UNIVERSITY OF
CAMBRIDGE
Department of Computer
Science and Technology

Ain't Nobody Got Time For That: Budget-aware Concept Intervention Policies

Thomas Yuan

Downing College

May 2024

Submitted in partial fulfillment of the requirements for the
Computer Science Tripos, Part III

Total page count: 23

Main chapters (excluding front-matter, references and appendix): 10 pages (pp 9–18)

Main chapters word count: 467

Methodology used to generate that word count:

```
$ make wordcount
gs -q -dSAFER -sDEVICE=txtwrite -o - \
   -dFirstPage=6 -dLastPage=11 report-submission.pdf | \
egrep '[A-Za-z]{3}' | wc -w
467
```

Declaration

I, Thomas Yuan of Downing College, being a candidate for the Computer Science Tripos, Part III, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

Signed: Thomas Yuan

Date: May 13, 2024 -

Abstract

With the developments made in Concept Bottleneck Models, finding optimal interventions for these models to maximize their performance under a given budget is an important issue, as it allows us to maximize the performance of these interpretable models while minimizing the costs related to expert interventions. This project investigates solving this problem using a new approach that combines Reinforcement Learning with surrogate models, with the goal of maximizing intervention performance while maintaining similar performance to the original model when no interventions are performed.

Compared to previous approaches that only use greedy policies, such as heuristic-based methods or learning a Machine Learning model for a greedy policy, the main idea of this project is to use Reinforcement Learning to learn a non-greedy policy to maximize the performance gain from interventions over a budget rather than at each step. This is done by modelling the problem of deciding what concept to intervene on as a Reinforcement Learning problem to maximize the final performance, using surrogate models that model the distribution of concepts as intermediate rewards. When used in conjunction with methods to increase sensitivity of models to interventions, this project successfully develops a novel RL-based method that outperforms existing greedy intervention policies for different budgets, while maintaining similar un-intervened performance.

This project proposes the idea of learning non-greedy intervention policies for Concept Bottleneck Models and incorporates budgets into the problem setting, and shows that different budgets can have different optimal intervention trajectories where non-greedy policies outperform greedy policies. This project also successfully shows that Reinforcement Learning can be applied to this approach to generate more optimal intervention policies that lead to better performance compared to existing greedy approaches.

Acknowledgements

This project would not have been possible without the wonderful support of my lovely supervisors Mateo Espinosa Zarlenga, Dr Mateja Jamnik and Dr. Zohreh Shams. I would also like to thank my friends and family for their support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Regular supervised Machine Learning (ML) models learn to predict the labels of inputs. Concept Bottleneck Models [3] (CBMs) and Concept Embedding Models [5] (CEMs) are ML models designed to increase the interpretability of model predictions by decomposing a model into two submodels, splitting the original process into predicting a set of human-interpretable concepts / features present in the input, then predicting the label using these concepts. Since these concepts are human-interpretable, it is much more easier to understand the reasoning behind the predicted labels, thus mitigating some of the potential dangerous downsides associated with using ML models as "black-box" models, especially in fields where these predictions can have significant consequences to human life, such as medicine, criminal justice, autonomous vehicles, etc.

During inference time, professionals can intervene on CBMs and CEMs by correcting the predicted concepts leading to more accurate predicted labels. Due to the costs associated with performing such an intervention, the question of finding what concepts to intervene on in order to maximize the accuracy of the model becomes an important research question. Past methods have mainly focused on greedy approaches, this includes using heuristic-based approaches such as intervening on concepts that minimize the model's uncertainty [2], or ML-based greedy approaches that learn a intervention policy model like in IntCEMs [6]. Compared to existing work, we investigate the possibility of learning non-greedy policies where we model the problem of deciding what concepts to intervene on as a Reinforcement Learning [4] problem. We model the costs associated with interventions and budgets for interventions, which are constraints present in real life applications of CBMs, and set the main research question to be determining the concepts to intervene on for different budgets for a given CEM to maximize its performance.

To answer the above question, we first investigate the differences between greedy and non-greedy policies, proving that non-greedy algorithms can outperform their greedy counterparts. We then build a Reinforcement Learning agent that learns to select the next concept to intervene in order to maximize the model's performance for a given budget.

To guide the agent throughout the intervention process, we utilize surrogate models that model the conditional distribution of concepts. A detailed description of these models are in Section 4.3. These surrogate models guide the RL agent throughout the intervention process by rewarding the agent for intervening on concepts that lead to higher probabilities of intervened concepts, and provide auxiliary information such as which of the unintervened concepts are more likely to be correct in order for the agent to make more informed decisions. We augment CEMs with this RL agent and a pretrained surrogate model to form RLCEM, where the CEM and the RL agent is trained simultaneously, which helps learn a non-greedy policy and increase the sensitivity of the CEM to these interventions.

The results show that such a CEM outperform existing model when intervened using the learnt policy, while simultaneously achieving similar performance under the absence of interventions. This project successfully demonstrates the power of using non-greedy policies over greedy policies, and how a non-greedy policy can be learnt that outperforms existing greedy policies using Reinforcement Learning, showing that Reinforcement Learning is a viable approach for learning intervention policies.

Chapter 2

Background

2.1 CBM

Concept Bottleneck Models (CBMs), initially proposed by Koh et al. [3], are a class of models that consist of a model g that learns a mapping from input $\mathbf{x}$ to a concept vector $\mathbf{c} = g(\mathbf{x})$, where $\mathbf{c}$ is a multi-hot encoding of the concepts present in the input, and a model f that learns a mapping from such concepts vector c to the output label $\mathbf{y} = f(\mathbf{c})$, as shown in Figure 2.1. These types of model can be created by simply adding a new layer in traditional models with the same number of neurons as the number of concepts, where this layer is referred to as the "CBM Bottleneck". Henceforth g is referred to as the "concept predictor $\mathbf{x} \to \mathbf{c}$ model" and f as the "label $\mathbf{c} \to \mathbf{y}$ predictor model". Training such a model requires a dataset of inputs $\mathbf{x}$ annotated with the corresponding concepts $\mathbf{c}$ and labels $\mathbf{y}$.

CBMs allow for interventions, which are using experts to correct the intermediate concepts predicted by the $\mathbf{x} \to \mathbf{c}$ model to improve the performance of the $\mathbf{c} \to \mathbf{y}$ model.

2.1.1 Training CBMs

There are several different ways to train a CBM. If we let the concept loss L_{concept} be a loss function that measures the discrepancy between the predicted concepts $\hat{\mathbf{c}}$ and the actual concepts $\mathbf{c}$, and similarly the label loss L_{label} measuring the discrepancy between the predicted concepts $\hat{\mathbf{y}}$ and the actual concept $\mathbf{y}$, both losses as illustrated in Figure 2.1. There are the following ways to train a CBM as proposed in [3].

1. Independent: Training the two models independently by minimizing $L_{\text{concept}}(g(\mathbf{x}), \mathbf{c})$ and $L_{\text{label}}(f(\mathbf{c}), \mathbf{y})$ independently.

2. Sequential: Training the models one by one, first learning $\hat{g}$ by minimizing

   $L_{\text{concept}}(g(\mathbf{x}), \mathbf{c})$, then learning f by minimizing $L_{\text{label}}(f(\hat{\mathbf{g}}(\mathbf{x})), \mathbf{y})$

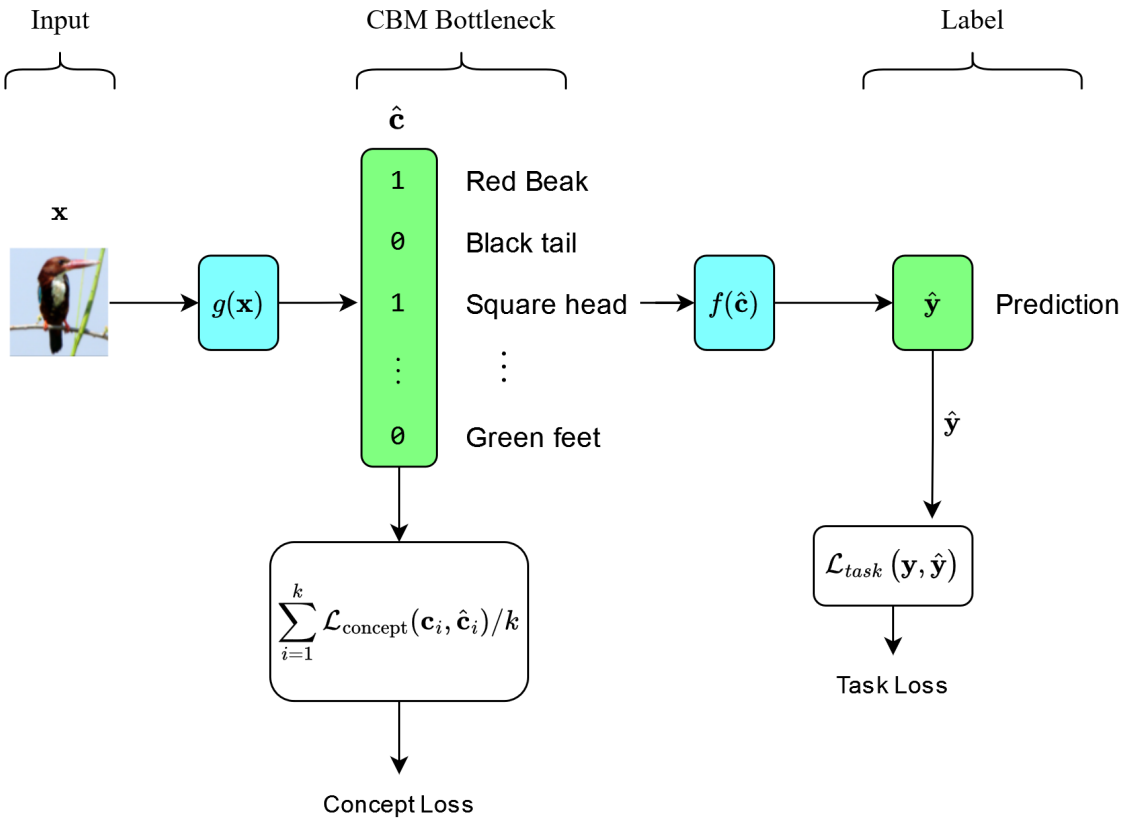

Figure 2.1: The CBM Architecture [3].

3. Joint: The model is trained via a weighted sum of the losses given by
   $\lambda_{\text{concept}} L_{\text{concept}}(g(\mathbf{x}), \mathbf{c}) + \lambda_{\text{label}} L_{\text{label}}(f(g(\mathbf{x})), \mathbf{y})$
   such that both losses are minimised simultaneously.

It has been shown experimentally that while the joint models perform the best without interventions, followed by the sequential model, and the independent model performs the worst. This is because the sequential model allows the $\mathbf{c} \to \mathbf{y}$ model to learn a mapping from the concepts produced by the $\mathbf{x} \to \mathbf{c}$ model to label $\mathbf{y}$, where the concepts produced by the $\mathbf{x} \to \mathbf{c}$ model is often different from the true concepts, an underlying requirement for the independent model to perform well. Additionally, the joint model allows the $\mathbf{x} \to \mathbf{c}$ model to simultaneously learn to output a representation of concepts that allow for best performance of the $\mathbf{c} \to \mathbf{y}$ model [3].

When comparing performance under interventions, independent models outperform the two models. They are more sensitive to interventions and each successive intervention step leads to a bigger increase in performance compared to the other two, with better performances after the same number of interventions. The reason behind this is that the independent model learns a mapping from the true concepts to the label, whereas the other two learn a mapping from the predicted concepts to the label. Each intervention modifies the predicted concepts to be closer to the true concepts, which is what the $\mathbf{c} \to \mathbf{y}$ independent model is trained to do [3].

2.2 CEM

While CBMs proved to be useful in achieving machine learning models with high interpretability, they do not perform as well as traditional models that learn a direct mapping from input to labels. This is because the CBM label prediction model relies only on a set of human-interpretable concepts, which limits the performance of the model as traditional models can extract information outside of these concepts that are potentially not human-interpretable [5]. This is even more apparent if the dataset does not contain a complete set of concepts that cover all features present in the input, which is very common in real-life scenarios. As such, there is a trade-off between performance and interpretability, where researchers have developed methods such as extending the CBM bottleneck with a set of unsupervised neurons to increase accuracy at a cost of decreasing interpretability [1].

To overcome this trade-off, Concept Embedding Models (CEMs) were proposed by Zarlenga et al. [5], these are CBMs that further add an additional layer of learnable embeddings before the original bottleneck, learning two embedding vectors for each concept: one for when the concept is and is not present. The architecture of CEMs is shown in Figure **??**. An intermediate scoring function ϕ_i is learnt for each concept i, and the embedding assigned to the bottleneck is an interpolation of the two embeddings based on the scoring function predicting the possibility of the concept to be present.

This architecture also allows for interventions during run-time. By simply modifying the output of the scoring function to be that of the true concept, the bottleneck can be modified similarly to CBMs and improve the performance of the model. Additionally to increase the performance of CEMs, the authors utilised observations mentioned in Section 2.1, where models trained on the true concepts are more sensitive to interventions. They proposed RandInt, a method to randomly intervene during training with $\lambda_{\text{int}} = 0.25$ probability of intervening on a concept. They show that this effectively boosts the performance of the model under interventions during test time without notable effects to the performance without interventions [5].

CEMs successfully solves the trade-off problem between performance and interpretability, allowing for similar performance to traditional models while maintaining the interpretability, along with high concept accuracy. This is because the embedding structure for CEMs allow for encoding of more information in the concept representations and is more machine-interpretable, where the additional information in the bottleneck compared to scalar representations in CBMs lead to a better performing label predictor model. Additionally CEMs are still trained on the same set of human-interpretable concepts as CBM via a similar concept loss, which leads to high interpretability and good intervention performance. It has been shown experimentally that CEMs are able to provide better performance for concept-incomplete dataset tasks (where the concepts do not cover all features present in input), and these learnt concept embedding representations effectively represent the true concepts measured by an alignment score [5].

2.3 IntCEM

Building on top of CEMs, Zarlenga et al. [6] introduced Intervention-aware CEM (IntCEM), which are CEMs that are augmented with a learnable concept intervention policy model. IntCEMs' novelty lies in framing the problem of training a CEM and finding an intervention policy as a joint optimization problem by augmenting existing CEMs with a trainable intervention policy model ψ. This approach offers significant improvements in performance after interventions while maintaining similar performance without interventions. IntCEM achieves this because the intervention policy model learn a good intervention policy specific to the CEM, and the CEM also learns to be more sensitive to interventions by the model, through the introduction of an intervention loss L_{int} and task loss L_{task} for the intervened concepts.

During training, ψ first samples intervention logits for the next concept to intervene on, then a Cross Entropys loss L_{int} is used to compute the discrepancy with the output of a greedy optimal policy, found by searching over all concept to yield the concept that leads to the highest increase in model performance when intervened. This is a Behavioural Cloning [] approach where ψ learns to clone the behaviour of a greedy optimal policy.

As mentioned in Section 2.1, training using true concept labels increases the model's sensitivity to interventions, leading to better intervention performance. IntCEM incorporates this idea by computing task loss using the intervened concepts by ψ during training. Not only does this increase the model's sensitivity to interventions, it specifically increases the model's sensitivity to interventions sampled by ψ to further improve its intervention performance.

2.4 RL

Reinforcement Learning is a machine learning subfield that focuses on training agents to make sequential decisions in an environment. Contrary to supervised learning where the goal is to minimize the discrepancy between predicted outputs and true outputs, the goal of Reinforcement Learning is to maximize the cumulative rewards received by the agent by taking actions over time. The agent has access to observations which reflect the current state of the environment and can take actions to progress to different states until

Chapter 3

Related Work

3.1 CooP

3.2 Active Feature Acquisition with Generative Surrogate Models

Chapter 4

Design and implementation

4.1 Intervention Policies

4.2 Surrogate Models

4.3 RLCEM

4.4 Datasets

Chapter 5

Evaluation

5.1 Non-greedy policies

5.2 Surrogate Models

5.3 RLCEM Performance

5.4 Limitations

Chapter 6

Summary and conclusions

6.0.1 Future Work

Bibliography

[1] *Promises and Pitfalls of Black-Box Concept Learning Models*, volume 1, 2021.

[2] Kushal Chauhan, Rishabh Tiwari, Jan Freyberg, Pradeep Shenoy, and Krishnamurthy Dvijotham. Interactive concept bottleneck models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37:5948–5955, 06 2023.

[3] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5338–5348. PMLR, 13–18 Jul 2020.

[4] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018.

[5] Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Zohreh Shams, Frederic Precioso, Stefano Melacci, Adrian Weller, Pietro Lio, and Mateja Jamnik. Concept embedding models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[6] Mateo Espinosa Zarlenga, Katherine M. Collins, Krishnamurthy Dj Dvijotham, Adrian Weller, Zohreh Shams, and Mateja Jamnik. Learning to receive help: Intervention-aware concept embedding models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Appendix A

Dataset Details

Appendix B

Surrogate Model Details

Appendix C

Hardware Specifications

Appendix D

Training Graphs