observations. All methods often use variational inference for inferring local variable z. However, while SVB, PVB, SVB-PP aim to full Bayesian approximation for global variable β based on variational inference, iDropout and aiDropout use the maximum a posterior (MAP) to learn a point estimate. The model is consecutively trained on collected mini-batches in a general learning scenario in a streaming environment. Let η^t and ϕ^t be variational parameters of β and z at mini-batch t respectively, γ be the hyperparameter of prior distribution $p(\beta^t|\gamma)$, and ρ be a forgetting factor. When learning the model on a mini-batch D^t, the objective function of each method is re-written as bellow:

For SVB[12]:

$$\{\hat{\eta}^t, \hat{\phi}^t\} = \arg\max_{\eta^t, \phi^t} \left\{ E_{q(z^t|\phi^t)} \left[\log \frac{p(D^t, z^t|\beta^t)}{q(z^t|\phi^t)} \right] - KL\left[q(\beta^t|\eta^t)||q(\beta^t|\eta^{t-1})\right] \right\} \tag{26}$$

For PVB:

$$\{\hat{\eta}^t, \hat{\phi}^t\} = \arg\max_{\eta^t, \phi^t} \left\{ E_{q(z^t|\phi^t)} \left[\log \frac{p(D^t, z^t|\beta^t)}{q(z^t|\phi^t)} \right] - KL\left[q(\beta^t|\eta^t)||p(\beta^t|\gamma)\right] \right\} \tag{27}$$

For SVB-PP:

$$\{\hat{\eta}^t, \hat{\phi}^t\} = \arg\max_{\eta^t, \phi^t} \left\{ E_{q(z^t|\phi^t)} \left[\log \frac{p(D^t, z^t|\beta^t)}{q(z^t|\phi^t)} \right] - KL\left[q(\beta^t|\eta^t)||q(\beta^t|\rho\eta^{t-1} + (1-\rho)\gamma)\right] \right\} \tag{28}$$

For iDropout:

$$\{\hat{\beta}^t, \hat{\phi}^t\} = \arg\max_{\beta^t, \phi^t} \left\{ \log p(\beta^t|\beta^{t-1}) + E_{q(z^t|\phi^t)} \left[\log \frac{p(D^t, z^t|\tilde{\beta}^t)}{q(z^t|\phi^t)} \right] \right\} \tag{29}$$

where $\tilde{\beta}^t = f(\beta^t \odot \pi^t)$ and π^t is sampled from Bernoulli distribution with hyperparameter p.

For aiDropout:

$$\begin{aligned}\{\hat{\beta}^t, \hat{\phi}^t, \hat{\lambda}^t, \hat{p}^t\} = \arg\max_{\beta^t, \phi^t, \lambda^t, p^t} \Bigg\{ & \log p(\beta^t|\beta^{t-1}) + E_{q(z^t|\phi^t)} \left[\log \frac{p(D^t, z^t|\tilde{\beta}^t)}{q(z^t|\phi^t)} \right] \\ & - KL\left[q(\pi^t|\lambda^t)||p(\pi^t|p^t)\right] \Bigg\}\end{aligned} \tag{30}$$

where $\tilde{\beta}^t = f(\beta^t \odot \pi^t)$ and π^t is sampled from variational distribution $q(\pi^t|\lambda^t)$.

We emphasize that the five methods have the same objective function w.r.t the variational parameter ϕ^t of local variable z, the parameters related to the global variable make them different. We will focus on discussing their computational and memory complexity on global variables.

[12] The objective function can be seen in some studies [52, 45]

Table 5: The average training time (second) of minibatches when using the five methods to learn LDA and NB

Method	LDA		NB	
	News Aggregator	The Irish Times	News Aggregator	The Irish Times
aiDropout	185	227	3.3103	3.8347
iDropout	155	161	1.0774	1.1521
PVB	24	26	0.0424	0.0334
SVB-PP	20.6	23	0.0405	0.0325
SVB	20.5	23.2	0.0413	0.0333

Regarding memory complexity, compared to the three baselines (SVB, PVB, and SVB-PP), both aiDropout and iDropout must use dropout matrix π^t and sample it in each iteration. Note that π^t has the same size as the global variable β. Moreover, because aiDropout approximates the true posterior of π^t by a variational distribution $q(\pi^t|\lambda^t)$ in order to create an adaptive droprate mechanism, it must store variational parameter λ^t. In our work, the size of λ^t is set the same as π^t. Therefore, iDropout and aiDropout must store more parameters than the baselines by once and twice the size of β^t, respectively.

In terms of computational complexity, both aiDropout and iDropout are often more complex than the baselines. They use gradient-based optimizers that often require a large number of iterations to learn global parameters. Moreover, due to optimizing the dropout rate, aiDropout requires more computation than iDropout to do an iteration. Meanwhile, SVB, PVB, and SVB-PP achieve closed-form solutions when they apply for conjugate models (as in our case studies); therefore, they often run considerably faster than both iDropout and aiDropout in these cases. When working with non-conjugate models, iDropout can compare to the baselines. However, it is difficult to make a clear comparison because they learn different models when both iDropout and aiDropout change the prior distribution of β compared to the baselines.

From the discussion above, it is evident that aiDropout trades off the quality of the learned model against computational and memory complexity. However, we consider a general learning scenario in a streaming environment where data is often collected in mini-batches based on the fixed mini-batch size or timestamp; then a model is trained on each mini-batch. Therefore, the velocity and volume of arriving mini-batches directly make the computational and memory complexity requirements for learning methods. iDropout and aiDropout are effective solutions when working on a streaming environment in which they can respond to time and memory requirements well.

To show the trade-off in aiDropout more obviously, we measure mini-batch's average training time when using the five methods to learn LDA as in subsection 6.3.1 (Fig.7) and NB as in 6.2.2 (Fig.5). We use our server (64 cores CPU, 128G DDR) to train LDA and Google colab with free GPU Tesla K80 to train NB. Table 5 shows the results. aiDropout and iDropout are noticeably slower than the baselines in both LDA and NB. Because LDA and NB are conjugate models, the baselines obtain closed-form solutions. Meanwhile, both aiDropout and iDropout require large iterations (100 in our experiments) when conducting gradient-based algorithms.

7. A list of references

	Recommender 1	**Recommender 2**
Prefix	Prof.	Dr.
First Name	Nhat	Hung
Last Name	Ho	Bui
Name of Institution	The University of Texas at Austin	VinAI Research
Title	Assistant Professor	Doctor, Director at VinAI Research
Relationship	Mentor at VinAI Research	Advisor at VinAI Research
Email	minhnhat@utexas.edu	v.hungbh1@vinai.io

	Recommender 3	**Recommender 4**
Prefix	Prof.	Prof.
First Name	Khoat	Dinh
Last Name	Than	Phung
Name of Institution	Hanoi University of Science and Technology, VinAI Research	Monash University, VinAI Research
Title	Associate Professor	Professor, Research Director at VinAI Research
Relationship	Thesis supervisor at university	Research Director at VinAI Research
Email	khoattq@soict.hust.edu.vn	dinh.phung@monash.edu