Variational Beam Search for Novelty Detection

Aodong Li AODONGL1@UCI.EDU
Alex Boyd ALEXJB@UCI.EDU
Padhraic Smyth SMYTH@ICS.UCI.EDU
Stephan Mandt MANDT@UCI.EDU
University of California, Irvine

Abstract

We consider the problem of online learning in the presence of sudden distribution shifts, which may be hard to detect and can lead to a slow but steady degradation in model performance. To address this problem we propose a new Bayesian meta-algorithm that can both (i) make inferences about subtle distribution shifts based on minimal sequential observations and (ii) accordingly adapt a model in an online fashion. The approach uses beam search over multiple change point hypotheses to perform inference on a hierarchical sequential latent variable modeling framework. Our proposed approach is model-agnostic, applicable to both supervised and unsupervised learning, and yields significant improvements over state-of-the-art Bayesian online learning approaches.

1. Introduction

Deployed machine learning systems are often faced with the problem of distribution shift, where the new data that the model processes is systematically different from the data the system was trained on. Furthermore, the shift can happen anytime after deployment, unbeknownst to the users, with dramatic consequences for systems such as self-driving cars, robots, trading algorithms, among many other examples.

Updating a deployed model on new, representative data can help mitigate these issues, as well as improve on general performance in most cases. This task is commonly referred to as *online learning*. A particular variant of online learning that focuses on adapting models to new or novel data (in either features and/or outputs) over time is known as *continual learning*. Approaches developed on this task typically focus on mitigating the degradation of performance over earlier data, often referred to as *catastrophic forgetting*. For instance, variational continual learning (VCL) (Nguyen et al., 2017) employs a Bayesian online learning framework to prevent forgetting by modeling the model's prior distribution for each new batch of data (referred to as a "task" in the continual learning literature) as the posterior from the previous one.

VCL and other Bayesian solutions are typically robust to catastrophic forgetting; however, they may suffer from an opposite problem that we are defining as *catastrophic remembering*. In continually training a Bayesian model, the posterior distribution becomes progressively more and more confident as more data is used to inform it. Given enough data, the model's prior distribution (i.e. previous posterior distribution) will be too confident to adequately adapt to new data affected by a distributional shift (for brevity, we will refer to this shift as a *novelty*).

In this paper, we propose a more robust approach that copes with the problems of both catastrophic forgetting and remembering. The idea is that the model not only requires a good mechanism to aggregate data, but also is able to partially forget information that has become obsolete. To achieve this, we still use the Bayesian online learning framework; however, before combining the previously learned posterior with new data evidence, we introduce an intermediate step. This step allows the model to decide between two actions to take: reduce the previous posterior's confidence to provide more "room" for novel information, or remain in the same state (i.e. retain the unchanged, previous posterior as the new prior). We propose a mechanism for enabling this decision using a "spike and slab" novelty prior (described in Section 2.1). We further augment this decision process by introducing *variational beam search*, a new inference scheme that allows the model to consider multiple different hypothetical sequences of detected distributional shifts (or lack thereof).

We present experiments for Bayesian deep learning experiments using real datasets with artificially introduced shifts over time, as well as unsupervised experiments for analyzing semantic change over time in language (Section C.3 in supplement). Our approach both leads to more semantic and compact word embeddings, as well as significantly improves performance in the supervised tasks.

2. Methods

We consider a stream of data that arrives in batches ("tasks") $\mathbf{x}_t$ at discrete times t. For notational simplicity we focus on the unsupervised case, where the task is to model $p(\mathbf{x}_t)$ using a model $p(\mathbf{x}_t|\mathbf{z}_t)$ with parameters $\mathbf{z}_t$ that we would like to optimally tune to each new batch[1].

We furthermore assume that while the $\mathbf{x}_t$ are i.i.d. within batches, they are not i.i.d. across batches, but rather come from a time-varying distribution $p_t(\mathbf{x}_t)$ (or $p_t(\mathbf{x}_t, \mathbf{y}_t)$ in the supervised cases) which is subject to distribution shifts. We assume that these distribution shifts occur instantaneously (as opposed to gradually) and at unknown times, i.e., a change may (or may not) occur with each batch t and (if it occurs) will persist until the next change. The challenge is to optimally adapt the parameters $\mathbf{z}_t$ to each new task while borrowing statistical strength from previous tasks.

Variational Continual Learning The basis of our approach is the insight that for sequential data, one can use a Bayesian model's posterior at time $t-1$ as a prior for the next task at time t. Since typically the posterior is not available in closed-form, we must use approximate inference. It is natural to use a variational posterior $q_{t-1}(\mathbf{z}_t)$. This leads to a sequence of variational inference tasks (Zhang et al., 2018) known as variational continual learning (VCL) (Nguyen et al., 2017):

$$q_t(\mathbf{z}_t) = \underset{q(\mathbf{z}_t)\in Q}{\arg\max}\ \mathbb{E}_q[\log p(\mathbf{x}_t|\mathbf{z}_t)] - \mathrm{KL}(q(\mathbf{z}_t)||q_{t-1}(\mathbf{z}_t)), \tag{1}$$

where Q is the family of potential approximate posterior distributions (i.e. normal distributions).

Eq. 1 is known as the evidence lower bound (ELBO) in variationa inference (Jordan et al., 1999; Zhang et al., 2018); for every new task it is optimized until convergence. Note that in its original formulation, the goal of VCL is to train a model that performs well cumulatively on *all* the learning tasks previously encountered (Nguyen et al., 2017), as opposed to just the most recent task, as studied in this paper.

Catastrophic Remembering While continual learning mainly addresses catastrophic forgetting, where a model loses in performance on the tasks previously encountered, we address the opposite effect: catastrophic remembering (French, 1999). Here, a model becomes overconfident with time and loses its ability to adapt to distribution shifts. In Bayesian online learning, such catastrophic remembering is caused by an overconfident posterior. To explain this effect, we note that a posterior's variance shrinks as it encounters more data. Assume that after a long sequence of updates, the

1. In supervised setups, we consider a conditional model $p(\mathbf{y}_t|\mathbf{z}_t, \mathbf{x}_t)$ with features $\mathbf{x}_t$ and targets $\mathbf{y}_t$.

posterior $q_{t-1}(\mathbf{z}_t)$ can be well-approximated by a point mass $\delta(\mathbf{z}_t - \mathbf{z}_0)$ centered around some parameter $\mathbf{z}_0$. In this case, any new data evidence $p(\mathbf{x}_t|\mathbf{z}_t)$ will have a diminishing effect on the next posterior $q_t(\mathbf{z}_t)$ as $q_t(\mathbf{z}_t) \propto p(\mathbf{x}_t|\mathbf{z}_t)\delta(\mathbf{z}_t - \mathbf{z}_0) \approx q_{t-1}(\mathbf{z}_t)$, in other words, the strong prior over-rules the new data even though the training data that lead to it may have become obsolete due to a distribution shift. To prevent catastrophic remembering, we adapt Eq. 1 to the online learning scenario, specifically assuming irregular and instantaneous distribution shifts.

2.1. Posterior Broadening Mechanisms

In order to combat catastrophic remembering, the posterior in Eq. 1 needs to be broadened before it is combined with new data evidence. This broadening mechanism erases learned information to free-up model capacity to adjust to the new data distribution.

Among several possible options, we consider *relative broadening*, which amounts to tempering the prior by a fixed amount, resulting in $p_\beta(\mathbf{z}_t) \propto q_{t-1}(\mathbf{z}_t)^\beta$ for $0 < \beta \leq 1$. For a Gaussian q with diagonal variances σ_i^2 in dimension $\mathbf{z}_i$, relative broadening removes an equal amount of information in each dimension, $H_i = \frac{1}{2}\log(2\pi e \sigma_i^2/\beta^2) = \frac{1}{2}\log(2\pi e\sigma_i^2) - \log\beta$. Since tempering broadens the posterior non-locally, this scheme does not possess a continuous latent time series interpretation [2].

Conditional Broadening If novelties happened at a predictable and constant rate, we were done: tuning the parameter β to the expected rate of change (with a large β for a high change rate and $\beta = 1$ for no expected change). However, in reality, novelties can be of varying strength, irregular, and *unobserved*. We therefore propose to model the novelty at time t with a binary latent variable s_t, with $s_t = 0$ for no change occurring, and $s_t = 1$ indicating a shift. For $s_t = 1$, we broaden the posterior and use it as a prior. If no change occurs, we just use the previous posterior as the new prior and proceed. This gives rise the the following conditional prior:

$$p_\beta(\mathbf{z}_t|s_t) = \begin{cases} q_{t-1}(\mathbf{z}_t) & \text{for } s_t = 0 \\ q_{t-1}^\beta(\mathbf{z}_t) & \text{for } s_t = 1 \end{cases}. \tag{2}$$

We defined the tempered approximate posterior at time $t-1$ as $q_{t-1}^\beta(\mathbf{z}_t) = \frac{(q_{t-1}(\mathbf{z}_t))^\beta}{\int (q_{t-1}(\mathbf{z}_t))^\beta d\mathbf{z}_t}$. (Note that q^β has a closed-form expression for a Gaussian q.) The conditional broadening approach leads to the following online inference scheme:

$$\begin{aligned} q_t(\mathbf{z}_t|s_t) &= \underset{q(\mathbf{z}_t|s_t)\in Q}{\arg\max}\ \mathcal{L}(q|s_t), \\ \mathcal{L}(q|s_t) &:= \mathbb{E}_q[\log p(\mathbf{x}_t|\mathbf{z}_t)] - \mathrm{KL}(q(\mathbf{z}_t|s_t)||p_\beta(\mathbf{z}_t|s_t)). \end{aligned} \tag{3}$$

This involves a joint variational distribution $q(\mathbf{z}_t|s_t)$ over the latents $\mathbf{z}_t$ and change variable s_t. We place a Bernoulli prior $p(s_t)$ on the change variables. As described below, we jointly infer $\mathbf{z}_t$ and the change variables s_t from data.

2.2. Bayesian Reasoning over Distribution Shifts

Structured Variational Inference According to our assumptions, the novelties occur at discrete times and are unobserved. Therefore, both $\mathbf{z}_t$ and s_t have to be inferred from data. We first define the

2. This means that it is impossible to specify a conditional distribution $p(\mathbf{z}_t|\mathbf{z}_{t-1})$ that corresponds to relative broadening.

marginal likelihood over data given the change variable s_t to be $p_\beta(\mathbf{x}_t|s_t) := \int p(\mathbf{x}_t|\mathbf{z}_t)p_\beta(\mathbf{z}_t|s_t)d\mathbf{z}_t$, which is often intractable. This intractability leaves the exact posterior over s_t, $p_\beta(s_t|\mathbf{x}_t)$ not available as well (by Bayes rule). However, we can follow a structured variational inference approach (Wainwright and Jordan, 2008; Hoffman and Blei, 2015; Zhang et al., 2018), defining a joint variational distribution $q(\mathbf{z}_t, s_t) = q(s_t)q(\mathbf{z}_t|s_t)$.

By definition, the conditional distributions $q(\mathbf{z}_t|s_t)$ are solutions to the optimization problem in Eq. 3. Absorbing the optimized conditional ELBO in Eq. 3, $q(s_t)$ has a closed-form solution:

$$q^*(s_t) = \mathrm{Bern}(s_t; m); \quad m = \sigma\left(\mathcal{L}(q|s_t = 1) - \mathcal{L}(q|s_t = 0) + \xi_0\right), \tag{4}$$

where σ is the sigmoid function and $\xi_0 = \log p(s_t = 1) - \log p(s_t = 0)$ are the log-odds of the prior $p(s_t)$. This specifies the posterior over s_t. We provide the derivation details, the resemblance to the exact inference, and the interpretation as a likelihood ratio test in the Supplement.

Finally, we obtain the marginal distribution over latent variables $q(\mathbf{z}_t)$ at time t as a binary mixture with mixture weights $q(s_t = 0) = m$ and $q(s_t = 1) = 1 - m$:

$$q_t(\mathbf{z}_t) = m\, q(\mathbf{z}_t|s_t = 0) + (1 - m)q(\mathbf{z}_t|s_t = 1). \tag{5}$$

Exponential Branching We note that while we had originally started with a Gaussian variational posterior $q_{t-1}(\mathbf{z}_t)$ at the previous time, our inference scheme resulted in $q_t(\mathbf{z}_t)$ being a mixture of two Gaussians:[3] the inference scheme branches over two alternative hypotheses. When we iterate, we encounter an exponential branching of possibilities, or *hypotheses* over possible sequences of regime shifts. To still be able to carry out the filtering scheme efficiently, we need a truncation scheme, e.g., approximate the bimodal marginal distribution by a unimodal one. The next section will discuss several methods to achieve this goal.

2.3. Online Inference and Variational Beam Search

While the previous subsections have focused on a single update, we now investigate the possibility of doing multiple updates in a row. This amounts to working out a truncation scheme to restrict the variational posterior to be a unimodal Gaussian (*Variational Greedy Search* (VGS)) or, a mixture of fixed size (*Variational Beam Search* (VBS)) at every time step.

The simplest VGS trains the model in an online fashion by iterating over time steps t. For each t, it explores all possible branches and then truncates the branches. In general, VGS first optimizes the conditional ELBO (Eq. 3) for both $s_t = 0$ and $s_t = 1$ (which corresponds to branches). With the optimized conditional ELBO, it computes the binary mixture weights using Eq. 4. Finally VGS projects $q_t(\mathbf{z}_t)$ (Eq. 5) on the more probable component. Detailed steps are in the Supplement.

Note that VGS updates $\mathbf{z}_t$ at every time step t, resulting in continuously changing variational parameters μ_t and σ_t. The fact that μ_t and σ_t change even between two detected change points makes the output of the algorithm poorly interpretable. One can get a better fit if one outputs the variational parameters μ_t and σ_t at the end of a segment of constant $\mathbf{z}_t$, just before a detected change point $s_t = 1$. We call this a "shy" variant of VGS. More details are in supplement.

VGS and its shy variant make *greedy* decisions on s_t, in the sense that they ignore the subsequent tasks. A greedy search is prone to missing change points in data sets with a low signal/noise ratio per time step because it cannot accumulate evidence for a change point over a series of time steps.

3. See also Fig. 4 of the Supplementary Material.

The most obvious improvement over greedy search that has the ability to accumulate evidence for a change point is beam search. Rather than deciding greedily whether a change occurred or not at each time step, beam search considers both cases in parallel, and it delays the decision as to which one is more likely. Empirically, we find that the naive beam search procedure does not reveal its full potential. As commonly encountered in beam search, histories over change points are largely shared among all members of the beam. We thus developed a simple beam diversification scheme to encourage diverse beams. We found this beam diversification scheme to work robustly across a variety of experiments. More details about diversified VBS are described in the supplement.

3. Experiments

Overview The objective of our experiments is to show that compared to other methods, variational beam search (VBS) (1) better reacts to novelties in supervised setups, while (2) revealing interpretable and temporally sparse latent structure in unsupervised setups. To this end, we experiment on artificial data (Section 3.1), study Bayesian deep learning approaches on sequences of transformed CIFAR and SVHN images (3.2), and study the dynamics of word embeddings on historical text corpora (Section C.3 in supplement). Additional details are in the Supplement.

MODELS	CIFAR-10	SVHN
VBS (K=6) (PROPOSED)	69.7 ± 0.7	89.8 ± 0.4
VBS (K=3) (PROPOSED)	69.1 ± 0.8	89.4 ± 0.5
VGS (PROPOSED)	68.2 ± 0.8	88.9 ± 0.5
VCL [NGUYEN ET AL., 2017]	66.7 ± 0.8	88.7 ± 0.5
LP [SMOLA ET AL., 2003]	62.6 ± 1.0	82.8 ± 0.9
INDEPENDENT TASK	63.7 ± 0.5	85.5 ± 0.7

Table 1: Average Test Acc.

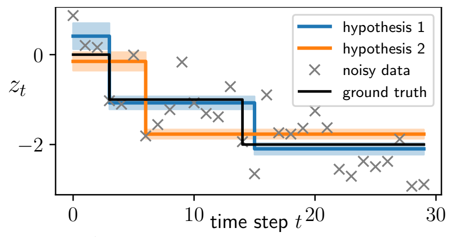

Figure 1: VBS on Toy Data

3.1. Toy Experiments

To test VBS in a very simple setup, we simulated noisy data points centered around a piecewise-constant step function with two steps (Fig. 1). The task is to infer the mean (black line) of a time-varying data distribution (black samples). We evaluated VBS with beam sizes 1 and 2 in terms of their ability to correctly identify the latent jumps in hindsight. Both algorithms start with similar performance initially. However, VBS with beam size 1 ("greedy", orange) fails to recognize the two-fold jump correctly. In contrast, beam size 2 operates with two hypotheses over function levels. Around step 20, the initially unlikely hypothesis with two jumps becomes the dominant one.

3.2. Supervised Experiments

Next, we considered a supervised learning setup, in which an algorithm is exposed to a sequence classification tasks, consisting of batches of CIFAR-10 and SVHN images. To introduce novelties, every few tasks we transform all images globally by combining rotations, shifts, and scaling transformations. To make the approach compatible with our framework, we used a Bayesian convolutional neural network and applied variational beam search to the network's weights. We focused on the latest task performance and evaluated the classification accuracy on a test set subject to the same transformations.

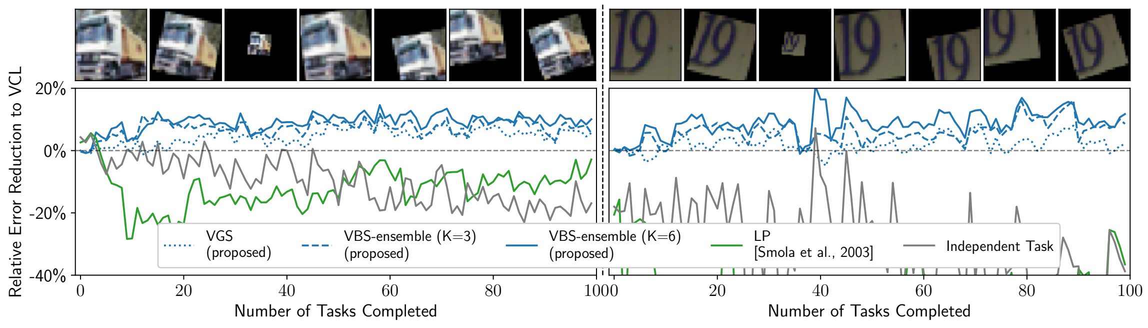

Figure 2: Test performance of our proposed VBS and VGS algorithms compared to various baselines (see main text) on transformed CIFAR-10 (left) and SVHN (right). The top panel shows example transformations of both data.

Datasets We used two standard datasets for image classification: CIFAR-10 (Krizhevsky et al., 2009) and SVHN (Netzer et al., 2011), adopting the original training and test set splits. We further split the training set into batches for online learning, each batch consisting of a third of the full data. Each transformation (either rotation, translation, or scaling) is generated from a fixed, predefined distribution (see Supplement C.2). Changes are introduced every three tasks, where the total number of tasks was 100. Fig 2 (top panel) shows typical resulting transformations.

Baselines. In our supervised experiments, we compared VBS against established Bayesian online learning baselines and an independent batch learning baseline. In addition to Variational Continual Learning (VCL, see Section 2), we also compared against Laplace Propagation (LP) (Smola et al., 2003). We refer to (Nguyen et al., 2017) for technical descriptions of LP. Finally, we also adopt a trivial baseline: learning independent classifiers on each task. Here, we adopt a non-Bayesian neural network with the same architecture. See Supplement for more details.

Architectures and Protocol. All Bayesian methods use the same neural network architecture. We used a truncated version of the VGG convolutional neural network (Supplementary Material C.2) on both datasets and confirmed that our architecture achieved similar performance on CIFAR10 compared to the results reported by Zenke et al. (2017) and Lopez-Paz and Ranzato (2017) in a similar setting. We stress that lower accuracies are obtained in our online learning setting due to the distribution shifts. We initialize each algorithm by training the model on the full, untransformed dataset. We set the relative broadening $\beta = 2/3$ for all supervised experiments. During every new task, all algorithms are trained until convergence (see Supplement for details).

Results The bottom panel of Fig. 2 shows our main results on CIFAR-10 (left) and SVHN (right). To account for varying task difficulties, we show the percentage of the relative error reduction relative to our main baseline, VCL.

VBS with a large beam size of $K = 6$ performs best, followed by VBS with $K = 3$. Variational greedy search (VGS), which corresponds to a beam size $K = 1$, performed comparably with and slightly better than VCL. The reason is that, empirically, the greedy version of our algorithm only detected a small fraction of distribution shifts. This makes its performance sometimes similar to VCL. This stresses the importance of beam search: for larger K, multiple changes were detected.

Table 1 shows the absolute performances of all considered methods, averaged across all of the 100 tasks for the two datasets. Our proposed methods improved significantly over the best-performing baseline VCL by 1.1 percentage points on SVHN and by 3 percentage points on CIFAR-10. The effect of beam search is also evident, with larger beam sizes consistently performing better.

References

Robert Bamler and Stephan Mandt. Dynamic word embeddings. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 380–389. JMLR. org, 2017.

Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020. ISSN 2078-2489. doi: 10.3390/info11020125. URL https://www.mdpi.com/2078-2489/11/2/125.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3 (4):128–135, 1999.

Matthew Gentzkow, JM Shapiro, and Matt Taddy. Congressional record for the 43rd-114th congresses: Parsed speeches and phrase counts. In *URL: https://data. stanford. edu/congress text*, 2018.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

Matthew D Hoffman and David M Blei. Structured stochastic variational inference. In *Artificial Intelligence and Statistics*, 2015.

Slava Jankin Mikhaylov, Alexander Baturo, and Niheer Dasandi. United Nations General Debate Corpus, 2017. URL https://doi.org/10.7910/DVN/0TJX8Y.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

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.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6467–6476, 2017.

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

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

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

Alexander J Smola, Vishy Vishwanathan, and Eleazar Eskin. Laplace propagation. In *NIPS*, pages 441–448, 2003.

Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008.

Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. *arXiv preprint arXiv:1803.04386*, 2018.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *Proceedings of machine learning research*, 70:3987, 2017.

Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

Appendix A. Structured Variational Inference

According to the main paper, we consider the generative model $p_\beta(\mathbf{x}_t, \mathbf{z}_t, s_t) = p(s_t)p_\beta(\mathbf{z}_t|s_t)p(\mathbf{x}_t|\mathbf{z}_t)$ at time step t. Upon observing the data $\mathbf{x}_t$, both $\mathbf{z}_t$ and s_t are inferred. However, exact inference is not available due to the intractability of the marginal likelihood $p_\beta(\mathbf{x}_t|s_t)$. To tackle this, we utilize structured variational inference for both the latent variables $\mathbf{z}_t$ and the Bernoulli change variable s_t. Towards this end, we define the joint variational distribution $q(\mathbf{z}_t, s_t) = q(s_t)q(\mathbf{z}_t|s_t)$. Then the updating procedure for $q(s_t)$ and $q(\mathbf{z}_t|s_t)$ is obtained by maximizing the ELBO $\mathcal{L}(q)$:

$$\begin{aligned} q_t(\mathbf{z}_t, s_t) &= \underset{q(\mathbf{z}_t, s_t)\in Q}{\arg\max}\ \mathcal{L}(q), \\ \mathcal{L}(q) &:= \mathbb{E}_q[\log p_\beta(\mathbf{x}_t, \mathbf{z}_t, s_t) - \log q(\mathbf{z}_t, s_t)]. \end{aligned}$$

Given the generative models, we can further expand $\mathcal{L}(q)$ to simplify the optimization:

$$\begin{aligned} \mathcal{L}(q) &= \mathbb{E}_{q(s_t)q(\mathbf{z}_t|s_t)}[\log p(s_t) + \log p_\beta(\mathbf{z}_t|s_t) + \log p(\mathbf{x}_t|\mathbf{z}_t) - \log q(s_t) - \log q(\mathbf{z}_t|s_t)] \\ &= \mathbb{E}_{q(s_t)}[\log p(s_t) - \log q(s_t) + \mathbb{E}_{q(\mathbf{z}_t|s_t)}[\log p_\beta(\mathbf{z}_t|s_t) + \log p(\mathbf{x}_t|\mathbf{z}_t) - \log q(\mathbf{z}_t|s_t)]] \\ &= \mathbb{E}_{q(s_t)}[\log p(s_t) - \log q(s_t) + \mathbb{E}_{q(\mathbf{z}_t|s_t)}[\log p(\mathbf{x}_t|\mathbf{z}_t)] - \mathrm{KL}(q(\mathbf{z}_t|s_t)||p_\beta(\mathbf{z}_t|s_t))] \\ &= \mathbb{E}_{q(s_t)}[\log p(s_t) - \log q(s_t) + \mathcal{L}(q|s_t)] \end{aligned} \tag{6}$$

where the second step pushes inside the expectation with respect to $q(\mathbf{z}_t|s_t)$, the third step re-orders the terms, and the final step utilizes the definition of conditional ELBO (Eq. 3 in the main paper).

It therefore implies a two-step optimization to maximize $\mathcal{L}(q)$: first maximize the conditional ELBO $\mathcal{L}(q|s_t)$ to find the optimal $q_t(\mathbf{z}_t|s_t = 1)$ and $q_t(\mathbf{z}_t|s_t = 0)$, respectively, then compute the Bernoulli distribution $q^*(s_t)$ by maximizing $\mathcal{L}(q)$ in which the conditional ELBOs $\mathcal{L}(q_t|s_t)$ are fixed.

While $q_t(\mathbf{z}_t|s_t)$ typically needs to be inferred by black box variational inference (Ranganath et al., 2014; Kingma and Welling, 2013; Zhang et al., 2018), the optimal $q^*(s_t)$ has a closed-form solution and bears resemblance to the exact inference counterpart (Eq. 7 in the main paper). To see this, we assume $\mathcal{L}(q_t|s_t)$ are present and $q(s_t)$ is parameterized by $m \in \mathbb{R}$ (for the Bernoulli distribution). Rewriting Eq. 6 gives

$$\begin{aligned}\mathcal{L}(q) = {} & m(\log p(s_t = 1) - \log m + \mathcal{L}(q_t|s_t = 1)) \\ & + (1 - m)(\log p(s_t = 0) - \log(1 - m) + \mathcal{L}(q_t|s_t = 0))\end{aligned}$$

which is concave due to the second derivative is negative. Thus taking the derivative and setting it to zero leads to the optimal solution of

$$\begin{aligned}\log \frac{m}{1-m} &= \log p(s_t = 1) - \log p(s_t = 0) + \mathcal{L}(q_t|s_t = 1)) - \mathcal{L}(q_t|s_t = 0)), \\ m &= \sigma(\mathcal{L}(q_t|s_t = 1)) - \mathcal{L}(q_t|s_t = 0)) + \xi_0),\end{aligned}$$

which attains the closed-form solution as stated in Eq. 4 in the main paper.

Exact Inference To see the similarities to the exact inference, recall $p_\beta(\mathbf{x}_t|s_t) = \int p(\mathbf{x}_t|\mathbf{z}_t)p_\beta(\mathbf{z}_t|s_t)d\mathbf{z}_t$. The exact posterior over s_t is again a Bernoulli $p_\beta(\mathbf{s}|\mathbf{x}_t) = \text{Bern}(s_t; m)$ with parameter

$$m = \sigma\left(\log \frac{p_\beta(\mathbf{x}_t|s_t = 1)p(s_t = 1)}{p_\beta(\mathbf{x}_t|s_t = 0)p(s_t = 0)}\right) \overset{Eq.\ 2}{=} \sigma\left(\log \frac{\int p(\mathbf{x}_t|\mathbf{z}_t)q^\beta_{t-1}(\mathbf{z}_t)d\mathbf{z}_t}{\int p(\mathbf{x}_t|\mathbf{z}_t)q_{t-1}(\mathbf{z}_t)d\mathbf{z}_t} + \xi_0\right), \qquad (7)$$

where σ is the sigmoid function and $\xi_0 = \log p(s_t = 1) - \log p(s_t = 0)$ are the log-odds of the prior $p(s_t)$.

Eq. 7 has a simple interpretation as a likelihood ratio test: a change is more or less likely depending on whether or not the observations $\mathbf{x}_t$ are better explained under the assumption of the broadened prior distribution $q^\beta_{t-1}(\mathbf{z}_t)$, in other words, a partial reset of previously learned information.

Then we demonstrate the resemblance between the exact inference and the approximate inference for s_t. Notice the conditional ELBO in Eq. 3 is a lower bound to the logarithm of the marginal likelihood $\log p_\beta(\mathbf{x}_t|s_t)$, we can use the former as a proxy of the latter in Eq. 7, resulting in the update in Eq. 4. In other words, the tighter the conditional ELBO to $\log p_\beta(\mathbf{x}_t|s_t)$, $q(s_t)$ is more precise.

Appendix B. Details on Online Learning and Variational Beam Search

Variational Greedy Search For each t, VGS updates a *truncated* variational distribution via the following three steps:

1. Compute the conditional prior $p_\beta(\mathbf{z}_t|s_t)$ (Eq. 2) based on q_{t-1} and optimize the conditional ELBO (Eq. 3) for both $s_t = 0$ and $s_t = 1$. This results in the optimized variational distributions $q(\mathbf{z}_t|s_t = 0)$ and $q(\mathbf{z}_t|s_t = 1)$.
2. Compute the binary mixture weights m and $(1-m)$ using Eq. 4, resulting in $q_t(\mathbf{z}_t) = m\, q(\mathbf{z}_t|s_t = 0) + (1 - m)q(\mathbf{z}_t|s_t = 1)$ (Eq. 5).

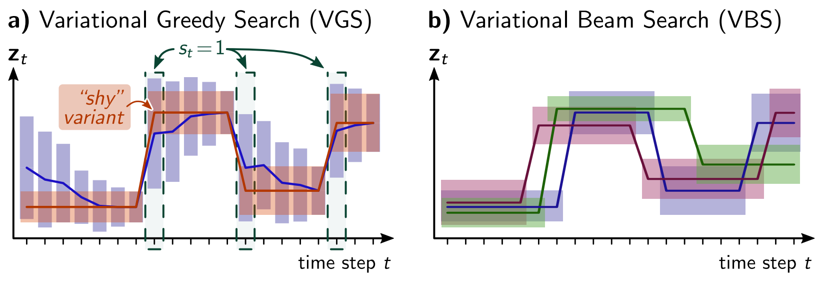

Figure 3: Sparse inference via greedy search (left) and variational beam search (right), see also Section B.

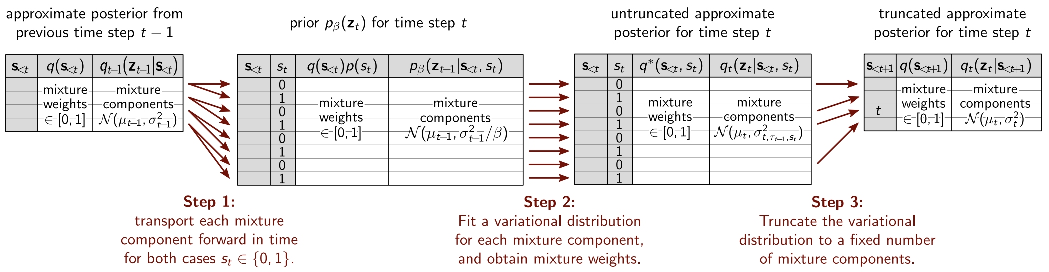
approximate posterior from previous time step $t-1$
prior $p_\beta(\mathbf{z}_t)$ for time step t
untruncated approximate posterior for time step t
truncated approximate posterior for time step t
$\mathbf{s}_{<t}$ | $q(\mathbf{s}_{<t})$ | $q_{t-1}(\mathbf{z}_{t-1}|\mathbf{s}_{<t})$
mixture weights $\in [0,1]$
mixture components $\mathcal{N}(\mu_{t-1}, \sigma^2_{t-1})$
$\mathbf{s}_{<t}$ | s_t | $q(\mathbf{s}_{<t})p(s_t)$ | $p_\beta(\mathbf{z}_{t-1}|\mathbf{s}_{<t}, s_t)$
0 1 0 1 0 1 0 1
mixture weights $\in [0,1]$
mixture components $\mathcal{N}(\mu_{t-1}, \sigma^2_{t-1}/\beta)$
$\mathbf{s}_{<t}$ | s_t | $q^*(\mathbf{s}_{<t}, s_t)$ | $q_t(\mathbf{z}_t|\mathbf{s}_{<t}, s_t)$
0 1 0 1 0 1 0 1
mixture weights $\in [0,1]$
mixture components $\mathcal{N}(\mu_t, \sigma^2_{t,\tau_{t-1},s_t})$
$\mathbf{s}_{<t+1}$ | $q(\mathbf{s}_{<t+1})$ | $q_t(\mathbf{z}_t|\mathbf{s}_{<t+1})$
t
mixture weights $\in [0,1]$
mixture components $\mathcal{N}(\mu_t, \sigma^2_t)$
Step 1: transport each mixture component forward in time for both cases $s_t \in \{0,1\}$.
Step 2: Fit a variational distribution for each mixture component, and obtain mixture weights.
Step 3: Truncate the variational distribution to a fixed number of mixture components.

Figure 4: Conditional probability table of variational beam search

3. Truncate $q_t(\mathbf{z}_t)$ to a uni-modal distribution to avoid branching. This can be achieved by projecting $q_t(\mathbf{z}_t)$ on the more probable component.

The filtering algorithm mentioned above iteratively updates the variational distribution over $\mathbf{z}_t$ each time it observes new data $\mathbf{x}_t$. In the version of variational greedy search discussed above, the approach decides immediately, i.e., before observing subsequent data points, whether a change in $\mathbf{z}_t$ has occurred (by projecting on one mixture component in the truncation step) or not. This approach is illustrated in the blue part of Fig. 3 (a). Here, the dark blue line shows the fitted mean μ_t over time steps t, with $\pm 1\sigma$ error bars in light blue. The fact that μ_t and σ_t change even between two detected change points makes the output of the algorithm poorly interpretable.

"Shy" Variational Greedy Search One obtains a better fit if one outputs the variational parameters μ_t and σ_t at the end of a segment of constant $\mathbf{z}_t$. More precisely, when the algorithm detects a change point $s_t = 1$, it outputs the variational parameters μ_{t-1} and σ_{t-1} from just before the detected change point t. These parameters define a variational distribution that has been fitted, in an iterative way, to all data points since the preceding detected change point. We call this the "shy" variant of the variational greedy search algorithm, because this variant quietly iterates over the data and only outputs a new fit when it is as certain about it as it will ever be. The red lines and regions in Fig. 3 (a) illustrate means and standard deviations outputted by the "shy" variant of variational greedy search.

Variational Beam Search As we see in the main paper, the above two algorithm variants ("greedy" and "shy") discussed so far are greedy in the detection of change points, prone to missing change points in noisy data sets. An obvious improvement that can delay decisions to accumulate evidence is beam search. This algorithm is illustrated schematically in Fig. 3 (b). The algorithm keeps track of a fixed number $K > 1$ of possible histories, i.e., sequences of change points. For each history, it iteratively updates a Gaussian variational distribution as in the greedy variants. At each time step t, each history splits up into two for the two cases $s_t \in \{0, 1\}$, thus doubling the number of histories of which the algorithm has to keep track of. To keep the computational requirements bounded, beam search thus discards half of the histories based on an exploration-exploitation trade-off.

As follows, we present a more detailed explanation of the variational beam search procedure outlined in Section 2.2 of the main paper. Our beam search procedure defines an effective way to search for potential hypotheses with regards to sequences of inferred change points. The procedure is completely defined by detailing three sequential steps, that when executed, take a set of hypotheses found at time step $t-1$ and transform them into the resulting set of likely hypotheses for time step t that have appropriately accounted for the new data seen at t. The red arrows in Figure 4 illustrate these three steps for beam search with a beam size of $K = 4$.

In Figure 4, each of the three steps maps a table of considered histories to a new table. Each table defines a mixture of Gaussian distributions where each mixture component corresponds to a different history and is represented by different a row in the table. We start on the left with the (truncated) variational distribution $q_{t-1}(\mathbf{z}_{t-1})$ from the previous time step, which is a mixture over $K = 4$ Gaussian distributions. Each mixture component (row in the table) is labeled by a 0-1 vector $\mathbf{s}_{<t} = (s_0, s_1, \cdots, s_{t-1})$ of the change variable values according to that history. Each mixture component $\mathbf{s}_{<t}$ further has a mixture weight $q(\mathbf{s}_{<t}) \in [0, 1]$, a mean, and a standard deviation.

We then obtain a prior for time step t by transporting each mixture component of $q_{t-1}(\mathbf{z}_{t-1})$ forward in time via the broadening functional ("Step 1" in the above figure). The prior $p_\beta(\mathbf{z}_t)$ (second table in the figure) is a mixture of $2K$ Gaussian distributions because each previous history splits into two new ones for the two potential cases $s_t \in \{0, 1\}$. The label for each mixture component (table row) is a new vector $(\mathbf{s}_{<t}, s_t)$ or $\mathbf{s}_{<t+1}$, appending s_t to the tail of $\mathbf{s}_{<t}$.

"Step 2" in the above figure takes the data $\mathbf{x}_t$ and fits a variational distribution $q_t(\mathbf{z}_t)$ that is also a mixture of $2K$ Gaussian distributions. To learn the variational distribution, we (i) numerically fit each mixture component $q(\mathbf{z}_t|\mathbf{s}_{<t}, s_t)$ individually, using the corresponding mixture component of $p_\beta(\mathbf{z}_t)$ as the prior; (ii) evaluate (or estimate) the conditional ELBO of each fitted mixture component, conditioned on $(\mathbf{s}_{<t}, s_t)$; (iii) compute the approximate posterior probability $q^*(s_t)$ of each mixture component, in the presence of the conditional ELBOs; and (iv) obtain the mixture weight equal to the posterior probability over $(\mathbf{s}_{<t}, s_t)$, best approximated by $q(\mathbf{s}_{<t})q^*(s_t)$.

"Step 3" in the above figure truncates the variational distribution by discarding K of the $2K$ mixture components. The truncation scheme can be either the "vanilla" beam search or diversified beam search outlined in the main paper. The truncated variational distribution $q_t(\mathbf{z}_t)$ is again a mixture of only K Gaussian distributions, and it can thus be used for subsequent update steps, i.e., from t to $t + 1$.

Beam Search Diversification To encourage diverse beams, we constructed the following simple scheme: Let K be the number of hypotheses in a beam. While transitioning from time $t-1$ to t, every hypothesis splits into two scenarios, one with $s_t = 0$ and one with $s_t = 1$, resulting in $2K$ hypotheses. If two resulting hypotheses only differ in their most recent s_t-value, we say that they come from the same "family." Each member among the $2K$ hypotheses is ranked according to its ELBO value (Eq. 3). In a first step, we discard the bottom $1/3$ of the $2K$ hypotheses, leaving $4/3K$ hypotheses. (We always take integer multiples of 3 for K). To truncate the beam size from $4/3K$

Table 2: Convolution Neural Network Architecture

LAYER	FILTER SIZE	FILTERS	STRIDE	ACTIVATION	DROPOUT
CONVOLUTIONAL	3×3	32	1	RELU	
CONVOLUTIONAL	3×3	32	1	RELU	
MAXPOOLING	2×2		2		0.2
CONVOLUTIONAL	3×3	64	1	RELU	
CONVOLUTIONAL	3×3	64	1	RELU	
MAXPOOLING	2×2		2		0.2
FULLYCONNECTED		10		SOFTMAX	

Table 3: Hyperparameters of Bayesian Deep Learning Models for CIFAR-10

MODEL	LEARNING RATE	BATCH SIZE	NUMBER OF EPOCHS	β	ξ_0
LP	0.001	64	150	N/A	N/A
VCL	0.0005	64	150	N/A	N/A
VBS	0.0005	64	150	2/3	0

down to K, we rank the remaining hypotheses according to their ELBO and pick the top K ones while *also* ensuring that we pick a member from every remaining family. The diversification scheme ensures that underperforming families can survive, leading to a more diverse set of hypotheses.

Appendix C. Experiment Details and Results

C.1. Toy Data Experiments

Data Generating Process To generate Figure 1 in the main paper, we used a step-wise function as ground truth, where the step size was 1 and two step positions were chosen randomly. We sampled 30 equally-spaced points with time spacing 1. To get noisy observations, Gaussian noise with standard deviation 0.5 was added to the points.

Model Parameters In this simple one-dimensional model, we used absolute broadening with a Gaussian transition kernel $K(\mathbf{z}_t, \mathbf{z}'_t) = \mathcal{N}(\mathbf{z}_t - \mathbf{z}'_t, D\Delta t)$ where $D = 1.0$ and $\Delta t = 1$. The inference is thus tractable because $p(\mathbf{z}_t|s_t)$ is conditional conjugate to $p(\mathbf{x}_t|\mathbf{z}_t, s_t)$ (and both are Gaussian distributed). We set the prior log-odds ϵ_0 to $\log \frac{p(s_t=1)}{p(s_t=0)}$, where $p(s_t = 1) = 0.1$. We used beam size 1 and beam size 2 to do the inference, respectively. We reported the results with beam size 2, which also involved the resulting hypothesis for beam size 1 (greedy search) as the one with lower probability.

C.2. Bayesian Deep Learning Experiments

Transformations We used Albumentations (Buslaev et al., 2020) to implement the transformations as covariate shifts. As stated in the main paper, the transformation involved rotation, scaling, and translation. Each transformation factor followed a fixed distribution: rotation degree conformed to $\mathcal{N}(0, 10^2)$; scaling limit conformed to $\mathcal{N}(0, 0.3^2)$; and the magnitude of vertical and horizontal

Table 4: Hyerparameters of Bayesian Deep Learning Models for SVHN.

MODEL	LEARNING RATE	BATCH SIZE	NUMBER OF EPOCHS	β	ξ_0
LP	0.001	64	150	N/A	N/A
VCL	0.00025	64	150	N/A	N/A
VBS	0.00025	64	150	2/3	0

translation limit conformed to Beta$(1, 10)$, and the sampled magnitude is then rendered positive or negative with equal probability. The final scaling and translation factor should be the corresponding sampled limit plus 1, respectively.

Neural Network Architecture We used the same convolutional neural network architecture for both datasets, which can be found in Table 2. We implemented the Bayesian models using TensorFlow Probability and the non-Bayesian counterpart (namely Laplace Propagation) using TensorFlow Keras. Every bias term in all the models were treated deterministically and were not affected by any regularization.

Tempered Conditional ELBO In the presence of massive observations and a large neural network, posterior distributions of change variables usually have very low entropy because of the very large magnitude of the difference between conditional ELBOs as in Eq. 4. Therefore change variables become over confident about the switch-state decisions. The situation gets even more severe in beam search settings where almost all probability mass is centered around the top hypothesis while the other hypotheses get little probability and thereby will not take effect in predictions. A possible solution is to temper the conditional ELBO (or the marginal likelihood) and introduce more uncertainty into the change variables. To this end, we divide the conditional ELBO by the number of observations. While the tempering strategy has the model no longer work in strict Bayesian paradigm, it renders every hypothesis effective in beam search setting.

Hyperparameters, Initialization, and Model Training The hyperparameters used across all of the models for the different datasets are listed in Tables 3 and 4. Regarding the model-specific parameters, we set ξ_0 to 0 for both datasets and searched β in the values $\{5/6, 2/3, 1/2, 1/4\}$ on a validation set. We used the first 5000 images in the original test set as the validation set, and the others as the test set. We found that $\beta = 2/3$ performs relatively well for both data sets. Optimization parameters, including learning rate, batch size, and number of epochs, were selected to have the best validation performance of the classifier on one independent task. To estimate the change variable s_t's variational parameter, we approximated the conditional ELBOs 3 by averaging 10000 Monte Carlo samples.

As outlined in the main paper, we initialized each algorithm by training the model on the full, untransformed dataset. The model weights used a standard Gaussian distribution as the prior for this meta-initialization step.

When optimizing with variational inference, we initialized $q(\mathbf{z}_t)$ to be a point mass around zero for stability. When performing non-Bayesian optimization, we initialized the weights using Glorot Uniform initializer (Glorot and Bengio, 2010). All bias terms were initialized to be zero.

We performed both the Bayesian and non-Bayesian optimization using ADAM (Kingma and Ba, 2014). For additional parameters of the ADAM optimizer, we set $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for

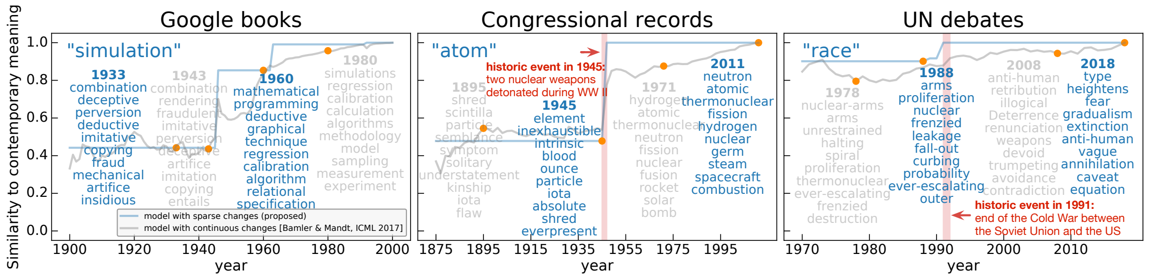

Figure 5: Dynamic Word Embeddings on Google books, Congressional records, and UN debates, trained with VBS (proposed, blue) vs. VCL (grey). In contrast to VCL, VBS reveals sparse, time-localized semantic changes (see main text).

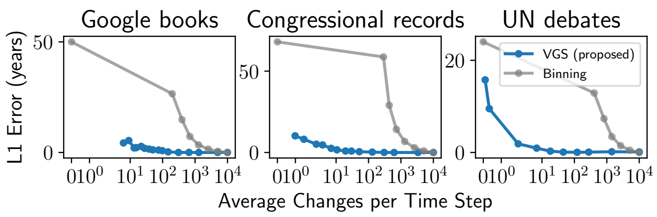

Figure 6: Document dating error as a function of model sparsity, measured in average parameter updates per year.

both data sets. For the deep Bayesian models specifically, which include VCL and VBS, we used stochastic black box variational inference (Ranganath et al., 2014; Kingma and Welling, 2013; Zhang et al., 2018). We also used the Flipout estimator (Wen et al., 2018) to reduce variance in the gradient estimator.

Predictive Distributions We evaluated the predictive posterior distribution of the test set by the following approximation:

$$p(\mathbf{y}_t|\mathbf{x}_t, \mathcal{D}_{1:t}) \approx \frac{1}{S}\sum_{k=1}^{K}\sum_{s=1}^{S} w_k p(\mathbf{y}_t|\mathbf{x}_t, \mathbf{z}_{t,k}^{(s)})$$

where K is the beam size, w_k is the normalized weight of the k^{th} hypothesis after truncation, and S is the number of Monte Carlo samples from the variational posterior distribution $q_t(\mathbf{z}_t)$. In our experiments we found $S = 10$ to be sufficient. We take $\arg\max_{\mathbf{y}_t} p(\mathbf{y}_t|\mathbf{x}_t, \mathcal{D}_{1:t})$ to be the predicted class.

VGS and VCL set $K = 1$ for the single hypothesis in the above formula. LP further only used the MAP estimation $\mathbf{z}_t^*$ to predict the test set: $p(\mathbf{y}_t|\mathbf{x}_t, \mathcal{D}_{1:t}) \approx p(\mathbf{y}_t|\mathbf{x}_t, \mathbf{z}_t^*)$.

C.3. Unsupervised Experiments

Our third experiment focused on unsupervised learning, where our focus was different: instead of showing performance improvements in terms of higher accuracy, we demonstrated that VBS and VGS help uncover interpretable latent structure in high-dimensional time series, such as localizing changes in the meanings of individual words over large time spans. We also show that these word embedding trajectories can be used for document dating.

Datasets We analyzed three large time-stamped text corpora, all of which are available online. Our first dataset is the Google Books corpus (Michel et al., 2011) consisting of n-grams, which is sufficient for learning word embeddings. We focused on the period from 1900 to 2000. To have an approximately even amount of data per year, we sub-sampled 250M to 300M tokens per year. Second, we used the Congressional Records data set (Gentzkow et al., 2018), which has 13M to 52M tokens per two-year period from 1875 to 2011. Third, we used the UN General Debates corpus (Jankin Mikhaylov et al., 2017), which has about 250k to 450k tokens per year from 1970 to 2018. The vocabulary size was 30000 for all three corpora. We further randomly split the corpus of every time step into training set (90%) and heldout test set (10%).

Model and Baselines To analyze the semantic changes of individual words over time, we used Dynamic Word Embeddings (Bamler and Mandt, 2017) as our base model in all experiments. The model relies on a probabilistic interpretation of Word2Vec and learns smooth word embedding trajectories by imposing a time-series prior on the embeddings. We compared against the online version of dynamic word embeddings with a diffusion prior, which is equivalent to VCL. By construction, this prior does not encourage temporal sparsity, meaning the word embeddings change every year. For our proposed approach, we imposed our spike and slab novelty prior (Eq. 2) that induces temporal sparsity, allowing us to time-localize semantic changes of words. We set the relative broadening constant $\beta = 0.5$ for Google books and Congressional records and let $\beta = 0.25$ for UN debates. We combined this prior with beam search ($K = 2$), where we reported results on the most likely beam.

We also considered regular word embeddings trained on individual years (Mikolov et al., 2013). While (Bamler and Mandt, 2017) already carried-out the comparison to dynamic word embeddings, we used them to measure the compressibility of word embedding trajectories for document dating tasks, as we describe below.

Qualitative Results Our first experiments demonstrate that a spike & slab prior is more interpretable and results in more meaningful word semantics than the diffusion prior of (Bamler and Mandt, 2017). Figure 5 shows three selected words ("simulation", "atom", and "race"–one taken from each corpus) and their nearest neighbors in latent space. As time progresses the nearest neighboring words change, reflecting a semantic change of the words. While the horizontal axis shows the year, the vertical axis shows the cosine distance of the word's embedding vector at the given year to its embedding vector in the last available year.

The plot reveals several interpretable semantic changes of each word. For example, “atom” changes its meaning from “element” to “nuclear” in 1945–the year when two nuclear bombs were detonated. The word “race” changes from the cold-war era “arms”(-race) to its more prevalent meaning after 1991 when the cold war ended. The word "simulation" changes its dominant context from "deception" to "programming" with the advent of computers. The plot also compares the nearest

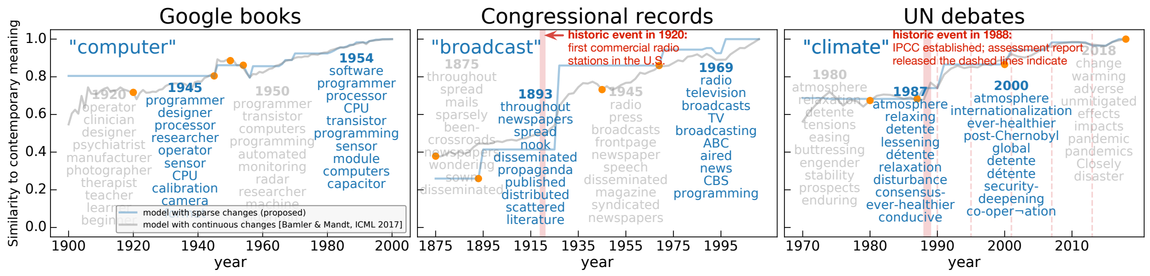

Figure 7: Additional results of Dynamic Word Embeddings on Google books, Congressional records, and UN debates.

neighbors obtained with the diffusion prior (Bamler and Mandt, 2017) in grey to our VBS-based approach in blue, with arguably more interpretable results.

Quantitative Results. Another benefit of dynamic sparsity, as induced by VBS, is a more compressible representation of the revealed latent states. For dynamic word embeddings, word embedding trajectories can be stored as a list of embedding vectors with associated time stamps–one vector per update. In contrast, the work of (Bamler and Mandt, 2017) or (Mikolov et al., 2013) would require a separate embedding vector for every word for all years.

We analyzed a *rate-distortion tradeoff* in the classical compression sense, where we tuned the prior on the change variable s_t (the prior log-odds ξ_0 in Eq. 4) to suppress semantic changes. With changes being suppressed, naturally the word embeddings get worse in quality, but lead to smaller file sizes as only *updated* word embeddings have to be stored. Figure 6 shows the results on Congressional Records data. To measure model performance, we predicted the year in which each held-out document's word-word co-occurrence statistics have the highest likelihood and measured $L1$ error.

As a simple baseline, we assumed that we had separate word embeddings associated with episodes of L consecutive years. For T years in total, the associated memory requirements would be proportional to $V * T/L$, where V is the vocabulary size. Assuming we could perfectly date the document up to L years results in the "binning" baseline in Figure 6. We see that our approach results in a much better rate-distortion tradeoff since we are allowing each word to change at different times, as opposed to only collectively.

C.4. More on Dynamic Word Embeddings Experiments

Model Assumptions As outlined in the main paper, we analyzed the semantic changes of individual words over time. We augmented the probabilistic models proposed by Bamler and Mandt (2017) with the spike and slab novelty prior (Eq. 2 in the main paper) to encourage temporal sparsity. We pre-trained the *context* word embeddings[4] using the whole corpus, and kept them constant when updating the *target* word embeddings. This practice denied possible interference on one target word embedding from the updates of the others. If we did not employ this practice, the spike and

4. We refer readers to (Mikolov et al., 2013; Bamler and Mandt, 2017) for the difference between target and context word embeddings.

Table 5: Hyerparameters of Dynamic Word Embedding Models

CORPUS	VOCAB	DIMS	β	LEARNING RATE	EPOCHES	ξ_0	BEAMS
GOOGLE BOOKS	30000	100	0.5	0.01	5000	-10	2
CONGRESSIONAL RECORDS	30000	100	0.5	0.01	5000	-10	2
UN DEBATES	30000	20	0.25	0.01	5000	-1	2

slab prior on word i would lead to two branches of the "remaining vocabulary" (embeddings of the remaining words in the vocabulary), conditioned either on the spike prior of word i or on the slab prior. This hypothetical situation gets severe when every word in the vocabulary can take two different priors, thus leading to exponential branching of the sequences of inferred change points. When this interference is allowed, the exponential scaling of hypotheses translates into exponential scaling of possible word embeddings for a single target word, which is not feasible to compute for any meaningful vocabulary sizes and number of time steps. To this end, while using a fixed, pre-trained context word embeddings induces a slight drop of predictive performance, the computational efficiency improves tremendously and the model can actually be learned.

Hyperparameters and Optimization Qualitative results in Figure 5 in the main paper were generated using the hyperparameters in Table 5. The initial prior distribution used for all latent embedding dimensions was a standard Gaussian distribution. We also initialized all variational distributions with standard Gaussian distributions. For model-specific hyperparameters β and ξ_0, we first searched the broadening constant β to have the desired jump magnitude observed from the semantic trajectories mainly for medium-frequency words. We then tuned the bias term ξ_0 to have the desired change frequencies in general. We did the searching for the first several time steps. We performed the optimization using black box variational inference and ADAM. For additional parameters of ADAM optimizer, we set $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for all three corpora. In this case, we did **not** temper the conditional ELBO by the number of observations.

Quantitative results of VGS in Figure 6 in the main paper were generated by setting a smaller vocabulary size and embedding dimension, 10,000 and 20, respectively for all three corpora. Other hyperparameters were inherited from the qualitative experiments. On the other hand, the baseline, "binning", and had closed-form performance if we assume (i) a uniformly distributed year in which a document query is generated, (ii) "binning" perfectly locates the ground truth episode, and (iii) the dating result is uniformed distributed within the ground truth episode. The $L1$ error associated with "binning" with episode length L is $\mathbb{E}_{t \sim \mathcal{U}(1,L), t' \sim \mathcal{U}(1,L)}[|t - t'|] = \frac{L-1}{2}$. By varying L, we get binning's rate-distortion curve in Figure 6 in the main paper.

Additional Results Additional qualitative results can be found in Figure 7. it, again, reveals interpretable semantic changes of each word: the first change of "computer" happens in 1940s–when modern computers appeared; "broadcast" adopts its major change shortly after the first commercial radio stations were established; "climate" changes its meaning at the time when Intergovernmental Panel on Climate Change (IPCC) was set up, and when it released the assessment reports to address the implications and potential risks of climate changes.

Predictive Distributions In the demonstration of the quantitative results, i.e., the document dating experiments, we predicted the year in which each held-out document's word-word co-occurrence statistics $\mathbf{x}$ have the highest likelihood and measured $L1$ error. To be specific, for a given document in year t, we approximated its likelihood under year t' by evaluating $\frac{1}{|V|}\log p(\mathbf{x}_t|\mathbf{z}^*_{t'})$, where $\mathbf{z}^*_{t'}$ is the mode embedding in year t' and $|V|$ is the vocabulary size. We predicted the year $t^* = \arg\max_{t'} \frac{1}{|V|}\log p(\mathbf{x}_t|\mathbf{z}^*_{t'})$. We then measured the $L1$ error by $\frac{1}{T}\sum_i^T |t_i - t_i^*|$ given T truth-prediction pairs.