POST-HOC STOCHASTIC CONCEPT BOTTLENECK MODELS

Wiktor Jan Hoffmann, Sonia Laguna[*], Moritz Vandenhirtz, Emanuele Palumbo, Julia E. Vogt
Department of Computer Science, ETH Zurich, Switzerland

ABSTRACT

Concept Bottleneck Models (CBMs) are interpretable models that predict the target variable through high-level human-understandable concepts, allowing users to intervene on mispredicted concepts to adjust the final output. While recent work has shown that modeling dependencies between concepts can improve CBM performance, especially under interventions, such approaches typically require retraining the entire model, which may be infeasible when access to the original data or compute is limited. In this paper, we introduce Post-hoc Stochastic Concept Bottleneck Models (PSCBMs), a lightweight method that augments any pre-trained CBM with a multivariate normal distribution over concepts by adding only a small covariance-prediction module, without modifying the backbone model. We propose two training strategies and show on real-world data that PSCBMs consistently match or improve both concept and target accuracy over standard CBMs at test time. Furthermore, we show that due to the modeling of concept dependencies, PSCBMs perform much better than CBMs under interventions, while remaining far more efficient than retraining a similar stochastic model from scratch.

1 INTRODUCTION

The adoption of machine learning models in high-stakes domains is often limited by their lack of interpretability due to their black-box nature (Lipton, 2016; Doshi-Velez & Kim, 2017). Concept Bottleneck Models (CBMs), first introduced by Koh et al. (2020), address this by inserting a layer of neurons aligned with human-understandable concepts before the target prediction. A CBM is composed of a concept encoder that predicts the concepts, and a prediction head that takes these concepts as inputs; the final prediction is thus explained through the intermediate concept values. While the original formulation assumes independence among concepts, accounting for correlations has been shown to improve the model's performance (Havasi et al., 2022; Singhi et al., 2024; Vandenhirtz et al., 2024). However, existing approaches that capture concept dependencies typically require training the entire model with dedicated objectives. In contrast, we demonstrate that such dependencies can be incorporated post-hoc into a pre-trained CBM. A more detailed discussion of related work is provided in Appendix A.

Building on this idea, we introduce Post-Hoc Stochastic Concept Bottleneck Models (PSCBMs)[1], which extend pre-trained CBMs with a lightweight module that predicts concept covariance. Inspired by Stochastic Concept Bottleneck Models (SCBMs) (Vandenhirtz et al., 2024), we model concept dependencies with a multivariate normal distribution. This approach not only improves predictive performance, but also enables more efficient concept interventions—the process by which a user modifies predicted concepts to directly influence downstream predictions.

This work contributes to model interpretability through human-understandable concepts in three ways: *(i)* We introduce PSCBM, a lightweight post-hoc module that incorporates concept correlations into existing CBMs without requiring full retraining, thereby reducing compute requirements. *(ii)* We propose a simple intervention-based training procedure that further improves intervention efficiency in (P)SCBM-like models. *(iii)* We show on real-world data that PSCBM improves both predictive accuracy and intervention effectiveness while remaining substantially more efficient than full model retraining.

[*] Correspondence to `slaguna@inf.ethz.ch`

[1] The code is available at `https://github.com/wiktorhof/PSCBM`.

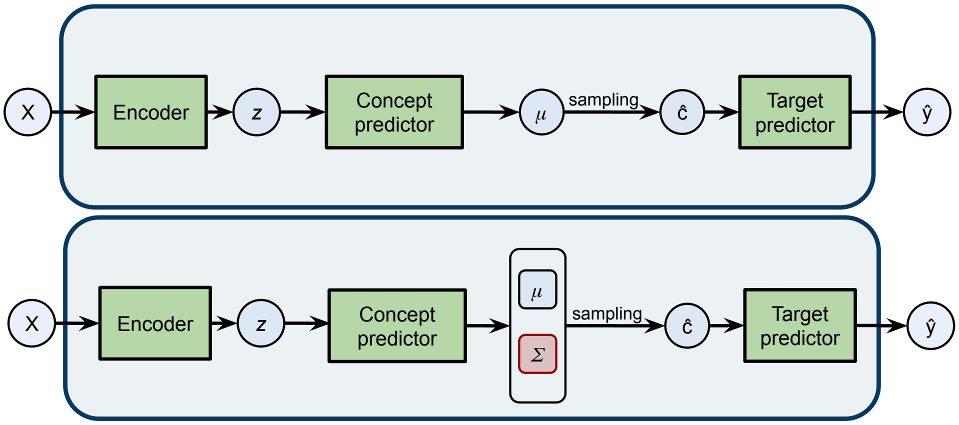

Figure 1: Comparison of a hard CBM (top), SCBM (bottom), and our proposed PSCBM (bottom, red block). All methods input $\boldsymbol{x}$ to an encoder to produce the feature vector $\boldsymbol{z}$ from which concepts are obtained and fed to the target predictor. CBM directly predicts concept values $\hat{\boldsymbol{c}}$, while in SCBM, the predictor outputs an expected value $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$ that define a multivariate normal distribution to sample from. PSCBM incorporates the $\boldsymbol{\Sigma}$ predictor (red box) to a frozen pre-trained CBM.

2 METHOD

Modeling concept dependencies with a normal distribution In this paper we propose PSCBMs, a post-hoc extension of CBMs that enables to model dependencies between concepts without retraining the model from scratch. A CBM Koh et al. (2020) consists of three main components: a feature encoder, a concept predictor, and a target predictor. The feature encoder h (e.g., a CNN or ResNet) extracts features $\boldsymbol{z} = h(\boldsymbol{x})$ from an input $\boldsymbol{x}$. Then the concept predictor g outputs concept probabilities $\boldsymbol{p} = g(\boldsymbol{z})$. When focusing on binary concepts for downstream prediction (i.e. *hard* CBMs (Havasi et al., 2022)), rather than their probabilities (*soft* CBMs), the next step is sampling M concept values from a Bernoulli distrbution $\hat{\boldsymbol{c}}^{(1)}, \dots \hat{\boldsymbol{c}}^{(M)} \sim \text{Bernoulli}(\boldsymbol{p})$. Then the target predictor f maps the sampled concepts to the final prediction $\hat{y} = \frac{1}{M}\sum_{m=1}^{M} f(\hat{\boldsymbol{c}}^{(m)})$. SCBMs (Vandenhirtz et al., 2024) extend this framework by explicitly capturing correlations between concepts: the concept predictor is extended to feature a mean predictor g_μ and a covariance predictor g_Σ, which define a multivariate normal distribution that models concept dependencies. Concepts are sampled as $\hat{\boldsymbol{c}}^{(m)} = \sigma(\eta^{(m)})$ for $m = 1, \dots, M$, where $\eta^{(m)} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and σ is the sigmoid function. Our proposed PSCBMs directly build upon this idea: given a pre-trained CBM, we reuse its concept predictor g as g_μ and add a lightweight covariance predictor g_Σ, training only the latter while keeping the rest of the model frozen. In this way, any existing CBM can be turned into a stochastic, dependency-aware model post-hoc, as illustrated in Figure 1.

Interventions in Stochastic Concept Bottleneck Models A distinguishing property of CBMs is that they allow users to modify predicted concept values at test time and thereby directly update the downstream prediction. This process, known as intervention, can be factorized into two parts: the policy π, which selects the concepts to be intervened on (e.g., randomly or based on uncertainty), and the strategy τ, which determines how their values are updated. We describe several policies and strategies in Appendix B. Intervening in SCBMs or PSCBMs additionally accounts for concept dependencies and proceeds in four steps:

1. **Select concepts for intervention** Use a policy π to choose a subset $\mathcal{S}$ of concepts.
2. **Set intervened logits** Assign values to $\boldsymbol{\eta}'_{\mathcal{S}}$, the logits of intervened concepts, according to a chosen intervention strategy τ.
3. **Update remaining logits** Compute the conditional normal distribution parameterized by $\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\Sigma}}$ for the non-intervened concept set $\setminus\mathcal{S}$, using the equations for a conditional normal distribution: $\boldsymbol{\eta}_{\setminus\mathcal{S}} \mid \boldsymbol{x}, \boldsymbol{\eta}'_{\mathcal{S}} \sim \mathcal{N}\left(\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\Sigma}}\right)$,

   where $\bar{\boldsymbol{\mu}} = \boldsymbol{\mu}_{\setminus\mathcal{S}} + \boldsymbol{\Sigma}_{\setminus\mathcal{S},\mathcal{S}}\boldsymbol{\Sigma}^{-1}_{\mathcal{S},\mathcal{S}}(\boldsymbol{\eta}'_{\mathcal{S}} - \boldsymbol{\mu}_{\mathcal{S}}), \bar{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}_{\setminus\mathcal{S},\setminus\mathcal{S}} - \boldsymbol{\Sigma}_{\setminus\mathcal{S},\mathcal{S}}\boldsymbol{\Sigma}^{-1}_{\mathcal{S},\mathcal{S}}\boldsymbol{\Sigma}_{\mathcal{S},\setminus\mathcal{S}}$.

4. **Sample probabilities** Sample binary concept values $\boldsymbol{c}_{\setminus\mathcal{S}}$ from the logits $\boldsymbol{\eta}_{\setminus\mathcal{S}}$.

Learning the covariance matrix post-hoc The covariance predictor $g_{\boldsymbol{\Sigma}}$ is trained by minimizing the loss function of a regular SCBM. It teaches the model to predict both the concepts and target variable correctly and encourages sparsity in the covariance matrix:

$$\mathcal{L} = \underbrace{-\log \sum_{m=1}^{M} \exp \sum_{i=1}^{C} -\mathrm{BCE}\left(c_i, \sigma(\eta_i^{(m)})\right)}_{\text{Concept Loss}} + \lambda_1 \underbrace{\mathrm{CE}\left(y, \frac{1}{M} \sum_{m=1}^{M} g_{\boldsymbol{\psi}}(\boldsymbol{c}^{(m)})\right)}_{\text{Target Loss}} + \lambda_2 \underbrace{\sum_{i \neq j} \boldsymbol{\Sigma}_{i,j}^{-1}}_{\text{Regularization}} .$$

Here, M is the number of Monte Carlo concept samples, C is the number of concepts, BCE and CE denote the (Binary) Cross-Entropy, $\boldsymbol{x}$ is the input, y the true label, c_i the true concept values, $\eta^{(m)}$ the sampled logit vectors, $\sigma(\cdot)$ the sigmoid function, $\boldsymbol{c}^{(m)}$ the binary concept values sampled from the Bernoulli distribution defined by $\sigma(\eta^{(m)})$, $\boldsymbol{\Sigma}(\boldsymbol{x})$ the predicted covariance matrix and $g(\cdot)$ the target predictor. Finally, λ_1 and λ_2 are weighting parameters regulating the loss strength.

We propose two training paradigms for PSCBM, which can be extended to SCBM: *(i)* The loss function is applied to the model's predictions without concept interventions. *(ii)* We encourage more responsive predictors to concept interventions. In every training iteration, for every sample in the training dataset, we randomly select a set $\mathcal{S}$ of concepts to intervene on using some strategy τ and calculate the loss after the intervention. The cardinality of $\mathcal{S}$ is fixed throughout training. This is done N times for each data point and the average of the loss for all these interventions is calculated. This method is similar to *RandInt* introduced by Espinosa Zarlenga et al. (2022) but it differs in two ways: *(i)* Unlike *RandInt*, which decides independently for each concept whether to replace it with its true value, our method enforces that a fixed number of concepts is intervened on in every case. *(ii)* At each training iteration, multiple random interventions per sample are made and the loss is averaged. This reduces gradient variance and yields more stable training compared to applying a single intervention. A detailed pseudocode for the calculation of this loss can be found in Appendix C.

3 Results

In our experiments, we address two main questions: **1.** How does post-hoc learning of the covariance affect the test accuracy without interventions? **2.** How does it affect predictions after interventions?

Experimental setup We compare the performance of PSCBM to SCBM and regular CBM. PSCBM indicates training without concept interventions, while PSCBMi includes interventions during training. SCBM proposes two ways to learn the covariance matrix, amortized per instance, $\boldsymbol{\Sigma}(\boldsymbol{x})$, or learnt globally, $\boldsymbol{\Sigma}$. For brevity, we present the results on the global covariance, and include their amortized counterparts in Appendix D. All implementation details are described in Appendix E. For test-time interventions, concepts are selected based on their associated prediction uncertainty, i.e., the concept with predicted probability closest to 0.5 is chosen. Unlike CBM where the intervened probabilities are set to 0/1, SCBMs and PSCBMs set intervened concept logits to those corresponding to ϵ or $1 - \epsilon$, respectively, where ε is small for stability. We evaluate our methods on the Caltech-UCSD Birds-200-2011 dataset (Wah et al., 2011), composed of photographs of birds of 200 different classes. We use the variant introduced in Koh et al. (2020) for CBMs with 112 binary per-class concepts and, as evaluation metrics, we focus on concept and target accuracy.

Test performance In Table 1 we present concept and target accuracy of the evaluated models without interventions. The results show that adding our covariance learning module improves performance. Both variants of PSCBM outperform SCBM on both concept and target accuracy. For target accuracy, PSCBM trained without interventions clearly surpasses the regular CBM, while on concept accuracy the two achieve comparable results. The training times presented in the last column show that training a PSCBM without interventions is computationally inexpensive due to the fact that only one module is trained. Training with interventions however requires significantly more time.

Intervention performance To evaluate intervention performance we rely on two approaches: visual inspection of the intervention curves, and the area under the intervention curve (AUC) (Singhi et al., 2024), which summarizes each curve into a single value for easier comparison. The last two columns

Table 1: Test-time target and concept accuracy without interventions and their AUCs under interventions. AUC is normalised to stay within the interval [0,1]. We report the mean and standard deviation over three runs for baselines and nine runs for PSCBM. The best scores for each metric are highlighted with **bold** font, and those within a standard deviation from the best one are underlined.

Method	Target Accuracy (%)	Concept Accuracy (%)	Target Accuracy AUC	Concept Accuracy AUC	Training time (s)
CBM	67.3973 ± 0.5722	$\mathbf{94.9403} \pm 0.1059$	0.9551 ± 0.0000	0.9825 ± 0.0000	7204 ± 247
SCBM	65.5173 ± 0.8539	94.4569 ± 0.1328	0.9671 ± 0.0001	$\mathbf{0.9870} \pm 0.0000$	8134 ± 767
PSCBM	$\mathbf{68.3983} \pm 0.1992$	$\underline{94.9285} \pm 0.0206$	0.9680 ± 0.0003	0.9859 ± 0.0001	740 ± 94
PSCBMi	$\underline{68.1970} \pm 0.1274$	$\underline{94.9026} \pm 0.0350$	$\mathbf{0.9704} \pm 0.0002$	0.9866 ± 0.0001	14084 ± 267

of Table 1 report the AUC scores. SCBM achieves the highest concept AUC, but both PSCBM variants outperform it on target AUC, with PSCBMi being the strongest overall. This highlights the benefit of training with interventions. Both PSCBM models also outperform the regular CBM on both metrics, showing that adding a covariance matrix post-hoc improves intervention performance. For a more detailed comparison, we examine the intervention trajectories under the uncertainty policy in Figure 2. We additionally include the corresponding results when selecting concepts with a random policy in Appendix F. Both PSCBM variants consistently outperform CBM, with the intervention-trained variant performing best. However, for a few interventions, PSCBM without interventions during training lags behind SCBM, indicating that joint training of the covariance is more effective. PSCBM with interventions surpasses SCBM on target accuracy after about 20 interventions, though it does not reach SCBM's performance on concept accuracy. We note that SCBM could also be trained with interventions, which might further boost its results. It is clear from these that adding a covariance on a trained CBM can significantly improve its performance.

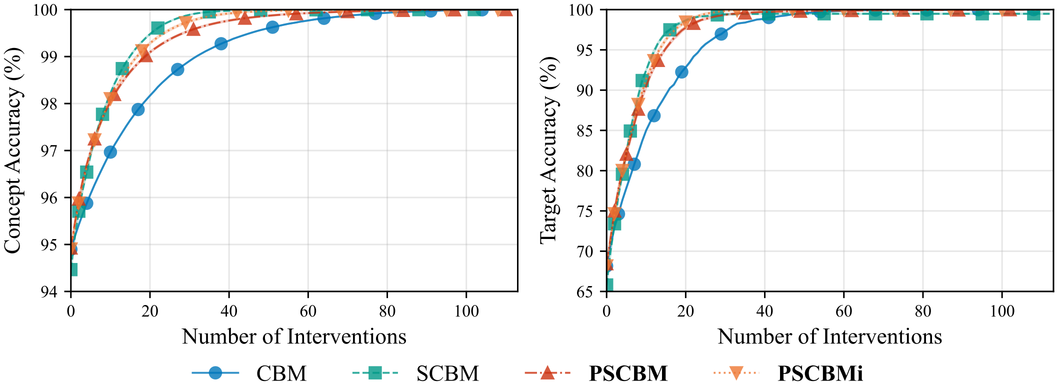

Figure 2: Intervention curves of Concept and Target Accuracy for PSCBM models and baselines when concept uncertainty policy is used. Confidence intervals are thinner than the lines.

4 Conclusion

In this paper, we introduce PSCBM, a lightweight extension to CBMs that models concept dependencies via a multivariate normal distribution. Crucially, PSCBM does not require retraining the full model: it only adds a compact covariance module. This makes it far less computationally demanding than training an SCBM from scratch, while retaining the benefits of modeling concept dependencies. We proposed two training procedures for the covariance module-optimizing the loss without interventions, and optimizing it after intervening on a random subset of concepts. Both variants outperform standard CBMs in test-time accuracy, and under interventions they adapt significantly faster than regular CBMs, though not as rapidly as a fully retrained SCBM. Importantly, training-time interventions improve intervention efficiency without harming accuracy when no interventions are in place, which demonstrates the efficacy of our approach.

Apart from its lightweight nature and efficacy, PSCBM also guarantees compatibility with the original CBM. By disabling the covariance module, PSCBM can revert to identical predictions as the baseline CBM. This property is particularly valuable in regulated domains such as healthcare, where a CBM may already have undergone approval (e.g., FDA testing). In such cases, retraining an SCBM might be prohibited or undermine user trust, whereas PSCBM preserves the validated predictions while still enabling stronger interventions when permitted.

The main limitation of our study is its evaluation on a single dataset; extending the analysis to additional benchmarks will be important for robustness. Future work should explore richer training-with-interventions schemes (e.g., varying the number of intervened concepts or using non-random policies), and investigate their applicability in regular SCBMs. Nonetheless, our results show that when retraining from scratch is infeasible, augmenting an existing CBM with a learned covariance matrix provides a simple and efficient way to substancially improve intervention effectiveness.

Acknowledgements

SL and MV are supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number MB22.00047. EP is supported by a fellowship from the ETH AI Center and received funding from the grant #2021-911 of the Strategic Focal Area "Personalized Health and Related Technologies (PHRT)" of the ETH Domain (Swiss Federal Institutes of Technology).

References

J. Ansel, E. Yang, H. He, N. Gimelshein, A. Jain, M. Voznesensky, et al. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, volume 2, pp. 929–947, 2024.

K. Chauhan, R. Tiwari, J. Freyberg, P. Shenoy, and K. Dvijotham. Interactive concept bottleneck models. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 37, pp. 5948–5955, 2023.

F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608, March 2017. doi: 10.48550/arXiv.1702.08608.

M. Espinosa Zarlenga, P. Barbiero, G. Ciravegna, G. Marra, F. Giannini, M. Diligenti, et al. Concept embedding models: Beyond the accuracy-explainability trade-off. In Advances in Neural Information Processing Systems, volume 35, pp. 21400–21413, 2022.

M. Espinosa Zarlenga, K. Collins, K. Dvijotham, A. Weller, Z. Shams, and M. Jamnik. Learning to receive help: Intervention-aware concept embedding models. In Advances in Neural Information Processing Systems, volume 36, 2024.

M. Havasi, S. Parbhoo, and F. Doshi-Velez. Addressing leakage in concept bottleneck models. In Advances in Neural Information Processing Systems, 2022.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 2015.

P. W. Koh, T. Nguyen, Y. S. Tang, S. Mussmann, E. Pierson, B. Kim, and P. Liang. Concept bottleneck models. In H. D. III and A. Singh (eds.), Proceedings of the 37th International Conference on Machine Learning, volume 119, pp. 5338–5348. PMLR, 2020.

Sonia Laguna, Ričards Marcinkevičs, Moritz Vandenhirtz, and Julia Vogt. Beyond concept bottleneck models: How to make black boxes intervenable? Advances in neural information processing systems, 37:85006–85044, 2024.

Sonia Laguna, Katarzyna Kobalczyk, Julia E Vogt, and Mihaela Van der Schaar. Interpretable reward modeling with active concept bottlenecks. ICLR 2025 Workshop: PRAL, 2025.

Z. C. Lipton. The mythos of model interpretability. Communications of the ACM, 61(10):35–43, June 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. International Conference on Learning Representations, 2019.

A. Mahinpei, J. Clark, I. Lage, F. Doshi-Velez, and W. Pan. Promises and pitfalls of black-box concept learning models. 38 th International Conference on Machine Learning, 2021.

Mikael Makonnen, Moritz Vandenhirtz, Sonia Laguna, and Julia E Vogt. Measuring leakage in concept-based methods: An information theoretic approach. ICLR 2025 Workshop: XAI4Science, 2025.

A. Margeloiu, M. Ashman, U. Bhatt, Y. Chen, M. Jamnik, and A. Weller. Do concept bottleneck models learn as intended? Workshop paper at ICLR 2021, 2021.

T. Oikarinen, S. Das, L. M. Nguyen, and T.-W. Weng. Label-free concept bottleneck models. In The 11th International Conference on Learning Representations, 2023.

A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, et al. Learning transferable visual models from natural language supervision. In International Conference on Machine Learning, pp. 8748–8763, 2021.

S. Shin, Y. Jo, S. Ahn, and N. Lee. A closer look at the intervention procedure of concept bottleneck models. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett (eds.), Proceedings of the 40th International Conference on Machine Learning, volume 202, pp. 31504–31520. PMLR, 2023.

S. Silvey. Statistical Inference. Taylor & Francis, 1975.

Nishad Singhi, Jae Myung Kim, Karsten Roth, and Zeynep Akata. Improving intervention efficacy via concept realignment in concept bottleneck models. In European Conference on Computer Vision, pp. 422–438. Springer, 2024.

Moritz Vandenhirtz, Sonia Laguna, Ričards Marcinkevičs, and Julia E. Vogt. Stochastic concept bottleneck models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), Advances in Neural Information Processing Systems, volume 37, pp. 51787–51810. Curran Associates, Inc., 2024.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset, 2011. Dataset.

M. Yuksekgonul, M. Wang, and J. Zou. Post-hoc concept bottleneck models. In The 11th International Conference on Learning Representations, 2023.

Mateo Espinosa Zarlenga, Zohreh Shams, Michael Edward Nelson, Been Kim, and Mateja Jamnik. Tabcbm: Concept-based interpretable neural networks for tabular data. Transactions on Machine Learning Research.

A Related Work

Concept Bottleneck Models Concept Bottleneck Models (CBM) have originally been proposed by Koh et al. (2020). The idea is to transform a regular neural network into an interpretable model by converting one of its layers into a concept bottleneck, where each neuron corresponds to a high-level concept that can be interpreted by a human. Such a model can be framed as a composite function $f(\boldsymbol{x}) = g(h(\boldsymbol{x}))$, where $h(\cdot)$ is a concept predictor that calculates the concepts $\hat{\boldsymbol{c}}$, and $g(\cdot)$ is the target predictor that predicts the final label from the concepts. Since all information that goes from the input to the output passes through the concept bottleneck, the network's prediction should be well explained by the concepts. Koh et al. (2020) use what has been called *soft* concept encoding: the target predictor uses real-valued concept logits or probabilities produced by the encoder. However, it has been shown that this method is prone to information leakage, where undesired information can pass through the bottleneck (Margeloiu et al., 2021; Mahinpei et al., 2021; Makonnen et al., 2025). As a remedy, Havasi et al. (2022) propose *hard* concepts: the concept encoder predicts concept probabilities, which parametrize Bernoulli distributions, from which binary inputs for the target predictor are sampled. Additionally, they introduce in their work a side channel which allows the target predictor to use information that cannot be expressed in terms of the high-level concepts. This modification can increase target accuracy if the concept set is incomplete. An alternative approach to modeling the concept bottleneck has been proposed by Espinosa Zarlenga et al. (2022) who introduce Concept Embedding Models (CEM), where each concept is modeled not by a single neuron but instead by a pair of learnable vector embeddings corresponding to the presence or absence of a concept in the input. This allows passing information not captured by the concepts through the bottleneck without the need for a side channel. Several methods have been proposed to transform regular models into Concept Bottleneck Models. Yuksekgonul et al. (2023) propose a method to convert one layer of a neural network trained without concepts into a concept bottleneck by using only a small concept-annotated subset. Oikarinen et al. (2023) demonstrate that concepts can also be generated with the help of a Vision-Language Model such as CLIP (Radford et al., 2021), reducing the human effort associated with annotating data. This work focuses on vision tasks but can be further extended to other domains as applications of CBMs have been explored as well in the context of text (Laguna et al., 2025) or tabular data (Zarlenga et al.).

Concept interventions A particular advantage of CBMs is the possibility of affecting downstream predictions by modifying incorrectly predicted concepts. This process is called a concept intervention. Koh et al. (2020) only consider interventions on randomly selected concepts. On the other hand, Chauhan et al. (2023); Shin et al. (2023) propose and evaluate more efficient concept selection policies, such as choosing the concept with the highest associated prediction uncertainty, that is, with probability closest to 0.5. This policy allows reducing target prediction error by half after 12 concept interventions on average, while with random concept selection an average of 43 interventions is necessary to achieve it. In Espinosa Zarlenga et al. (2022), a regularization technique called RandInt is introduced, whereby randomly selected concepts are set to their true value during training. This is then shown to increase the model's responsiveness to test-time concept interventions. As an alternative for an explicit concept selection policy, Espinosa Zarlenga et al. (2024); Singhi et al. (2024) parameterize it by a neural network which can be trained jointly with the model or in a post-hoc manner. A further extension of the interventions framework is done by Laguna et al. (2024), introducing a method for intervening on a black-box model without altering its architecture. They also define a measure of intervenability and show that fine-tuning a model for this quantity can improve the performance of interventions.

Modeling concept dependencies Since correlated concepts may require multiple interventions, several methods model dependencies. Havasi et al. (2022) use an autoregressive structure, by which the value of every concept depends on the previous concepts. It increases the model's accuracy, but has the disadvantage that concepts have to be evaluated sequentially, which increases evaluation time. Singhi et al. (2024) augment the CBM with a neural network that is trained to update all concepts in response to an intervention made to one of them. This can be combined with the trainable concept selection policy mentioned above. These elements can be either trained jointly with the full model or added to an existing CBM post-hoc. However, unlike our method, it does not provide an *explicit* representation of concept correlations. Finally, Vandenhirtz et al. (2024) introduce Stochastic Concept Bottleneck Models (SCBM), in which concept correlations are captured by a multivariate normal distribution. It requires predicting not only the expected value of the concepts, but also a covariance matrix. A special advantage of this model is the explicit representation of concept dependencies, which is used in an intervention strategy based on the distribution's confidence region.

B Comparing Different Intervention Strategies and Policies

In Section 2, we decomposed a concept intervention into the selection of concepts for intervention and updating their values. The former part we denote as *intervention policy* and the latter - as *intervention strategy*. In the following we describe and evaluate different possible choices. We consider two intervention policies: *random* concept selection (as in Koh et al. (2020)) and *concept uncertainty* policy, which selects the concept whose predicted probability is closest to 0.5 (that is, the most uncertain concept). For intervention strategies we consider four options:

1. *Empirical Percentile Strategy* sets the value of the intervened-on concept to the 5th (if it is 0) or 95th (if it is 1) percentile of the empirical distribution of concept predictions made by the model. Koh et al. (2020) use it for intervening in CBM with soft concept encoding.
2. *Hard Strategy* introduced by Havasi et al. (2022) sets concept probabilities to 0 or 1. In the case of PSCBM, where concepts are represented by their logits, one cannot directly use the logits of 0-1 probabilities, as they are infinite. Instead, we set concepts to the logits of ε for absent concepts and $1 - \varepsilon$ for present concepts, where ε is a small positive number.
3. *Simple Percentile Strategy* is a softened version of the hard strategy, where concept probabilities of absent or present are set to 0.05 or 0.95, respectively.
4. *Confidence Region Strategy* is a strategy that directly uses the concepts' multivariate normal distribution. If a subset $\mathcal{S} \subset \{1, \dots \mathcal{C}\}$ of concepts has been selected for intervention, the updated concept logits $\eta'_{\mathcal{S}}$ are calculated by solving the following optimization problem:

$$\begin{aligned} \eta'_{\mathcal{S}} &= \arg\max_{\eta_{\mathcal{S}}} \log p(\boldsymbol{c}_{\mathcal{S}} | \eta_{\mathcal{S}}), \\ \text{s.t.} &- 2 \left(\log p(\eta_{\mathcal{S}} | \mu_{\mathcal{S}}, \boldsymbol{\Sigma}_{\mathcal{S},\mathcal{S}}) - \log p(\boldsymbol{\mu}_{\mathcal{S}} | \boldsymbol{\mu}_{\mathcal{S}}, \boldsymbol{\Sigma}_{\mathcal{S},\mathcal{S}}) \right) \leq \chi^2_{d,1-\alpha} \\ &\quad \eta_i - \mu_i \geq 0 \quad \text{if } c_i = 1, \quad \forall i \in \mathcal{S} \\ &\quad \eta_i - \mu_i \leq 0 \quad \text{if } c_i = 0, \quad \forall i \in \mathcal{S} \end{aligned} \tag{1}$$

   where $\mathcal{S}$ is the subset of concepts that have been selected for intervention, $\boldsymbol{\mu}_{\mathcal{S}}$ and $\boldsymbol{\Sigma}_{\mathcal{S},\mathcal{S}}$ are the predicted concept mean and covariance of this concept subset, c_i are the concept values selected by the user, d is the dimensionality of the distribution, i.e. the number of concepts, and α is the requested confidence of the confidence region. In this problem we seek to find concept logits $\eta'_{\mathcal{S}}$ which maximize the log-probability of intervened concept values, and stay within the α-confidence region of the predicted distribution, that is, the model's prediction shouldn't be completely disregarded (first constraint). The log-probability of this distribution, after being multiplied by -2, asymptotically follows the χ^2 distribution (Silvey, 1975), which gives rise to the first inequality. The other two inequalities follow from the requirement that the new concept logits shouldn't move away from the ground truth: if $c_i = 1$, η_i should increase, and if $c_i = 0$, it should decrease. This last strategy can only be used in the context of SCBM, while the three others can be used by both SCBM and CBM.

In Table 2 we report the AUC for concept and target accuracy for all models and all intervention policies and strategies that can be used with them. The strategy based on confidence region cannot be used with a regular CBM, because it requires the covariance matrix.

Interestingly, there is no single best strategy. The *Hard* one typically produces the best or close-to-the-best results for target accuracy, but the *Confidence Region* strategy is often better on concept accuracy, especially with models using an amortized covariance. Hence, the choice of the best strategy is not an obvious one, and we encourage the reader to evaluate different policies in relation to the specific data and setup at hand.

Table 2: Comparison of AUC for all combinations of strategy and policy pairs for all baseline and PSCBM models. The values are normalized to lie in the interval $[0, 1]$. For each model, policy pair, the best value is **bold**, and those lying within a standard deviation are underlined. Likewise, the selected strategy is also **bold** and the one following up is underlined.

Method	Policy	Strategy	Target Accuracy AUC	Concept Accuracy AUC
CBM	Concept Uncertainty	**Hard**	**0.9551** ± 0.0000	**0.9825** ± 0.0000
		Simple Percentile	0.9513 ± 0.0000	0.9825 ± 0.0000
		Empirical Percentile	0.9550 ± 0.0000	0.9798 ± 0.0002
	Random	**Hard**	**0.8927** ± 0.0003	0.9658 ± 0.0000
		Simple Percentile	0.8847 ± 0.0015	**0.9659** ± 0.0001
		Empirical Percentile	0.8927 ± 0.0007	0.9642 ± 0.0001
SCBM (global covariance)	Concept Uncertainty	**Hard**	**0.9671** ± 0.0001	**0.9870** ± 0.0000
		Simple Percentile	0.9453 ± 0.0056	0.9835 ± 0.0002
		Empirical Percentile	0.9570 ± 0.0060	0.9864 ± 0.0001
		Confidence Region	0.9630 ± 0.0023	0.9866 ± 0.0001
	Random	**Hard**	**0.9085** ± 0.0010	0.9697 ± 0.0001
		Simple Percentile	0.8390 ± 0.0029	0.9147 ± 0.0002
		Empirical Percentile	0.9013 ± 0.0041	0.9637 ± 0.0004
		Confidence Region	0.8963 ± 0.0014	**0.9718** ± 0.0004
SCBM (amortized covariance)	Concept Uncertainty	Hard	0.9646 ± 0.0000	0.9860 ± 0.0000
		Simple Percentile	0.9653 ± 0.0001	**0.9873** ± 0.0000
		Empirical Percentile	0.9653 ± 0.0002	0.9863 ± 0.0000
		Confidence Region	**0.9664** ± 0.0001	0.9871 ± 0.0000
	Random	**Hard**	**0.9234** ± 0.0005	0.9785 ± 0.0000
		Simple Percentile	0.9099 ± 0.0019	0.9509 ± 0.0005
		Empirical Percentile	0.9230 ± 0.0007	0.9782 ± 0.0001
		Confidence Region	0.9204 ± 0.0008	**0.9789** ± 0.0001
PSCBM (global covariance)	Concept Uncertainty	**Hard**	**0.9680** ± 0.0003	**0.9859** ± 0.0001
		Simple Percentile	0.9680 ± 0.0002	0.9859 ± 0.0001
		Empirical Percentile	0.9665 ± 0.0005	0.9849 ± 0.0001
		Confidence Region	0.9657 ± 0.0002	0.9853 ± 0.0001
	Random	**Hard**	**0.9081** ± 0.0014	0.9705 ± 0.0002
		Simple Percentile	0.9078 ± 0.0012	0.9705 ± 0.0002
		Empirical Percentile	0.8817 ± 0.0021	0.9648 ± 0.0004
		Confidence Region	0.9070 ± 0.0014	**0.9716** ± 0.0001
PSCBM (amortized covariance)	Concept Uncertainty	Hard	0.9665 ± 0.0018	0.9859 ± 0.0004
		Simple Percentile	**0.9665** ± 0.0018	**0.9859** ± 0.0004
		Empirical Percentile	0.9591 ± 0.0025	0.9832 ± 0.0007
		Confidence Region	0.9661 ± 0.0013	0.9855 ± 0.0002
	Random	Hard	0.9129 ± 0.0018	0.9638 ± 0.0046
		Simple Percentile	0.9127 ± 0.0016	0.9638 ± 0.0045
		Empirical Percentile	0.9046 ± 0.0017	0.9711 ± 0.0008
		Confidence Region	**0.9140** ± 0.0011	**0.9744** ± 0.0004
PSCBMi (global covariance)	Concept Uncertainty	Hard	0.9708 ± 0.0001	**0.9857** ± 0.0000
		Simple Percentile	**0.9708** ± 0.0002	0.9857 ± 0.0000
		Empirical Percentile	0.9036 ± 0.0006	0.9666 ± 0.0001
		Confidence Region	0.9379 ± 0.0003	0.9767 ± 0.0000
	Random	Hard	0.8846 ± 0.0014	**0.9552** ± 0.0003
		Simple Percentile	**0.8851** ± 0.0007	0.9552 ± 0.0002
		Empirical Percentile	0.7797 ± 0.0021	0.9321 ± 0.0002
		Confidence Region	0.8321 ± 0.0013	0.9475 ± 0.0001

C Pseudocode for Interventions Training

In Algorithm 1 we present the pseudocode for one iteration of the intervention-aware training procedure.

Algorithm 1 Loss calculation for one iteration of training with random interventions.

```
Inputs:
    $g$ (target predictor)
    $\mathcal{L}$ (loss function)
    $L$ (number of concepts for intervention)
    $N$ (number of random interventions per datapoint)
    $\tau$ (intervention strategy, which modifies concept values)
    $(\boldsymbol{c}, y)$ (concept and target labels)
    $(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$ (predicted expected value and covariance of concepts)
Algorithm:
    $l = 0$ (Initialize the loss to 0)
for i in 1...N do
        $S = \text{Random}(L)$ (Randomly select $L$ concepts for intervention)
        $\boldsymbol{\mu}', \boldsymbol{\Sigma}' = \tau(S, \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$ (Use strategy $\tau$ to update the concept distribution)
        $\eta' \sim \mathcal{N}(\boldsymbol{\mu}', \boldsymbol{\Sigma}')$ (Sample concept logits from the updated distribution)
        $\boldsymbol{c}' = \sigma(\eta')$ (Pass concept logits through a sigmoid to obtain a vector of probabilities)
        $\boldsymbol{c}_{1:M} = \text{Bern}(\boldsymbol{c}')$ (Sample $M$ concept vectors from the Bernoulli distribution defined by $\boldsymbol{c}'$)
        $y' = \frac{1}{M} \sum_{m=1}^{M} g(\boldsymbol{c}^{(m)})$ (Calculate the target)
        $l = l + \mathcal{L}(y', y, \boldsymbol{c}', \boldsymbol{c})$ (Loss after intervention)
end for
return $\frac{l}{N}$ (Average loss)
```

D Results with amortized covariance

In this appendix, we show additional results for the studied models using an *amortized* covariance matrix instead of a globally learnt one as in the main manuscript. The amortized covariance, as introduced in SCBM Vandenhirtz et al. (2024), makes the $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ learnt per instance, i.e. amortized by the input $\boldsymbol{x}$. We show results for CBM and SCBM as baselines and for a PSCBM trained without interventions. In Table 3 we show test-time results for target and concept accuracy, as well as target and concept accuracy AUC for interventions with probability uncertainty policy. In Figure 3, we show the respective intervention curves. We observe that without interventions, the performance of PSCBM is better than thus of the baselines. With interventions, it scores the highest on target accuracy AUC, and on par with the rest on concept accuracy AUC. Intervention curves confirm these results. However, in the curve for target accuracy, one can observe that after the first few interventions, the SCBM improved faster than the PSCBM. This is important because in a real-world scenario it is expected that only a few concept interventions would be made.

Table 3: Test-time target and concept accuracy without interventions and their AUCs during interventions in PSCBM and baselines with an *amortized* covariance matrix. AUC is normalised to stay within the interval [0,1]. We report the mean and standard deviation over three runs. The best scores for each metric are emphasized with **bold** font, and those within a standard deviation from the best one are underlined.

Method	**Target Accuracy (%)**	**Concept Accuracy (%)**	**Target Accuracy AUC**	**Concept Accuracy AUC**	**Training time (s)**
CBM	67.3973 ± 0.5722	$\underline{94.9403} \pm 0.1059$	$\underline{0.9551} \pm 0.0000$	0.9825 ± 0.0000	7204 ± 247
SCBM	$\underline{68.6342} \pm 0.3790$	$\underline{94.9190} \pm 0.0074$	0.9646 ± 0.0000	$\mathbf{0.9860} \pm 0.0000$	8130 ± 1073
PSCBM	$\mathbf{68.6611} \pm 0.2067$	$\mathbf{94.9605} \pm 0.0639$	$\mathbf{0.9665} \pm 0.0018$	$\underline{0.9859} \pm 0.0004$	670 ± 55

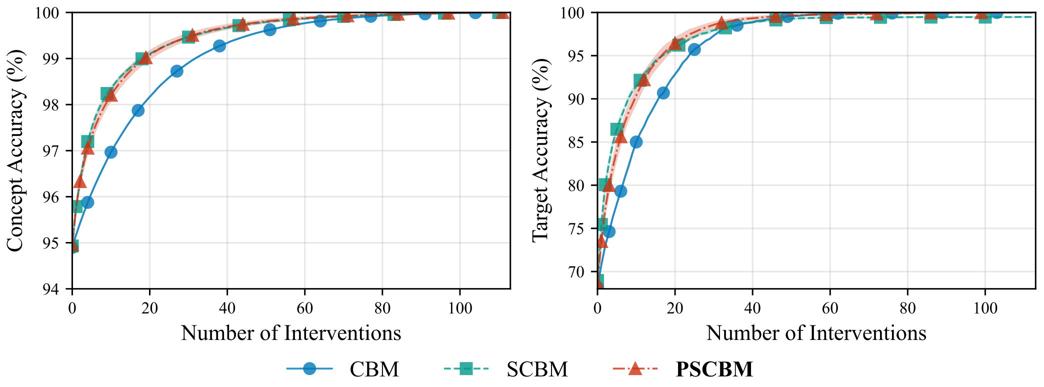

Figure 3: Concept and Target Accuracy for amortized PSCBM models and baselines when Concept Uncertainty Policy is used. The shadings represent the standard deviation.

E Implementation Details

All models have been implemented in PyTorch (Ansel et al., 2024) and optimized using Adam (Kingma & Ba, 2015). To apply weight decay, we used the optimizer AdamW, which decouples weight decay from gradient computation (Loshchilov & Hutter, 2019). All models use a ResNet-18 encoder. The remaninig parts including the concept mean and covariances, and the target predictor are all linear layers. All baselines have been trained with three random initializations. All PSCBM versions have been evaluated with each of the three CBMs with three random seeds, which amounts to a total number of nine evaluations. The computations have been done on a cluster composed mostly of NVIDIA GeForce RTX 2080 GPUs, and every job used a single GPU and two CPUs. When we train PSCBM with interventions, we use 20 random masks per datapoint per epoch, and found that a larger number does not lead to better performance. We intervene on 20% of the total number of concepts, and their values at training time are set according to the *Hard* Strategy, introduced in Appendix B. We performed hyperparameter optimization with respect to learning rate scheduling, initial learning rate, and weight decay factor. In Table 4 we report the optimal values based on validation loss after the last training epoch. It should be noted that PSCBMa is not very sensitive to weight decay and that for PSCBMg the advantage of step learning rate over cosine schedule was minimal, albeit statistically significant. Similarly, for CBM and for both SCBM variants, all four tested hyperparameter combinations achieved similar performance.

Table 4: Optimal hyperparameter values for each model.

Model	Learning rate scheduler	Initial learning rate	Weight decay
CBM	Step-wise	10^{-4}	0.1
SCBMg	Step-wise	10^{-4}	0.1
SCBMa	Step-wise	10^{-4}	0.1
PSCBMg	Step-wise	10^{-3}	1
PSCBMa	Cosine	10^{-4}	0
PSCBMg$_\mathrm{i}$	Cosine	10^{-4}	4

F Results with Random Policy

In Figure 4 we show the intervention curves when concepts for intervention are chosen randomly. One can observe generally lower improvement rates in comparison to the selection policy based on concept uncertainty. On concept accuracy, both PSCBM variants outperform SCBM. On target accuracy, they outperform the regular CBM but fall slightly behind the SCBM.

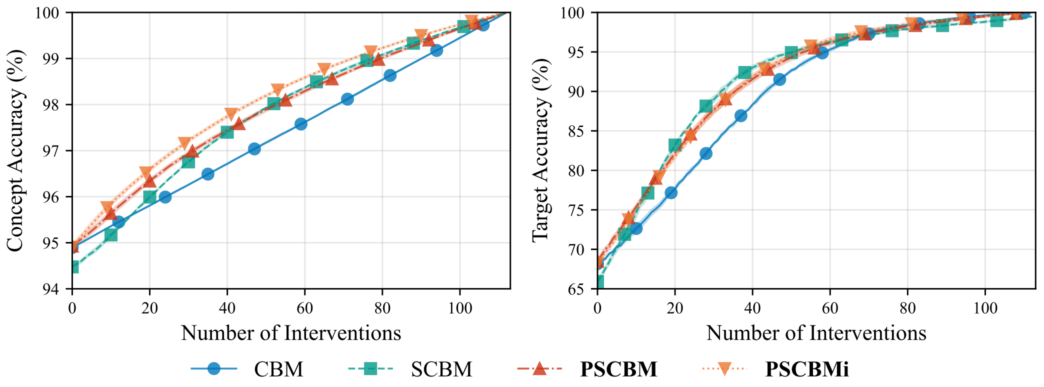

Figure 4: Concept and Target Accuracy for PSCBM models and baselines when Random policy is used. The shadings represent the standard deviation.