arXiv:2506.05014v1 [cs.LG] 5 Jun 2025

Towards Reasonable Concept Bottleneck Models

Nektarios Kalampalikis* **Kavya Gupta** **Georgi Vitanov** **Isabel Valera**
Department of Computer Science of Saarland University, Saarbrücken, Germany

Abstract

In this paper, we propose **C**oncept **REA**soning **M**odels (CREAM), a novel family of Concept Bottleneck Models (CBMs) that: (i) explicitly encodes concept-concept (`C-C`) and concept-task (`C→Y`) relationships to enforce a desired model reasoning; and (ii) use a regularized side-channel to achieve competitive task performance, while keeping high concept importance. Specifically, CREAM architecturally embeds (bi)directed concept-concept, and concept to task relationships specified by a human expert, while severing undesired information flows (e.g., to handle mutually exclusive concepts). Moreover, CREAM integrates a black-box side-channel that is regularized to encourage task predictions to be grounded in the relevant concepts, thereby utilizing the side-channel only when necessary to enhance performance. Our experiments show that: (i) CREAM mainly relies on concepts while achieving task performance on par with black-box models; and (ii) the embedded `C-C` and `C→Y` relationships ease model interventions and mitigate concept leakage.

1 Introduction

Deep neural networks (DNNs) have become ubiquitous, permeating various aspects of our daily lives. However, their black-box decision-making processes hinder transparency, making it difficult for users to understand or trust their predictions. Interpretability in DNNs is crucial for building trustworthy AI systems [16, 29], particularly in high-stakes domains such as finance [16], healthcare [47], and autonomous systems [49]. To address these concerns, there is a growing interest in inherently interpretable models [39, 47, 48], such as concept-based models. Such models aim to explain their predictions through meaningful and human-understandable concepts [4, 11, 27, 40, 43, 63, 64].

Among these models, Concept Bottleneck Models (CBMs)[27] have emerged as a compelling paradigm. CBMs are neural networks that introduce an intermediate concept bottleneck layer where concepts (e.g., "Tops" and "Shoes" in Fig. 1) are explicitly learned and predicted before making task predictions, allowing for *more transparent model predictions* and *human intervention*. CBMs have been successfully applied to fields such as medical diagnosis [13], predictive maintenance [18], and vision-language tasks [62].

Despite their advantages, current CBM implementations suffer from several limitations. First, standard CBMs assume that the concepts are conditionally independent, and are unable to model relationships among them or between concepts and the task [15]. We call the ability to do so, *structured model reasoning*. The absence of such reasoning can lead to misleading explanations and spurious correlations, where models reach correct predictions for the wrong reasons [1, 36]. Second, real-world datasets may not capture all the relevant concepts required for task prediction, leading to a problem of *concept incompleteness* [63]. This in turn harms predictive performance [20, 32, 63]. Furthermore, even when concept predictions are correct, the model may exploit unintended information, known as *concept leakage*, to make task predictions in a way that bypasses the intended model reasoning pathways [32, 38]. This betrays the promise of interpretability and may deceive users into over-trusting the model [28, 36].

*Correspondence to:kalampalikis@cs.uni-saarland.de.

Preprint. Under review.

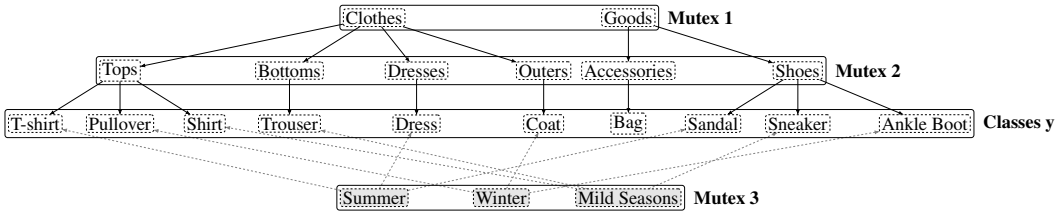


Figure 1: Reasoning graph for FMNIST from [52]. The white nodes represent concepts used in *iFMNIST*, while the gray nodes introduce additional seasonality-related concepts to form *cFMNIST*, addressing concept incompleteness. Concepts and classes within the solid boxes are mutually exclusive.

To address these limitations, we propose **C**oncept **REA**soning **M**odels (CREAM), a novel family of CBMs, which enforce a desired structured model reasoning in terms of explicit `C-C` and `C→Y` relationships. Such relationships are provided by a predefined reasoning graph that contains both hard constraints and probabilistic relationships. For example in Fig. 1, the hard constraints severs the edges from "Tops"→"Sneaker" to prevent irrelevant influence, and enforces mutual exclusivity between "Goods" and "Clothes" to match ontological structure. Meanwhile, probabilistic relationships allow information flow from "Goods"→"Shoes" when such dependencies are supported by training data. By incorporating these relationships into the model itself, CREAM ensures that predictions are grounded in interpretable and domain-aligned `C-C` and `C→Y` relationships, which both facilitates interventions and mitigates concept leakage, as shown in our experiments. Additionally, CREAM introduces a regularized side-channel, a black-box representation, that captures task-relevant information beyond the predefined concepts. Previous methods where a side-channel has been used, showed that it can undermine the use of the concept bottleneck [21]. In contrast, CREAM regularizes the side channel, thus encouraging the model to use it as a supplement when the concepts alone do not suffice to achieve high predictive performance. This design allows CREAM to balance task accuracy and interpretability, providing more transparent and controllable trade-offs between them.

2 Concept Reasoning Model

In this work, we propose *reasonable concept bottleneck models* that are guided, but not strictly constrained, by the `C-C` and `C→Y` relationships specified by domain experts. Unlike standard CBMs, which typically follow a rigid bipartite concept-to-task architecture, CREAM introduces a more expressive and structured framework that supports flexible, interpretable model reasoning while maintaining high predictive performance. To this end, CREAM as shown in Fig. 4, consists of three core components: i) a *representation splitter* which decomposes the backbone feature representation ($\mathbf{z}$) into a concept representation ($\mathbf{z}_C$) and a *side-channel* representation ($\mathbf{z}_Y$); ii) a *concept-concept block* that enforces expert-defined `C-C` relationships via a structured, possibly bidirected, dependency graph; and iii) a *concept-task block* that encodes the given `C→Y` reasoning while leveraging a regularized side-channel to boost task predictions.

At the core of CREAM lies a *model reasoning graph*, denoted as $G = (V, E)$, which encodes the given `C-C` and `C→Y` relationships. Here, the nodes V denote either concepts ($C \subset V$) or task labels ($Y \subset V$), and the edges $E \subseteq V \times V$ represent the (plausible) existence of (bi)directed probabilistic relationships between nodes capturing the flow of information. The domain experts' knowledge comes into play when designing G. To operationalize this structure, we partition G into two directed subgraphs: the concept graph[2] $G_C := G[C]$ that contains the `C-C` relationships (e.g., correlations, exclusivities, hierarchies), and a task graph G_Y that specifies the concepts that affect specific tasks (i.e., the `C→Y` reasoning).

2.1 Concept-Concept Reasoning

Although CREAM is not restricted to categorical concepts, we assume them in this context and represent them as one-hot encoded vectors of length equal to number of categories. For instance, in the reasoning graph for the *CUB* [59] dataset, visualized in Fig. 2, the concept "Tail Color" has a cardinality of 11, and is therefore converted into a vector of 11 mutually exclusive *(mutex)* binary

[2]The graph can be disconnected, as seen by the seasonality concepts in the cFMNIST case, shown in Fig. 1.

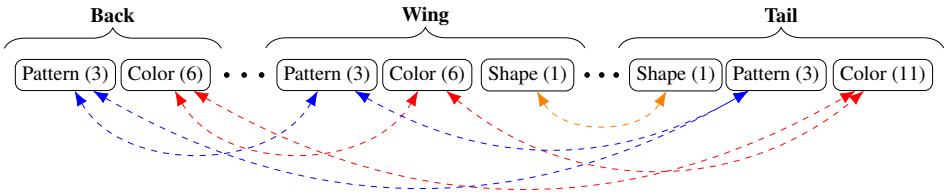


Figure 2: Partial illustration of CUB's reasoning graph is shown, where concepts are represented as nodes, with numbers in parentheses indicating the cardinality for each concept. Nodes within the same group are mutually exclusive and disconnected, while edges are one-to-one between nodes from different groups. Bidirected edges indicate statistical dependencies between concepts.

concepts. From this point onward, we assume the concept set C to be in its binarized version and refer to it as such, with a total of $K \coloneqq |C|$ concepts. To encode this graph into CREAM, we define the concept adjacency matrix, $A_C \in \{0,1\}^{K \times K}$ as follows:

$$A_C(i,j) = \begin{cases} 1 & \text{if } (c_i, c_j) \in E_{G_C}; \\ 1 & \text{if } i = j; \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where c_i and c_j denote the binarized concepts. Then, A_C is explicitly enforced in the concept-concept block to sever undesired information flows (i.e., zero entries in A_C), enabling `C-C` relationships.

Types of `C-C` Relationships. A_C allows us to capture various types of relationships among the concepts. First, it allows us to capture both the *hierarchical* and *bidirected* concept relationships that lead to asymmetric ($A_C(i,j) \neq A_C(j,i)$) and symmetric ($A_C(i,j) = A_C(j,i)$) entries respectively. For instance, while in the CelebA graph Fig. 3, all concept relationships are hierarchical (e.g., "High Cheekbones" lead to "Narrow Eyes" but not vice versa), in the graph in Fig. 2 for CUB, bidirected edges account for concepts that are not hierarchically related but correlated (such as the bird's back and wing colors).

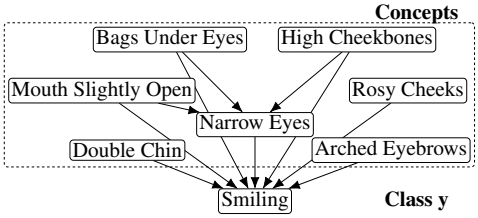


Figure 3: Reasoning graph for predicting "Smiling" (Y) in CelebA dataset [30]. The graph shows the relationships between most correlated concepts (C) within the face region that *directly* contribute to smiling.

Moreover, CREAM allows experts to specify their reasoning process at *different levels of granularity*. For instance, in CUB (Fig. 2) relationships are expressed at the concept level (i.e., "Back Color" from/to "Wing Color"). While in FashionMNIST (*FMNIST*)[60] relationships are between individual concept classes (Fig. 1) (e.g., a garment classified as "Tops" must be one of {"T-shirt", "Pullover", "Shirt"} but neither "Accessories" nor "Shoes").

2.2 Concept-Task Reasoning

Let $L \coloneqq |Y|$ be the number of possible values the target variable Y can take. As before, we assume that in case of a multiclass classification problem, the target variable has been binarized. We also assume that all `C→Y` edges are directed from concepts to task classes, i.e., that experts reason in a "downwards" fashion towards the targets. Based on these assumptions, we define the task graph G_Y as a DAG containing all task classes Y, the subset of concepts directly connected to them ($C_{direct} = \{v \in C | \exists\, y \in Y : (v,y) \in E\}$), and the corresponding directed edges that connect them (E_{G_Y}). Therefore, it is defined as the induced subgraph $G_Y = G[Y \cup C_{direct}]$. Unlike prior CBMs, our formulation does not require all concepts to have direct links to the task ($C_{direct} \subseteq C$). For example, in Fig. 1, the concept "Clothes" influences the final task prediction indirectly via its connection to "Tops", which in turn connects to the target classes "T-Shirt", "Bottoms", and "Dresses".
Remark. The concepts not directly connected to any task class contribute to predicting other concepts within G_C and enhance interpretability, enabling users to trace and verify the intermediate steps of the model's reasoning process.

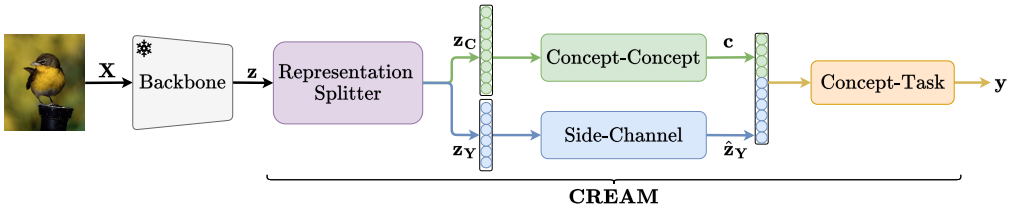


Figure 4: Sketch of CREAM, where the backbone output is split into concept ($\mathbf{z}_C$) and side-channel ($\mathbf{z}_Y$) representations for concept and task prediction, respectively. The Concept-Concept block models relationships between concepts, while the Concept-Task block uses both $\hat{\mathbf{z}}_Y$ and the concepts to predict the task label based on expert reasoning.

As before, we encode the C→Y relationships in G_Y using a task adjacency matrix[3] $A_Y \in \{0,1\}^{K \times L}$:

$$A_Y(i,j) = \begin{cases} 1 & \text{if } (c_i, y_j) \in E_{G_Y}; \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where c_i and y_j denote the i-th binarized concept and j-th one-hot encoded task class respectively. Similarly to A_C, we apply A_Y explicitly as a mask in the concept-task block, guiding the flow of information exclusively between directly connected concepts and task classes.

Supplementing C→Y reasoning. Sometimes the set of supervised concepts that make up the domain experts' C→Y reasoning might be incomplete, leading to suboptimal task performance. For example, in Fig. 1, how can we decide which class a garment classified as "Tops" belongs to? This presents a fundamental trade-off between interpretability and predictive performance. To address this limitation, we introduce an *information side-channel* [21]; a black-box representation that supplements the C→Y reasoning by capturing patterns in the input data that complement the predefined concepts to improve accuracy. However, we need to regularize the side-channel so it does not override the concept-based pathway (see Section 3.4). The idea is that, even when the side-channel contributes partially to the task prediction, the importance of concepts to the final decision should remain high and explicitly traceable via the concept-task graph.

Ease of Interventions. Given a known G, users can easily identify errors and correct them via interventions. Since the indirect concepts do not directly influence the prediction, they become irrelevant when intervening, leading to an upper limit $|C_{direct}|$ of *effective interventions*. Hence, the subset of directly related concepts C_{direct}, serves as a score-less concept selection criterion for interventions, reducing human effort and complementing existing criteria [55]. Moreover, G_C highlights relationships among concepts, such as mutually exclusive groups, facilitating grouped interventions rather than individual ones, as demonstrated in [27, 55]. Consequently, as empirically illustrated in Section 5, CREAM allows for efficient and targeted human interventions to address task prediction errors.

Concept Leakage. CBMs have shown to exhibit unexpected high task performance, due to: (i) soft concept representations encoding extra unintended information compared to their hard counterparts [21, 32, 35, 36, 38, 57], and (ii) tasks being predicted using irrelevant concepts [32, 57]. For example, in iFMNIST, a leakage-prone model might accurately distinguish among three types of apparel even when they should be indistinguishable, based solely on the binary concept "Tops". Similarly, consider cFMNIST, where the concept "Shirt" should be predicted based on just "Tops" and "Summer". CBMS, however, may instead rely on all available concepts, including irrelevant ones such as "Shoes", thus demonstrating concept leakage.

Following [36], we define concept leakage as the surplus task predictive accuracy of a CBM using the predicted concepts, compared to an optimal classifier that uses the ground-truth concepts. In this work, we investigate the extent to which imposing a specified model reasoning helps mitigating concept leakage. Our hypothesis (which we empirically validate) is that the reasoning constraints in A_C and A_Y discourage unintended information to leak from concept representations by blocking spurious or semantically invalid pathways between concepts and tasks. In doing so, the model is compelled to rely on the intended relationships, thereby reducing leakage and enhancing interpretability.

[3] Although adjacency matrices are conventionally square, we refer to this $K \times L$ binary matrix as an adjacency matrix for notational consistency.

3 Designing CREAM

3.1 Representation Splitter

CREAM (illustrated in Fig. 4) is built atop a frozen pre-trained and/or fine-tuned backbone network. Given an input image X, the backbone extracts a feature representation $\mathbf{z}$, which serves as the initial information bottleneck. This feature vector is then passed through a representation splitter, a learnable linear transformation that partitions $\mathbf{z}$ into two disjoint latent representations:

1. **Concept exogenous variables** $\mathbf{z}_C \in \mathbb{R}^{d_C K}$ that serve as *input to the concept-concept block*, which enforces the `C-C` relationships. We assume a uniform latent capacity per concept for simplicity.
2. **Side-channel information** $\mathbf{z}_Y \in \mathbb{R}^{|\mathbf{z}|-d_C K}$ captures latent patterns outside the predefined concept space. This representation is further projected via an MLP to $\hat{\mathbf{z}}_Y \in \mathbb{R}^L$ and *acts as the exogenous input to the tasks*. Thus every class has an assigned exogenous variable. Increasing the dimensionality $\hat{\mathbf{z}}_Y$ may enable greater performance, with the cost of reduced concept importance.

The dimensionalities of both $\mathbf{z}_C$ and $\mathbf{z}_Y$ (controlled by d_C), respectively, are hyperparameters that influence the model's performance.

To enforce the desired model reasoning encoded by G we draw inspiration from Structural Causal Models (SCMs) [42]. In SCMs, each observable endogenous variable x_i (in our case $X = C \cup Y$) is modeled as function of its causal parents $\text{pa}(x_i)$ (defined by `C-C` and `C→Y`) and an unobservable exogenous noise variable $\mathbf{z}_i$, through a function: $x_i = f_i(\text{pa}(x_i), \mathbf{z}_i)$. Note that we do not aim to be causally consistent (i.e., we do not impose any causal assumptions, nor explicitly define the equations f), but impose a given reasoning.

Structured Neural Networks. Structured Neural Networks (StrNNs) [10] enable us to enforce the functional (in)dependence constraints implied by G, meaning a variable x_i must not be influenced by the exogenous noise of a x_j that is not its parent ($\frac{\partial x_i}{\partial \mathbf{z}_j} = 0, \{\forall j \mid x_j \notin pa(x_i)\}$) . Given the number of hidden layers d, layer widths $(h_1, h_2, \ldots, h_d)$, and an adjacency matrix $\mathbf{A} \in \{0,1\}^{p \times q}$, StrNN constructs series of binary masks: $M_1 \in \{0,1\}^{h_1 \times p}, M_2 \in \{0,1\}^{h_2 \times h_1}, \ldots, M_d \in \{0,1\}^{q \times h_d}$ using binary matrix factorization. These masks zero out non-permitted connections, ensuring the desired independencies are effectively encoded while preserving maximal expressivity within those constraints. A detailed explanation of StrNNs can be found in App. A.

3.2 Concept-Concept Block

In this block we enforce the `C-C` relationships encoded in A_C. Usually exogenous noise variables are one dimensional per endogenous variable. However, to improve predictive performance, without loss of generality, we assume each concept is associated with a d_C-dimensional exogenous embedding, yielding an input representation $\mathbf{z}_C \in \mathbb{R}^{d_C K}$. To enforce the concept graph G_C, we generate binary masks using StrNNs: $M_C := A_C^T \otimes \mathbb{1}_{1 \times d_C}$, where $\otimes$ denotes the Kronecker product and $\mathbb{1}_{1 \times d_C}$ is a row vector of ones. This ensures that each concept receives input only from the appropriate parents' exogenous vectors. The resulting concept-concept block $g : \mathbb{R}^{d_C K} \to \mathbb{R}^K$ receives as input $\mathbf{z}_C$ and relies on the concept mask M_C to compute the *concept logits*, $\hat{l}_C \in \mathbb{R}^K$ as :

$$\hat{l}_{C_i} = g(\mathbf{z}_{C_i}, \text{pa}(\mathbf{z}_C)), \quad \text{where pa}(C_i) = \{v \in V | A[v, C_i] = 1\}. \tag{3}$$

This formulation follows the compacted SCM principle from [25], in which each concept depends only on its parents' and its corresponding exogenous variables. In contrast, vanilla CBM lacks this structure and connects all $\mathbf{z}_C$ to every concept allowing for entangled and unwanted reasoning paths.

Handling mutually exclusive concepts. Some concepts in G_C may be mutually exclusive (mutex), e.g., in Fig. 1 the concepts "Clothes" and "Goods". These typically emerge from categorical variables encoded via one-hot representations or are dictated by domain knowledge. For such concepts, a softmax activation is applied over the logits, across the group. For the rest non-exclusive soft concepts, we apply a sigmoid activation over their logits independently: $\hat{C} := \sigma(\hat{l}_C)$, allowing multiple concepts to be active simultaneously.

3.3 Concept-Task Classifier

The final stage of CREAM maps concept predictions to task logits while incorporating side-channel information in a controlled manner. Similar to the concept-concept block, we use StrNN to

enforce C→Y relationships expressed by A_Y. To incorporate the side-channel representation $\mathbf{z}_Y$, we parameterize the concept-task StrNN using the binary mask $M_Y := [A_Y^T; I_L]$, where I_L denotes the identity matrix of size L that connects each element in the side-channel representation $\mathbf{z}_Y$ with just one of the tasks classes. This ensures each class is dependent only on its parent concepts (Eq. 3) and, class-specific latent feature from the side-channel. Formally, the task prediction for class j is computed as:

$$\hat{y}_j = f([\hat{c}_{Pa_j}, \mathbf{z}_{Y_j}]) \quad \text{where } f : \mathbb{R}^{K+L} \to \mathbb{R}^L \tag{4}$$

where f is usually a single layer classifier for interpretability between concepts and tasks.

3.4 Training

To train CREAM, we adopt the joint bottleneck training scheme [27], which optimizes both the task loss ($\mathcal{L}_Y$) and the concept loss ($\mathcal{L}_C$) simultaneously, through linear scalarization objective. The optimization objective is to minimize the weighted sum of these losses ($\mathcal{L}$), for the observed training samples $\{(x^{(n)}, y^{(n)}, c^{(n)})\}_{n=1}^N$, where concepts $c \in \mathbb{R}^K$ and target class $y \in \{0, 1\}^L$:

$$\mathcal{L} = \sum_n \mathcal{L}_Y(\hat{y}^{(n)}; y^{(n)}) + \lambda \sum_n \sum_k \mathcal{L}_{C_k}(\hat{c}^{(n)}; c^{(n)}) \tag{5}$$

where $\lambda > 0$ is the weight of concept performance compared to task performance.

Regularization of side-channel. While adding a black-box side channel to a CBM model can boost task performance, it harms interpretability, since the task predictions can be based on non-interpretable predictors. To encourage the model to prioritize the concept-based reasoning we apply a dropout based regularization strategy [24]. During training, the entire side-channel is dropped out with probability p, forcing the model to rely only on the concepts for predictions. Thus the model learns to use concepts as the primary reasoning pathway and to fall back on the side-channel only when necessary. At inference time, this setup allows a user to optionally disable the side-channel, yielding a purely interpretable prediction path grounded in the concepts.

4 Related Work

Connection to Neurosymbolic approaches. CBMs are complementary to Neurosymbolic (NeSy) approaches [3, 7, 34, 36]; the former require supervision solely on the concepts while the latter require knowledge in the form of logical programs. In our case, we require knowledge about *(directed) statistical (in)dependencies* between interpretable variables, to constrain the relationships between them. In App. H we provide a logic-based viewpoint to CREAM. For instance, a mutex constraint is described as: Clothes $\sqcap$ Goods $\sqsubseteq \bot$, and concepts are calculated by: Tops $\leftarrow \mathbf{z}_{Clothes} \sqcap \mathbf{z}_{Tops}$.

Concept and Task Relationships. Standard CBMs assume that concepts are *independent* and that all of them directly contribute to the task, thus forming a bipartite graph. To address this, several works have incorporated concept interdependencies. Relational CBMs[5] use graph-structured data and message-passing algorithms to propagate relational dependencies, while Stochastic CBMs (SCBMs)[58] model concepts using a learnable covariance matrix. Similarly, Autoregressive CBMs (ACBMs) [21] introduce an autoregressive structure to learn sequential dependencies between concepts. These methods do not explicitly model expert-desired C-C and C→Y relationships. Concurrent work also contains Causal CGMs [15] and C^{2}BMs [14]. The core differences lie in modeling G. The former learns relationships between endogenous variables and their copies and does not assume an acyclic C→Y reasoning. While the latter models each connection in G with a distinct MLP and assumes linear relationships. Neither account for importance of the endogenous variables. Lastly, none of the aforementioned works explicitly handle mutually exclusive concepts.

Concept Incompleteness. CBMs rely on predefined concept sets, which can lead to degraded accuracy when the concept set is incomplete (i.e. not sufficient statistic for predicting the target) [32, 63, 67]. Such cases can arise when: (i) annotation of concepts is expensive, (ii) the experts require small partial explanations, or (iii) not enough domain knowledge is available. In order to tackle concept incompleteness, CBMs have been extended to incorporate side-channels. These hybrid CBMs have a lower upper bound of the generalization error than the standard model [22], they can be trained in a two-stage scheme [21], capture other unsupervised concepts [51], or residuals [64, 65]. Our work proposes to regularize the side-channel to give precedence to concept importance.

Concept Leakage. Furthermore, CBMs suffer from concept leakage, a phenomenon tied to reasoning shortcuts [7, 8, 19, 37], where extra unintended information is encoded into concepts [32, 33, 36, 38, 44], leading to high accuracy even with irrelevant concepts and thus unreliable reasoning. Existing methods to mitigate leakage include using binary concept representations [21, 31, 57, 58], training a CBM model independently [38], using orthogonality losses [54] or disentanglement techniques [35, 56]. Meanwhile, the reasoning structure used in CREAM mitigates concept leakage without the need for hard concepts, or introducing extra terms in the loss function.

5 Experiments

We evaluate CREAM across multiple datasets, demonstrating its ability to balance performance and interpretability. Our experiments assess: (i) concept and task accuracies, (ii) the importance of concept-channel in task predictions, (iii) intervenability, and (iv) mitigation of concept leakage.

5.1 Setup

Datasets. We conduct experiments on three real-world image datasets to evaluate the effectiveness of CREAM. FashionMNIST *(FMNIST)* [60] consists of grayscale images of ten different clothing categories. We define two variations of this dataset based on the level of concept supervision: incomplete FMNIST *(iFMNIST)* follows the hierarchical clothing categories defined in [52] with $K = 8$, while complete FMNIST *(cFMNIST)* extends the concept set to $K = 11$ by incorporating three additional seasonal attributes, as illustrated in Fig. 1. We also evaluate on the Caltech-UCSD Birds-200-2011 *(CUB)* dataset [59], which contains photographs of 200 bird species. Following the preprocessing of [27], we use 112 expert-defined concepts that describe attributes such as tail color and wing pattern. Additionally, we test our method on CelebA [30], a large-scale dataset of celebrity images. The task here is to predict whether a person is smiling while using a subset of seven facial attributes as concepts (Fig. 3). Further details on these datasets, can be found in App. C.

Model Baselines. We compare CREAM with state-of-the-art models, evaluating its performance against both black-box models and concept-based models with concept dependencies. As a baseline, we train a black-box model to establish a reference for task performance. Among concept-based architectures, we include the autoregressive CBM *(ACBM)* [21], where each concept is predicted sequentially as a function of the input x and previously predicted concepts, i.e., $c_k = f(x, c_{1:k-1})$.[4] Additionally, we compare against Stochastic CBM *(SCBM)* [58], which models concept dependencies using a multivariate normal distribution. Both ACBM and SCBM operate with hard concept representations. Finally, we include a vanilla CBM with soft concepts.

Implementation Details. All models are initialized from a shared backbone network, which is fine-tuned for a few epochs on the respective datasets and then frozen to ensure consistent feature extraction. For each dataset, we perform hyperparameter tuning; detailed hyperparameter configurations are provided in App. D.[5] The construction of masking pathways in StrNNs follows the algorithm found in Zuko [46]. For baseline models such as SCBM and ACBM, we adopt the experimental settings from [58]. Notably, since the Amortized SCBM consistently outperforms the Global SCBM in their reported benchmarks, we include only the amortized variant in our evaluations.

5.2 Key Findings

Bridging the gap between interpretability and performance. Our results in Table 1 demonstrate that CREAM achieves task performance on par with black-box classifiers even in incomplete concepts settings while also achieving competitive concept accuracy relative to concept-based baselines. Notably, for CelebA, CREAM outperforms black-box models in task accuracy, suggesting that structured domain knowledge, can provide inductive biases that enhance generalization. Crucially, we find that the two main components of CREAM offer orthogonal and complementary benefits. Structured reasoning promotes interpretability and robustness to spurious correlations, while the side-channel addresses the limitations of incomplete or noisy concepts. App. E.1 further shows that even without the side-channel, CREAM maintains strong task performance across datasets, underscoring the strength of structure-aware reasoning alone. One notable deviation is CUB, where CREAM exhibits a drop in concept accuracy. This highlights an important insight: while structure can enhance interpretability, it is crucial that the encoded knowledge reflects the true dependencies,

[4] We use the implementation available in SCBM's repository.

[5] We will release our code publicly upon acceptance to ensure full reproducibility.

or it may act as a constraint rather than a guide. Nevertheless, CREAM's task performance remains competitive, demonstrating the side-channel's ability to supplement reasoning, enabling the model to recover performance when the concept pathway is insufficient. In App. F.2, we provide an ablation study exploring the effect of d_C on concept and task accuracy. We explore hard concepts in App. G.

Table 1: Task and concept accuracy(%). Reported values represent the mean and standard deviation. CREAM consistently achieves a better balance between performance and interpretability. The best-performing method is highlighted in **bold** and the second-best is underlined.

Model	iFMNIST		cFMNIST		CUB		CelebA	
	Test ACC_Y	**Test** ACC_C	**Test** ACC_Y	**Test** ACC_C	**Test** ACC_Y	**Test** ACC_C	**Test** ACC_Y	**Test** ACC_C
Black-box	92.70 ± 0.07	-	92.70 ± 0.07	-	74.85 ± 0.12	-	78.09 ± 0.45	-
CBM	91.14 ± 0.17	96.80 ± 0.45	91.91 ± 0.11	97.33 ± 0.05	62.03 ± 0.51	73.83 ± 0.17	78.26 ± 1.35	80.81 ± 0.49
ACBM	53.76 ± 5.21	$\mathbf{99.16} \pm 0.02$	$\mathbf{92.87} \pm 0.26$	$\mathbf{98.33} \pm 0.04$	68.60 ± 0.24	$\mathbf{95.21} \pm 0.06$	74.45 ± 1.52	$\mathbf{82.65} \pm 0.69$
SCBM	58.23 ± 1.16	98.86 ± 0.02	90.93 ± 0.21	97.55 ± 0.08	70.52 ± 0.37	90.29 ± 0.01	76.30 ± 0.55	80.68 ± 0.19
CREAM	$\mathbf{92.43} \pm 0.23$	99.07 ± 0.03	92.38 ± 0.16	98.08 ± 0.06	$\mathbf{72.93} \pm 0.15$	86.85 ± 0.04	$\mathbf{79.99} \pm 3.14$	80.32 ± 0.47

Interpretability in the presence of a side-channel. Given that our model incorporates a black-box side-channel, it is crucial to verify that predictions primarily rely on the concepts rather than the side-channel. To quantify this, we introduce a new metric called Concept Channel Importance (CCI). CCI is adapted from SAGE values [12], which measure *global feature importance* [39] and conditional mutual information when used with an optimal model. CCI is defined as the normalized importance of the concept channel relative to total predictive capacity: $CCI = \frac{\phi_c}{\phi_c + \phi_y}$, where ϕ_c and ϕ_y denotes the SAGE value of the whole concept and side channels respectively. Values closer to 1 indicate that the model relies more on the concepts, and thus a more interpretable model. An analysis of CCI and permutation feature importance [9, 17] for CREAM, can be found in App. B. We show that $CCI > 0.5$ for our desiderata to hold. Fig. 5 illustrates the impact of the dropout rate p on CCI. As expected, increasing the dropout rate regularizes more the side-channel, ensuring the model prioritizes concept-based reasoning. Also, the need for side-channel regularization decreases when using complete concept sets. This underscores the necessity of dropout to counteract the model's tendency to overly rely on the side-channel. These findings lead to a key conclusion: *dropout rate controls interpretability*. Additional results on the effect of dropout on model performance are provided in App. F.1. We observe increasing p mildly decreases the task performance in complete cases. Surprisingly, we find that even with an extreme dropout rate of $p \approx 1$, CREAM maintains black-box level accuracy in incomplete datasets.

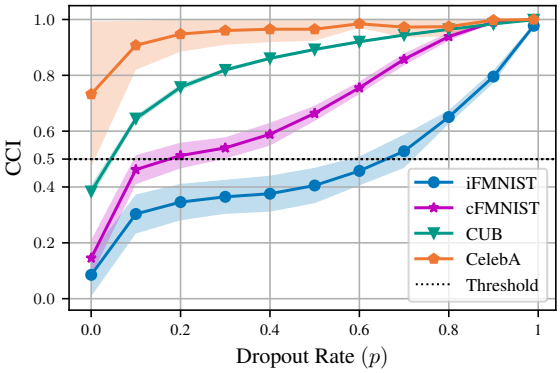


Figure 5: Impact of dropout rates on channel importance. Models with complete concept sets require less side-channel regularization.

Intervenability. Additionally, we assess task accuracy after concept interventions, measuring how effectively the model adapts when users manually correct its concept predictions. For models with hard concepts, interventions are straightforward: the true concept values are directly inserted. However, for soft concept models, we set the concept activations to the 5th and 95th percentiles as proposed in [27]. Concept selection for intervention follows a random policy [55], with a key distinction in CREAM, where we prioritize interventions on direct parents of the target classes. Fig. 6 illustrates task accuracy across models following interventions. In iFMNIST, all models but CBM improve with increasing number of interventions. CBM drops down to the $C_{true} \rightarrow Y$ baseline's performance, a phenomenon that appears in leaky models [38]. Conversely, even after intervening on all available concepts, no model achieves 100% accuracy, highlighting the inherent limitation of the incomplete concept set. Similarly, in CUB, all models improve with more interventions, except soft CBM, whose performance degrades when excessive interventions are applied, mirroring observations from Figure 4 in [27]. We also notice SCBM drops performance for a low number of interventions. Lastly in both cFMNIST and CelebA, CREAM reaches the accuracy of the baseline after full interventions, with CUB being an exception; we assume that performance gap is due to the side-channel. Note that both SCBM and ACBM, propagate interventions effects to the non-intervened

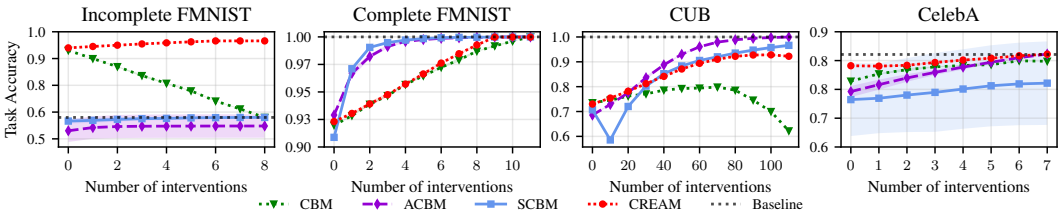


Figure 6: Impact of individual interventions on task accuracy. The baseline model is $C_{true} \rightarrow Y$. CREAM's accuracy improves with increasing number of interventions up to the number of C_{direct}, even in iFMNIST. The number of C_{direct} for each dataset are, from left to right, 6, 9, 112 and 7.

Table 2: CREAM's performance on iFMNIST without the side-channel. Leakage is completely avoided when using the `C→Y` relationships, while `C-C` relationships help mitigate it.

Activation	**Model**	**Test** ACC_Y	**Test** ACC_C	**Leakage** Λ
Softmax	No side-channel	57.35 ± 0.23	99.07 ± 0.01	No Leakage
	Only `C-C`	67.81 ± 7.84	98.51 ± 1.09	7.81 ± 7.84
	Only `C→Y`	57.35 ± 0.37	99.04 ± 0.03	No Leakage
Sigmoid	No side-channel	57.39 ± 0.58	99.04 ± 0.02	No Leakage
	Only `C-C`	90.27 ± 4.21	96.37 ± 0.83	30.37 ± 0.83
	Only `C→Y`	57.32 ± 0.32	99.07 ± 0.01	No Leakage

concepts, which leads to steeper intervention curves. However, CREAM has an upper-bound on the number of interventions, as seen in iFMNIST (6 interventions) and cFMNIST (9 interventions). In App. E.2, we study the efficiency of group interventions on mutex concepts.

Concept Leakage. We empirically show that imposing structured reasoning mitigates concept leakage. Drawing inspiration from [36], we define concept leakage as the difference in task accuracy between a classifier f that uses the predicted concept values, and the Bayes optimal predictor using the true concept values: $\Lambda = \max(ACC_f - ACC_{Optimal}, 0)$. In other words, the model's accuracy should not exceed the accuracy of the $C_{true} \rightarrow Y$ model when relying solely on the concepts. Note that we modify the definition in [36] to accuracy instead of classification loss. We focus our analysis on iFMNIST for two reasons: (1) the baseline $C_{true} \rightarrow Y$ model is outperformed only in iFMNIST, and (2) models with soft concepts are expected to exhibit significant leakage in concept incomplete datasets [32]. For instance, soft CBM is leaking in iFMNIST: as shown in Table 1, its accuracy exceeds that of the $C_{true} \rightarrow Y$ model (60%) reported in Table 5. To isolate which mechanisms mitigate leakage in CREAM, we first remove the side-channel and then: (i) remove the `C-C` reasoning (ii) remove the `C→Y` reasoning, and (iii) stop treating concepts as mutually exclusive by using a sigmoid activation. According to Table 2, CREAM does not suffer from concept leakage, despite being a soft model. Specifically, the `C→Y` reasoning seems to completely avoid it, while the `C-C` relationships help mitigate the grade of leakage, sometimes even completely avoiding it. In App. E.3 we investigate the impact of the each block on the correlations between exogenous variables. We observe that A_C leads to correlation matrices mirroring the imposed relationships.

6 Conclusion and Future work

In this work, we have introduced CREAM, a novel class of CBMs that incorporates expert reasoning and a regularized black-box side-channel, in a sensible package. CREAM achieves performance equivalent to black-boxes, bridges the interpretability-performance gap while primarily basing its predictions on concepts, and eases the intervention procedure. We also empirically showed that the provided reasoning mitigates concept leakage, thus making CREAM the only CBM framework that, to the best of our knowledge, uses soft concepts and is leakage-free.

Future Work and Limitations. CREAM effectively incorporates domain knowledge, though this approach may face challenges in scenarios with limited prior information. Future work could explore structure-learning techniques, such as causal discovery [66] to automate dependency inference. Although, flexibility can be compromised by misaligned reasoning, this also presents an opportunity to identify mismatches between data and the expert knowledge. Additionally, implementing adaptive dropout strategies, where the side-channel is dynamically leveraged based on concept prediction uncertainty, could improve both robustness and interpretability. The side-channel can also be used to discover new concepts [51] that are not present in the concept bottleneck.

7 Acknowledgments

We would like to thank Pablo M. Olmos and every member of the Probabilistic Machine Learning lab for their invaluable feedback, as well as the anonymous reviewers whose feedback helped us improve this manuscript. This work was funded in part by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – GRK 2853/1 "Neuroexplicit Models of Language, Vision, and Action" - project number 471607914 and by the European Union (ERC-2021-STG, SAML, 101040177). However, the views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible.

References

[1] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. *Advances in neural information processing systems*, 2018.

[2] Franz Baader. *The description logic handbook: Theory, implementation and applications*. Cambridge university press, 2003.

[3] Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks. *Artificial Intelligence*, 2022.

[4] Pietro Barbiero, Gabriele Ciravegna, Francesco Giannini, Mateo Espinosa Zarlenga, Lucie Charlotte Magister, Alberto Tonda, Pietro Lió, Frederic Precioso, Mateja Jamnik, and Giuseppe Marra. Interpretable neural-symbolic concept reasoning. In *International Conference on Machine Learning*. PMLR, 2023.

[5] Pietro Barbiero, Francesco Giannini, Gabriele Ciravegna, Michelangelo Diligenti, and Giuseppe Marra. Relational concept bottleneck models. In *Advances in Neural Information Processing Systems*, 2024.

[6] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *ArXiv*, 2013.

[7] Samuele Bortolotti, Emanuele Marconato, Tommaso Carraro, Paolo Morettin, Emile van Krieken, Antonio Vergari, Stefano Teso, Andrea Passerini, et al. A neuro-symbolic benchmark suite for concept quality and reasoning shortcuts. *Advances in neural information processing systems*, 2024.

[8] Samuele Bortolotti, Emanuele Marconato, Paolo Morettin, Andrea Passerini, and Stefano Teso. Shortcuts and identifiability in concept-based models from a neuro-symbolic lens. *ArXiv*, 2025.

[9] Leo Breiman. Random forests. *Machine learning*, 2001.

[10] Asic Chen, Ruian Ian Shi, Xiang Gao, Ricardo Baptista, and Rahul G Krishnan. Structured neural networks for density estimation and causal inference. *Advances in Neural Information Processing Systems*, 2024.

[11] Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2020.

[12] Ian Covert, Scott M Lundberg, and Su-In Lee. Understanding global feature contributions with additive importance measures. *Advances in Neural Information Processing Systems*, 2020.

[13] Roxana Daneshjou, Mert Yuksekgonul, Zhuo Ran Cai, Roberto Novoa, and James Y Zou. Skincon: A skin disease dataset densely annotated by domain experts for fine-grained debugging and analysis. In *Advances in Neural Information Processing Systems*, 2022.

[14] Giovanni De Felice, Arianna Casanova Flores, Francesco De Santis, Silvia Santini, Johannes Schneider, Pietro Barbiero, and Alberto Termine. Causally reliable concept bottleneck models. *ArXiv*, 2025.

[15] Gabriele Dominici, Pietro Barbiero, Mateo Espinosa Zarlenga, Alberto Termine, Martin Gjoreski, Giuseppe Marra, and Marc Langheinrich. Causal concept graph models: Beyond causal opacity in deep learning. In *International Conference on Learning Representations*, 2025.

[16] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *ArXiv*, 2017.

[17] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *Journal of Machine Learning Research*, 2019.

[18] Florent Forest, Katharina Rombach, and Olga Fink. Interpretable prognostics with concept bottleneck models. *ArXiv*, 2024.

[19] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2020.

[20] Andreas Grivas, Antonio Vergari, and Adam Lopez. Taming the sigmoid bottleneck: Provably argmaxable sparse multi-label classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

[21] Marton Havasi, Sonali Parbhoo, and Finale Doshi-Velez. Addressing leakage in concept bottleneck models. *Advances in Neural Information Processing Systems*, 2022.

[22] Naoki Hayashi and Yoshihide Sawada. Upper bound of bayesian generalization error in partial concept bottleneck model (cbm): Partial cbm outperforms naive cbm. *ArXiv*, 2024.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[24] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*. Springer, 2016.

[25] Adrián Javaloy, Pablo Sánchez-Martín, and Isabel Valera. Causal normalizing flows: from theory to practice. *Advances in Neural Information Processing Systems*, 2024.

[26] Diederik P Kingma. Adam: A method for stochastic optimization. *ArXiv*, 2014.

[27] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*. PMLR, 2020.

[28] Himabindu Lakkaraju and Osbert Bastani. " how do i fool you?" manipulating user trust via misleading black box explanations. In *ACM Conference on AI, Ethics, and Society*, 2020.

[29] Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 2018.

[30] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision*, 2015.

[31] Joshua Lockhart, Nicolas Marchesotti, Daniele Magazzeni, and Manuela Veloso. Towards learning to explain with concept bottleneck models: mitigating information leakage. *ArXiv*, 2022.

[32] Anita Mahinpei, Justin Clark, Isaac Lage, Finale Doshi-Velez, and Weiwei Pan. Promises and pitfalls of black-box concept learning models. *ArXiv*, 2021.

[33] Mikael Makonnen, Moritz Vandenhirtz, Sonia Laguna, and Julia E Vogt. Measuring leakage in concept-based methods: An information theoretic approach. In *ICLR 2025 Workshop: XAI4Science: From Understanding Model Behavior to Discovering New Scientific Knowledge*, 2025.

[34] Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming. *Advances in neural information processing systems*, 2018.

[35] Emanuele Marconato, Andrea Passerini, and Stefano Teso. Glancenets: Interpretable, leak-proof concept-based models. *Advances in Neural Information Processing Systems*, 2022.

[36] Emanuele Marconato, Andrea Passerini, and Stefano Teso. Interpretability is in the mind of the beholder: A causal framework for human-interpretable representation learning. *Entropy*, 2023.

[37] Emanuele Marconato, Stefano Teso, Antonio Vergari, and Andrea Passerini. Not all neuro-symbolic concepts are created equal: Analysis and mitigation of reasoning shortcuts. *Advances in Neural Information Processing Systems*, 2023.

[38] Andrei Margeloiu, Matthew Ashman, Umang Bhatt, Yanzhi Chen, Mateja Jamnik, and Adrian Weller. Do concept bottleneck models learn as intended? *ArXiv*, 2021.

[39] Christoph Molnar. *Interpretable Machine Learning*. 3 edition, 2025. ISBN 978-3-911578-03-5. URL `https://christophm.github.io/interpretable-ml-book`.

[40] Tuomas Oikarinen, Subhro Das, Lam M. Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. In *International Conference on Learning Representations*, 2023.

[41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 2019.

[42] J. Pearl. *Causality*. Causality: Models, Reasoning, and Inference. Cambridge University Press, 2009. ISBN 9780521895606. URL `https://books.google.de/books?id=f4nuexsNVZIC`.

[43] Eleonora Poeta, Gabriele Ciravegna, Eliana Pastor, Tania Cerquitelli, and Elena Baralis. Concept-based explainable artificial intelligence: A survey. *ArXiv*, 2023.

[44] Angelos Ragkousis and Sonali Parbhoo. Tree-based leakage inspection and control in concept bottleneck models. *ArXiv*, 2024.

[45] Goutham Rajendran, Simon Buchholz, Bryon Aragam, Bernhard Schölkopf, and Pradeep Ravikumar. From causal to concept-based representation learning. *Advances in Neural Information Processing Systems*, 37:101250–101296, 2024.

[46] François Rozet et al. Zuko: Normalizing flows in pytorch, 2022. URL `https://pypi.org/project/zuko`.

[47] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 2019.

[48] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistic Surveys*, 2022.

[49] Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. *Explainable AI: interpreting, explaining and visualizing deep learning*. Springer Nature, 2019.

[50] Pablo Sanchez-Martin, Kinaan Aamir Khan, and Isabel Valera. Improving the interpretability of gnn predictions through conformal-based graph sparsification. *ArXiv*, 2024.

[51] Yoshihide Sawada and Keigo Nakamura. Concept bottleneck model with additional unsupervised concepts. *IEEE Access*, 2022.

[52] Yian Seo and Kyung-shik Shin. Hierarchical convolutional neural networks for fashion image classification. *Expert systems with applications*, 2019.

[53] Lloyd S Shapley et al. A value for n-person games. 1953.

[54] Ivaxi Sheth and Samira Ebrahimi Kahou. Auxiliary losses for learning generalizable concept-based models. *Advances in Neural Information Processing Systems*, 2023.

[55] Sungbin Shin, Yohan Jo, Sungsoo Ahn, and Namhoon Lee. A closer look at the intervention procedure of concept bottleneck models. In *International Conference on Machine Learning*. PMLR, 2023.

[56] Sanchit Sinha, Guangzhi Xiong, and Aidong Zhang. Colidr: Concept learning using aggregated disentangled representations. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024.

[57] Ao Sun, Yuanyuan Yuan, Pingchuan Ma, and Shuai Wang. Eliminating information leakage in hard concept bottleneck models with supervised, hierarchical concept learning. *ArXiv*, 2024.

[58] Moritz Vandenhirtz, Sonia Laguna, Ričards Marcinkevičs, and Julia Vogt. Stochastic concept bottleneck models. *Advances in Neural Information Processing Systems*, 2024.

[59] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[60] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[61] Kaiyu Yang, Jacqueline H Yau, Li Fei-Fei, Jia Deng, and Olga Russakovsky. A study of face obfuscation in imagenet. In *International Conference on Machine Learning*. PMLR, 2022.

[62] Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In *Conference on Computer Vision and Pattern Recognition*, 2023.

[63] Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in neural information processing systems*, 2020.

[64] Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. In *International Conference on Learning Representations*, 2023.

[65] Renos Zabounidis, Ini Oguntola, Konghao Zhao, Joseph Campbell, Simon Stepputtis, and Katia Sycara. Benchmarking and enhancing disentanglement in concept-residual models. *ArXiv*, 2023.

[66] Alessio Zanga, Elif Ozkirimli, and Fabio Stella. A survey on causal discovery: Theory and practice. *International Journal of Approximate Reasoning*, 2022.

[67] Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Frederic Precioso, Stefano Melacci, Adrian Weller, Pietro Lio, et al. Concept embedding models. In *Advances in Neural Information Processing Systems*, 2022.

A Structured Neural Networks for Concept-Concept and Concept-Task Relationships

Structured Neural Networks (StrNN) [10] enforce functional independence between inputs and outputs using masking pathways that preserve the structural constraints dictated by an adjacency matrix. Given a function $f : \mathbb{R}^m \to \mathbb{R}^n$, where $z \in \mathbb{R}^m$ is the input and $\hat{z} \in \mathbb{R}^n$ is the predicted output, StrNN ensures that dependencies between inputs and outputs adhere to a given adjacency matrix $A \in \{0,1\}^{m \times n}$. This is enforced through the condition:

$$A_{ij} = 0 \implies \frac{\partial \hat{z}_j}{\partial z_i} = 0. \tag{6}$$

This means that if $A_{ij} = 0$, the output $\hat{z}_j$ remains independent of input z_i, maintaining the prescribed reasoning structure of the G.

Masks in Structured Neural Networks For both concept and task prediction networks, layer-wise masking pathways are applied to enforce structured reasoning $A^{m \times n}$. Given a network with d hidden layers, each with widths $h_1, h_2, \ldots, h_d$, we define binary masks:

$$M_1 \in \{0,1\}^{h_1 \times m}, \quad M_2 \in \{0,1\}^{h_2 \times h_1}, \quad \ldots, \quad M_d \in \{0,1\}^{n \times h_d}, \tag{7}$$

such that:

$$M' = M_d \cdot \ldots \cdot M_2 \cdot M_1 \sim M, \tag{8}$$

where $M' \in \{0,1\}^{n \times m}$ maintains the same sparsity pattern as A^T, ensuring structured dependencies are preserved. The structured neural network function StrNN_M is defined as:

$$\hat{z} = \text{StrNN}_M(z) = f_{d+1}\left(\ldots f_1\left(\left(W_C \odot M_1\right) z + b_C\right)\right), \tag{9}$$

where each layer transformation follows:

$$f_i(z) = a\left(\left(W_{C(i)} \odot M_i\right) z + b_i\right), \quad \forall i \in \{1, \ldots, d\}, \tag{10}$$

where: $W_{C(i)} \in \mathbb{R}^{h_i \times h_{i-1}}$ is the learnable weight matrix at layer i, $M_i \in \{0,1\}^{h_i \times h_{i-1}}$ is the binary mask ensuring structured dependencies, $b_i \in \mathbb{R}^{h_i}$ is the bias term and $a(\cdot)$ is the activation function.

For the concept-concept learning block StrNN_{M_C}:

$$m = d_C K, \quad n = K, \quad h_{i \geq 1} = d_C K. \tag{11}$$

For the classifier StrNN_{M_Y}:

$$m = K + L, \quad n = L, \quad h_{i \geq 1} = K + L. \tag{12}$$

The depth d is treated as a hyperparameter. By enforcing structured dependencies, StrNN ensures that both concept and task predictions follow expert-defined reasoning pathways, enhancing interpretability without sacrificing predictive performance.

A.1 Concept-Concept Masking

The adjacency matrix for concept relationships is given by $A_C \in \{0,1\}^{K \times K}$, as defined in Section 2.1. The input to the Concept-Concept block is obtained from the representation splitter and is denoted as:

$$\mathbf{z}_C = (z_1, z_2, \ldots, z_{d_C K}) \in \mathbb{R}^{d_C K}. \tag{13}$$

Note that each dimension in $\mathbf{z}_C$ is a result of an expansion; each exogenous variable was duplicated from dimensionality of 1 to dimensionality of d_C. This operation is represented by the Kronecker product ($\otimes$). These extra dimensions must still follow the independencies of the original variable. Thus, we construct the concept mask $M_C \in \{0,1\}^{K \times d_C K}$ by:

$$M_C = A_C^T \otimes \mathbb{1}_{1 \times d_C}, \tag{14}$$

where $\mathbb{1}_{1 \times d_C}$ is a row vector of ones that replicates A_C^T column-wise, ensuring structured reasoning of concept-concept relationships across all feature dimensions.

A.2 Concept-Task Masking

For the concept-task classifier (Section 3.3), the input combines both concept predictions and the side-channel information. Thus, its input is the concatenated $\hat{C}$ and $\hat{\mathbf{z}}_Y$, and thus it is of size $\mathbb{R}^{K+L}$. The C→Y relationships are described in the adjacency matrix $A_Y \in \{0,1\}^{K \times L}$. Meanwhile, since we assign one dimension of side-channel variables ($\hat{\mathbf{z}}_Y$) to each task, we will need to expand the adjacency matrix used in StrNN, using an identity matrix I_L. From this, we can define the concept-task mask as:

$$M_Y = \left[A_Y^T; I_L\right]. \tag{15}$$

This ensures that each class prediction depends only on the relevant parent concepts and its assigned side-channel node.

B Feature Importance Metrics

Given that CREAM integrates a side-channel that can contribute to task predictions, it is crucial to assess the relative importance of the concept set C compared to it. To quantify this, we employ two model-agnostic metrics: Concept Channel Importance (CCI) and Permutation Feature Importance (PFI). These two metrics collectively provide an assessment of whether CREAM effectively balances interpretability and performance by ensuring that predictions remain grounded in human-understandable concepts rather than being dominated by the side-channel. In Table 3, we report the values of these importance metrics, for the models used in Section 5.

B.1 Concept Channel Importance

Concept Channel Importance (CCI) is based on Shapley Additive Global Explanations (SAGE) [12], which provide model-agnostic feature importance scores. Specifically, the SAGE value $\phi_i(v_f)$ of feature i, represents the Shapley values [53] for the cooperative game $v_f(S)$. The cooperative game v_f represents the expectation of the per-instance reduction in risk when using a subset of features $S \subseteq D$:

$$v_f(S) = \mathbb{E}[\mathcal{L}(f_\varnothing(X_\varnothing), Y)] - \mathbb{E}[\mathcal{L}(f_S(X_S), Y)]$$

where $f_\varnothing(x_\varnothing)$ is the model prediction without using any features, (i.e., the mean model prediction $\mathbb{E}[f(X)]$), and $f_S(X_S)$ the prediction using only the subset of features S of all features D ($S \subseteq D$). Given $v_f(S)$, SAGE is then calculated by:

$$\phi_i(v_f) = \frac{1}{|D|} \sum_{S \subseteq D \setminus \{i\}} \binom{|D|-1}{|S|}^{-1} \Big(v_f(S \cup \{i\}) - v_f(S)\Big).$$

SAGE values provide global interpretability [39] instead of explanations for individual predictions. The features that the model deems most useful will have positive SAGE values, non informative features have values close to zero, and harmful for the prediction features have negative values. Lastly, SAGE values can also measure a weighted average of conditional mutual information when they are used with an optimal model trained with the cross entropy or MSE loss. Specifically, the SAGE value of a feature i used in optimal model f^*, is equal to:

$$\phi_i(v_{f^*}) = \frac{1}{|D|} \sum_{S \subseteq D \setminus \{i\}} \binom{|D|-1}{|S|}^{-1} I(Y; X_i | X_S).$$

We also briefly mention some of the properties that SAGE satisfies:

- **Efficiency:** SAGE values sum up to the total predictive power of all of the features (SAGE value using all the features D):. $\sum_{i=1}^{K} \phi_i(v_f) = v_f(D)$.
- **Dummy:** If a feature makes zero contribution, i.e., if it is an uninformative feature, then $\phi_i(v_f) = 0$.

Interestingly, SAGE values can be generalized to group of features. Since our objective is to measure the overall contribution of the concept channel, we treat all of the concepts as one coalition. The SAGE value of a coalition represents how much this group of features improves the model's predictive ability.

Table 3: Interpretability metrics: CREAM variants (S-CREAM for soft concepts, H-CREAM for hard concepts) show higher concept-channel importance relative to the side-channel's across all datasets. Also, S-CREAM exhibits larger CCI values compared to H-CREAM in FMNIST.

Dataset	**Model**	$CCI \uparrow$	$PCI \uparrow$	$PSI \downarrow$
iFMNIST	H-CREAM	0.79 ± 0.01	0.61 ± 0.09	0.35 ± 0.00
	S-CREAM	0.80 ± 0.02	0.59 ± 0.06	0.35 ± 0.00
cFMNIST	H-CREAM	0.88 ± 0.02	0.65 ± 0.05	0.12 ± 0.06
	S-CREAM	0.94 ± 0.01	0.72 ± 0.03	0.03 ± 0.01
CUB	S-CREAM	0.96 ± 0.00	0.72 ± 0.00	0.01 ± 0.00
CelebA	S-CREAM	0.98 ± 0.03	0.30 ± 0.03	0.00 ± 0.00

Concept channel importance (CCI) can also be expressed is terms of total predictive power $v_f(D)$:

$$CCI = \frac{\phi_c(v_f)}{\phi_c(v_f) + \phi_y(v_f)} \overset{\text{Efficiency}}{=} \frac{\phi_c(v_f)}{v_f(D)}$$

where $\phi_c(v_f), \phi_y(v_f)$ denote the SAGE value of the whole concept-channel and side-channel respectively.

B.1.1 Deriving the desired importance threshold

We desire the importance of the concept channel to be greater or equal to the importance of the side channel, i.e., $\phi_c(v_f) > \phi_y(v_f)$. Assuming that *both channels are informative* (i.e., $\phi_c(v_f), \phi_y(v_f) > 0$), we derive a desired lower threshold for the concept channel importance:

$$\begin{aligned}
& \phi_c(v_f) \geq \phi_y(v_f) && \text{add } \phi_c(v_f) \text{ to both sides} \\
\iff & 2\phi_c(v_f) \geq \phi_y(v_f) + \phi_c(v_f) && \text{assuming } \phi_c(v_f), \phi_y(v_f) > 0 \\
\iff & \frac{1}{2\phi_c(v_f)} \leq \frac{1}{\phi_y(v_f) + \phi_c(v_f)} && \text{multiply both sides with } \phi_c(v_f) \\
\iff & \frac{\cancel{\phi_c(v_f)}}{2\cancel{\phi_c(v_f)}} \leq \frac{\phi_c(v_f)}{\phi_y(v_f) + \phi_c(v_f)} \\
\iff & \frac{1}{2} \leq CCI
\end{aligned}$$

Meanwhile, if the side channel is uninformative, then due to the dummy property $\phi_y(v_f) = 0$, then $CCI = 1$. For the sake of completion, in the edge case where the side-channel makes the prediction less accurate, i.e., $\phi_y(v_f) < 0$, then $CCI \in (-\infty, 0) \cup (1, +\infty)$. Note that we do not notice such cases in our experiments. Lastly, $CCI = 0$, if the concepts are not used by the Concept-Task Block.

In conclusion, assuming that both SAGE values are positive, CCI is bounded between $CCI \in [0, 1]$. A CCI value close to 1 indicates that the model relies primarily on the concept channel, reinforcing interpretability. When $CCI \approx 0.5$, it suggests that both the concept and side-channels contribute equally to the predictions. To compute CCI, we evaluate the trained model on the entire test set, storing the classifier's inputs to estimate feature attributions.

B.2 Permutation Feature Importance

Permutation Feature Importance (PFI) [9, 17] measures the significance of a feature by evaluating how much the model's accuracy deteriorates when its values are randomly shuffled. A feature is considered important if permuting its values leads to a significant drop in accuracy, whereas an unimportant feature results in little to no change. The PFI score for a feature j is computed as:

$$PFI_j = ACC_Y - \frac{1}{K} \sum_{k=1}^{K} ACC_{Y_{j,k}} \tag{16}$$

where ACC_Y is the test accuracy of the model, and $ACC_{Y_{j,k}}$ is the accuracy after randomly permuting the values of feature j for the k-th iteration. In our case, we focus on evaluating the importance of entire channels (i.e., groups of features). Thus we permute all values within each channel (concept channel and side-channel) simultaneously. We denote concepts' feature importance as **P**ermutation **C**oncept **I**mportance (PCI) and the side-channel's as **P**ermutation **S**ide-Channel **I**mportance (PSI). We measure PFI on the test dataset [39], and we permute for 100 iterations. Lastly, as mentioned, we desire the concept-channel to be more important than the side-channel, i.e., $PCI > PSI$. This inequality plays the same role as the importance threshold $CCI > 0.5$ derived in B.1.1. As seen in Table 3, the PFI-based metrics also indicate that CREAM prioritizes the concept-channel instead of the side-channel.

B.3 Dropout Rate and PFI

In Fig. 7, we present the Permutation Feature Importance curves for all datasets and models , when trained with different dropout rates (p). For each p, we train a model with it, and then we plot its PCI and PFI, showing how much does the test accuracy drop if we randomly permute the concept-concept and side-channel, respectively. As expected, the findings are consistent with the observed trends in Section 5.2. Increasing the dropout rate p leads to increased PCI and decreased PSI. Also, the need for side-channel regularization decreases when using complete concept sets. The point where $PCI > PSI$, in iFMNIST, is around $p = 0.75$, but when moving to the complete FMNIST case it drops to around $p = 0.3$. One difference between the PFI metrics and CCI, is the lowest required dropout rate p such that the side-channel stops dominating the concept-concept block (i.e., $CCI > 0.5$ or $PCI > PSI$). Across all datasets, the PFI-based approach suggests that we should regularize the side-channel more than CCI suggests.

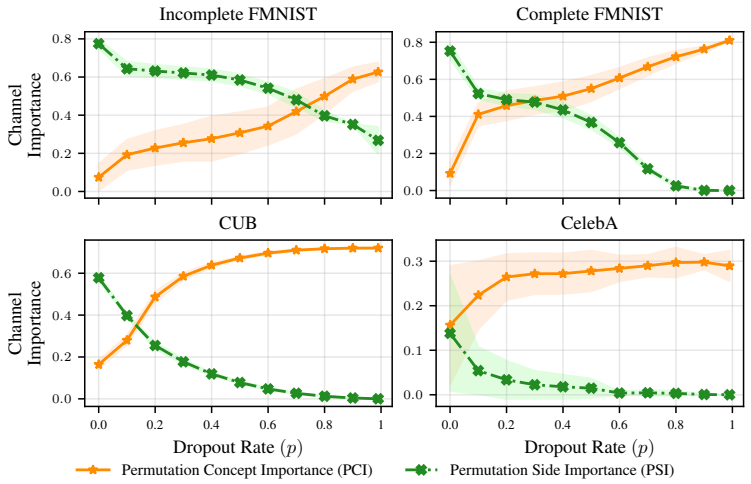


Figure 7: Mean values $\pm$ standard deviation of the permutation feature importance of both channels, in all datasets for S-CREAM, averaged over 5 seeds. Increasing p leads to an increase in PCI and decreases PSI.

C Dataset Details

A summary of the datasets used in our experiments is provided in Table 4.

C.1 Dataset Description

Apparel Classification (FashionMNIST) The FashionMNIST (FMNIST) [6] dataset [60] consists of 70,000 grayscale images of clothing items across 10 classes. However, FMNIST does not provide predefined concept annotations or a dependency graph. To define concepts, we use high-level apparel categories derived from the hierarchical structure in [52]. The resulting dependency graph G forms a hierarchical tree, as shown in Figure 1.

[6] `https://github.com/zalandoresearch/fashion-mnist`, MIT License

Table 4: Summary of each dataset. A concept group consists of a mutually exclusive group of concepts. C_{direct} refers to the number of concepts directly connected to any task, which coincides with the upper bound of interventions. The reported number of effective group interventions refers to the number of directly connected groups of concepts instead.

Dataset	Number of Classes (L)	Number of Concepts (K)	C_{direct}	Mutex Groups	Effective Group Interventions
iFMNIST	10	8	6	2	1
cFMNIST	10	11	9	3	2
CUB	200	112	112	27	27
CelebA	2	7	7	7	7

We call this dataset Incomplete FMNIST. Since in hierarchical classification the classes at each level are mutually exclusive, the concepts of the same depth of the tree are also mutually exclusive. This means that the mutex concepts are grouped as such: {Clothes, Goods} and {Tops, Bottoms, Dresses, Outers, Accessories, Shoes}, leading to two groups of concepts. Note that, in the hierarchical classification setting the second group would be split into: {Tops, Bottoms, Dresses, Outers} and {Accessories, Shoes}. However, if we applied that logic to our case, both concept groups can be active at any time, thus being one mutex group. Lastly, these concepts cannot fully predict the classes. For instance, the same "active" concept vector $c = \{$Clothes, Tops$\}$ is used to predict these three classes $y = \{$T-shirt, Pullover, Shirt$\}$ with no way of distinguishing between them. The same problem applies for $c = \{$Goods, Shoes$\}$ and the classes $y = \{$Sandal, Sneaker, Ankle Boot$\}$. This is where the side-channel shines; with the extra information from it, we can now successfully predict all classes.

For the Complete FMNIST dataset, we add a few more concepts to the hierarchical tree to make it a complete set. These concepts represent seasonality and are: {Summer, Winter, Mild Seasons}. We consider them to be mutually exclusive. The updated dependency graph is shown in Fig. 1.

Bird identification (CUB) The Caltech-UCSD Birds-200-2011 (CUB) [7] dataset [59] consists of 11,788 natural images spanning 200 bird species. Each image is annotated with 312 binary concepts describing visual characteristics such as color, shape, length, pattern, and size. Following [27], we process concept labels via majority voting across instance-level annotations and remove excessively sparse concepts, resulting in $K = 112$ concepts. These binary concepts originate from categorical attributes, leading to 27 mutually exclusive groups, similar to one-hot encoded features. To define concept relationships in G_C, we assume that concepts related to the same type of attribute (e.g., all colors, all patterns) are interconnected via bidirected edges i.e. the concepts that pertain to the same "type" of feature are related, but we do not know its direction. These types of features are: {Shape, Color, Length, Pattern, Size}. Thus, the colors, patterns, etc. of all body parts are related. This structured representation embeds reasoning by enforcing relationships among semantically related features. Furthermore, we assume that all concepts directly influence all classes, ensuring that the model can fully leverage fine-grained feature representations.

Table 5: Test accuracy of $C_{true} \rightarrow Y$ linear models. They assess the predictive power of concepts for each dataset.

Dataset	Test ACC_Y
iFMNIST	60.00
cFMNIST	100.00
CUB	100.00
CelebA	84.09

Smile Detection (CelebA) The CelebA [8] dataset [30] comprises over 200.000 celebrity face images annotated with 40 facial attributes. We select $K = 7$ facial attributes as concepts: ({Arched Eyebrows, Bags Under Eyes, Double Chin, Mouth Slightly Open, Narrow Eyes, High Cheekbones, Rosy Cheeks}), and use "Smiling" as the target label, making it a binary classification problem. The corresponding reasoning graph (G) is illustrated in Figure 3.

C.2 Baseline performance for the different datasets

Additionally, in Table 5 we report the test accuracy achieved by a simple linear layer trained to predict the target classes directly from concepts ($C_{true} \rightarrow Y$), offering insight into the predictive power of concepts alone. Specifically, this linear model is trained with similar settings as the ones mentioned in

[7] `https://www.vision.caltech.edu/datasets/cub_200_2011/`

[8] `https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html`

Table 7, and uses as input the ground-truth concepts. Its purpose is to serve as a baseline for the CBM models; all CBMs mentioned in Section 5 also use a linear layer as their concept to task classifier. Keep in mind that, for more complex tasks (such as the OAI dataset in [27]) deeper networks have been shown to be a better solution.

D Implementation Details

This section provides additional details on the implementation of our experiments. A more detailed version of CREAM's illustration is shown in Fig. 8. For datasets containing mutually exclusive concepts, we apply a softmax activation as described in Section 3. Our framework is implemented in PyTorch (v2.4.0) [41] and PyTorch Lightning (v2.3.0). We use 20% of the stored activation values to perform the missing value imputation in CCI, and set SAGE's convergence threshold to 5×10^{-2}. The Permutation Feature Importance (PFI) metric is computed by permuting the channel values 100 times. All reported experiments use five different seeds. Lastly, for all ablation studies including the dropout rate, we use these p values: {0.0001, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99}.

Details per Dataset Our experiments utilize three datasets, each with tailored architectures and preprocessing steps. We use the Adam optimizer [26] for all models. For FashionMNIST (FMNIST), we employ a lightweight CNN backbone with two convolutional layers, ReLU activations, Max Pooling, dropout, and a final linear layer. The dataset is split into $50k - 5k - 10k$ for training, validation, and testing, respectively. The backbone is trained for 50 epochs, using a learning rate of 10^{-3} and a batch size of 256. Standard normalization is applied to the dataset. In the experiments of the main text, the number of epochs was also set to 50. Lastly, the standard model was trained from scratch for the same number of epochs.

For CUB and CelebA, we use ImageNet ResNet-18 [23] as the backbone. CUB follows the same train-test splits, concept processing, and image preprocessing as in [27]. The backbone is fine-tuned for 50 epochs, with a learning rate of 10^{-4} and a batch size of 64. For CelebA, we fine-tune ImageNet ResNet-18 on a 5K image subset for 90 epochs, using a learning rate of 10^{-4} and a batch size of 256, following [61]. Both datasets undergo identical preprocessing: color jittering, random resized cropping, horizontal flipping, and normalization. In the experiments of the main text, the number of epochs for CUB was 300, while in CelebA we used early stopping with a patience of 15 epochs, and the maximum number of epochs was 200. Lastly, in CUB the standard model was trained from scratch for the same number of epochs, while in CelebA we used the pretrained backbone as a starting point.

Model Selection In all cases, the side-channel consists of a Linear layer with a ReLU activation. We perform grid search over hyperparameters (Table 6). The Masked MLP depth refers to the number of hidden layers in the masked algorithm from Zuko [46]; a depth of zero means the Masked MLP is equivalent to a Masked Linear layer. Given that CBMs require multiple selection criteria, we adopt a ranking-based approach that averages performance across task and concept accuracies on the validation set, as in [50]. Table 7 reports the best hyperparameter configurations for each dataset. Note that we eventually selected the models with the highest dropout rate in each case. Here, $d_C K + d_Y$ represents the total latent space dimension, while d_Y refers to the side-channel input size. The depth (number of hidden layers) of the masked MLP is d. For the CBM model, in both iFMNIST and cFMNIST we used a learning rate of 10^{-2}, and $\lambda = 1$.

Table 6: Hyperparameter search space explored for each dataset.

Dataset	iFMNIST	cFMNIST	CUB	CelebA
Dropout (p)	$\{0.2, 0.5, 0.8, 0.9\}$	$\{0.2, 0.5, 0.8\}$	$\{0.1, 0.2, 0.5, 0.8\}$	$\{0.2, 0.5, 0.8\}$
$d_C K + d_Y$	$\{8, 18, 76, 78, 128\}$	$\{11, 21, 22, 32, 43, 128\}$	$\{424, 512, 648, 848\}$	$\{7, 8, 36, 256, 512\}$
d_Y	$\{0, 10, 20, 30, 64\}$	$\{0, 10, 40, 62\}$	$\{64, 176, 200, 400\}$	$\{0, 1, 123, 162\}$
d	$\{0, 2\}$	$\{0, 2\}$	$\{0, 3\}$	$\{0, 5\}$

Resource Usage All experiments were conducted on a high-performance computing cluster with automatic job scheduling, ensuring efficient resource allocation. For FMNIST (both iFMNIST and cFMNIST), we utilized 2 CPU workers, 8GB RAM, and a GPU with at least 4GB of VRAM. For both datasets, model training required on average 3 minutes. For CUB, the setup included 8 CPU workers, 16GB RAM, and a GPU with at least 4GB of VRAM, and training and validation required on average 31 minutes. The CelebA experiments were more resource-intensive, requiring 12 CPU

Table 7: Configurations for the best-performing models in each dataset. Note that $d_C K + d_Y$ represents the dimensionality of latent space that is split, and d_Y the dimensionality of the input to the side-channel. H-CREAM and S-CREAM refers to the models with hard and soft concept representations respectively. The best models have $d = 0$ number of hidden layers in the Concept-Concept Block.

Dataset	Model	λ	lr	p	$d_C K + d_Y$	d_Y
iFMNIST	H-CREAM	1	1e-3	0.9	78	30
	S-CREAM	1	1e-3	0.9	76	20
cFMNIST	H-CREAM	1	1e-3	0.8	43	10
	S-CREAM	1	1e-3	0.8	128	40
CUB	S-CREAM	1	1e-4	0.8	648	200
CelebA	S-CREAM	1	1e-3	0.8	512	1

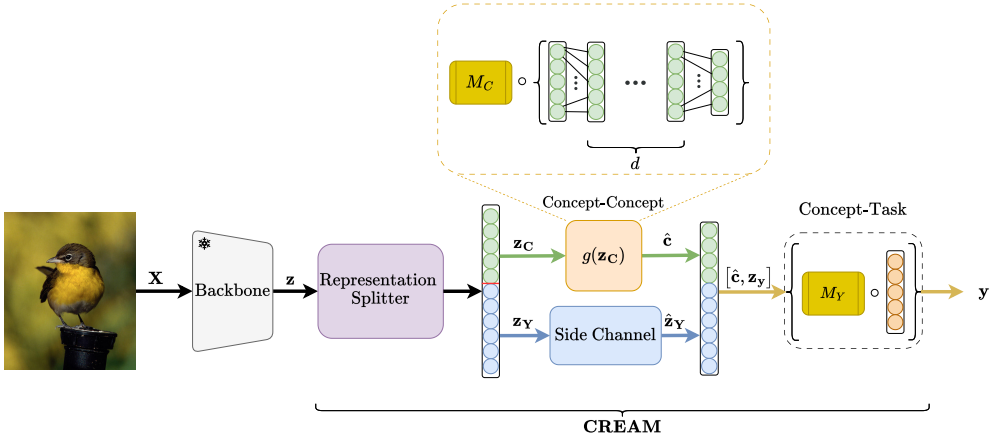


Figure 8: Expanded version of the illustration of CREAM found in the main text, providing additional details for clarity and completeness.

workers, 32GB RAM, and a GPU with at least 8GB of VRAM, and took on average 44 minutes. The aforementioned resources refer to CBM and CREAM. For those experiments in Section 5 we estimate a total of 14 hours of compute. For ACBM and SCBM execution time, we refer the reader to the corresponding Appendix section in [58].

E Additional Results

E.1 Model Performance Without Side-Channel

In this section we investigate CREAM's performance without including a side-channel. From the hyperparameter search performed in Section D, we pick the best models without a side channel ($d_Y = 0$ values from Table 6). The hyperparameters of these models are seen in Table 8.

Table 8: Hyperparameter configurations for the best-performing models (without a side-channel) in each dataset. The best models have $d = 0$ number of hidden layers in the Concept-Concept Block.

Dataset	Model	λ	lr	$d_C K + d_Y$
iFMNIST	CREAM	1	1e-3	128
cFMNIST	CREAM	1	1e-3	22
CUB	CREAM	1	1e-4	112
CelebA	CREAM	1	1e-3	7

We report the concept and task accuracies of the best performing models without a side-channel in Table 9. According to Table 9, CREAM performs similarly to the purely interpretable CBMs

reported in Table 1. These results reinforce our previous claims that enforcing expert reasoning does not lead to decreased performance. Oddly enough, in CelebA, the CREAM model without a side-channel is outperforming every other reported model in task accuracy. This is due to: (i) our model selection scheme leading (here) to increased task accuracy and a decreased concept accuracy, (ii) the side-channel in CelebA having an output size of 1 ($|\hat{\mathbf{z}}_Y| = 1$), thus being less predictive.

Table 9: Concept and Task accuracy for the best-performing CREAM models (without a side-channel) in each dataset across 5 seeds. These models perform on-par with the rest of the models. In all cases, removing the side-channel leads to decreased task accuracy. CelebA is an exception, where concept accuracy drops but task accuracy increases.

Dataset	**Test** ACC_Y	**Test** ACC_C
iFMNIST	57.10 $\pm$ 0.05	99.07 $\pm$ 0.02
cFMNIST	92.31 $\pm$ 0.15	97.52 $\pm$ 0.25
CUB	71.13 $\pm$ 0.28	85.65 $\pm$ 0.06
CelebA	81.02 $\pm$ 0.56	78.59 $\pm$ 0.79

E.2 Group Interventions

In this section we examine the benefits of the `C-C` mutually exclusive relationships, on the efficiency of interventions. Specifically, we investigate group interventions, where contrary to individual interventions, the human expert intervenes on a group of mutually exclusive concepts simultaneously. For instance, in cFMNIST, a human expert would change the value of all concepts belonging in Mutex 3 (“Summer”, “Winter” and “Mild Seasons”) with just one intervention, by "activating" the concept “Summer” and the rest being deactivated automatically, since they are mutually exclusive. Note that, in the case of group interventions the upper bound of interventions drops even lower, down to the number of directly connected concept mutex groups. As visualized in Fig. 9, in FMNIST, we observe that task performance peaks after one group intervention in the incomplete setting and two group interventions in the complete setting. For both cases, the one remaining intervention, does not improve accuracy since those are the indirect mutex concepts (“Clothes” and “Goods”). Similar conclusions are seen in CUB; with about 20 group interventions we achieve performance gains of almost 90 individual interventions. Lastly, group interventions lead to the same performance gains as individual interventions, but with less human effort. These findings indicate that group interventions on mutually exclusive concepts, identified through `C-C` relationships, can help scale the intervention procedure.

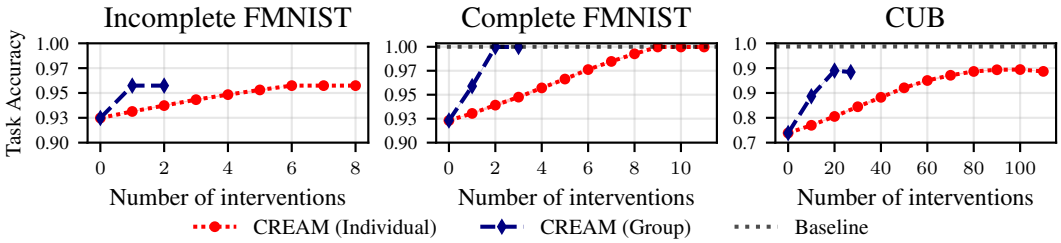


Figure 9: Impact of group interventions compared to individual ones, on task accuracy. Group interventions lead to the same performance gains with lower number of interventions. The number of relevant group interventions in CREAM for each dataset are, from left to right, 1, 2 and 27. Note the baseline’s performance is not visualized in iFMNIST, as it is comparatively too low.

E.3 Correlation of variables

In an effort to investigate the effect of different types of relationships on the model representations, we investigate their statistical relationships. Specifically, we visualize the correlations between the concept exogenous variables and the input to the side-channel. As seen in Fig. 10, in the FMNIST cases, where the `C-C` relationships create a DAG, the exogenous variables of the concepts form blocks in the correlation matrix. Each of the d_C dimensions seem to be strongly correlated with the rest of the dimensions of the exogenous variable they belong to. Furthermore, we can observe

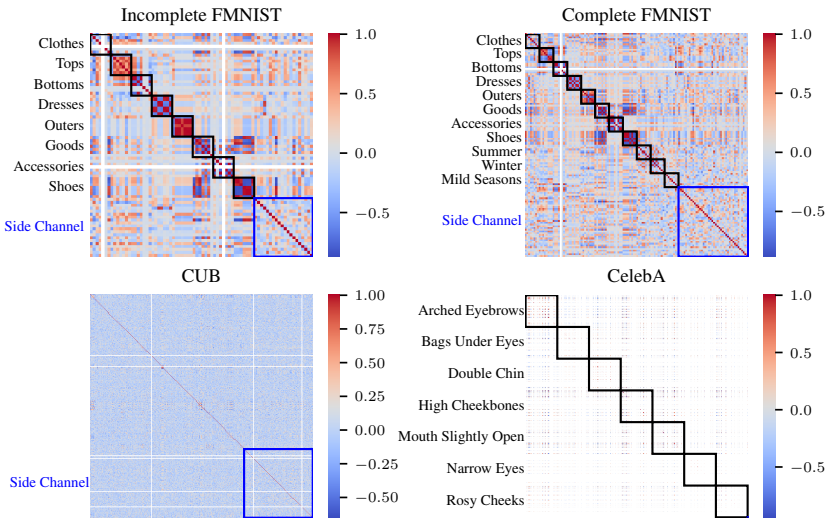


Figure 10: Correlation matrix for the CREAM models reported in Table 2. For each concept we draw a box around its assigned d_C dimensions. Note that CelebA's side-channel and CUB's exogenous are too small to plot. We notice that `C-C` reasoning leads block-like correlations, both within the exogenous of each concepts, and between their exogenous, revealing the hierarchical structure.

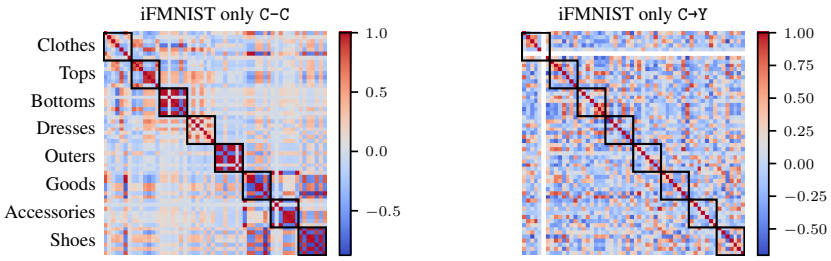


Figure 11: Correlation matrix for the CREAM models reported in Table 2. For each concept we draw a box around its assigned d_C dimensions. We notice that `C-C` reasoning leads block-like correlations, both within the exogenous of each concepts, and between their exogenous, revealing the hierarchical structure.

the hierarchical structure of the concepts. For example, the exogenous of "Goods" is also strongly correlated to "Accessories" and "Shoes". Meanwhile, in CUB, where the `C-C` relationships create cycles in G_C, we do not observe any particular structure. In addition, CelebA exhibits a lot of variables with zero variance; they represent "dead" neurons. Lastly, in all cases, the side-channel also does not show a particular structure.

Further investigating the block-diagonal structure We will now focus on identifying the cause of the near block-diagonal structure in the correlation matrices. Following the leakage experiments from Table 2, we visualize the correlations when the model reasoning is removed. According to Fig. 11, we identify `C-C` as the architectural choice leading to the near block-diagonal structure of the correlation matrix.

E.3.1 Variance of the side channel

Here, we will also check if some side-channel nodes are dead, since we set the output of the side-channel to be of dimension L, which leads to an excess of needed nodes. For instance, in the iFMNIST dataset, the classes: {Trouser, Dress, Coat, Bag} are fully distinguishable from their assigned concepts, meaning that they do not need the side-channel. Thus, it would be preferable if their assigned side-channel nodes had a constant output. To verify this, we visualize the covariance matrix of the side-channel ($\hat{\mathbf{z}}_Y$), and we inspect its main diagonal. Specifically, for CelebA, $L = 1$ and thus there is only one side-channel node, we report here its variance: $Var_{\hat{\mathbf{z}}} = 0.137095$. As seen

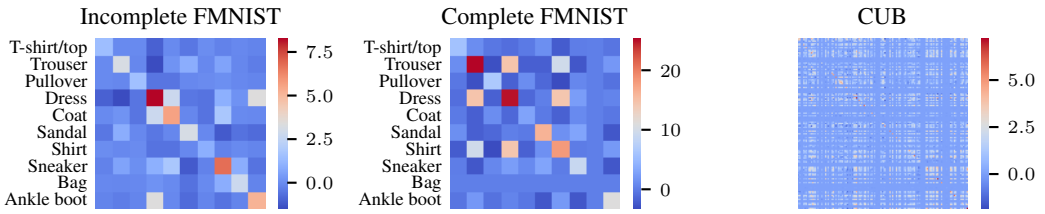


Figure 12: Covariance matrix of the outputs of the side-channel for CREAM. On the left, we report the corresponding class each node is assigned to. CUB has too many classes to clearly visualize.

in Fig. 12, some of the nodes in iFMNIST (e.g., for "Trouser") and cFMNIST (e.g., for "Bag") have a low variance.

F Ablation studies

In this section, we present additional experiments to offer a deeper understanding of CREAM's hyperparameters. We conduct ablation studies on multiple hyperparameters to illustrate their impact on different aspects of model performance, supplementing the main text with further insights.

F.1 Effect of Dropout Rate on Accuracy

In Fig. 13, we examine the impact of the dropout rate on test task and concept accuracies. Specifically, the x-axis represents the dropout rate $p \in (0, 1)$ applied to the side-channel during model training, while the y-axis indicates the model's accuracies on the test set. We notice that the test concept accuracy is not affected by the dropout rate, while the test task accuracy generally marginally drops when increasing the dropout probability. Surprisingly, even with an extreme dropout rate of $p \approx 1$, CREAM maintains black-box level accuracy on iFMNIST.

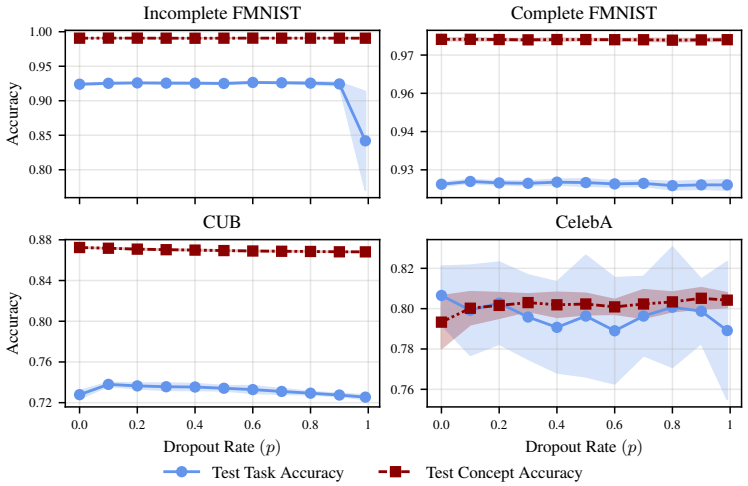


Figure 13: Concept and Task accuracies across different dropout rates (p). Values are presented as mean $\pm$ standard deviation across 5 seeds. Increased p leads to marginal drops in task accuracy.

F.2 Effect of Dimensionality of Exogenous Variables

As mentioned in Section 3.2 each exogenous variable $\mathbf{z}_C$ is of dimension d_C. Here we study the effect of d_C on model performance (as seen in Fig. 14), for $d_C \in \{1, 2, 3, 7, 10\}$. Note that, we keep every other hyperparameter to the same values as in Table 7. This means, we are increasing the dimensionality of the output of the representation splitter, while keeping $|\mathbf{z}_Y|$ constant.

We observe a general trend across all datasets; increasing the dimensionality d_C of the exogenous variables of the concepts leads to increased performance. That increase of performance is sometimes

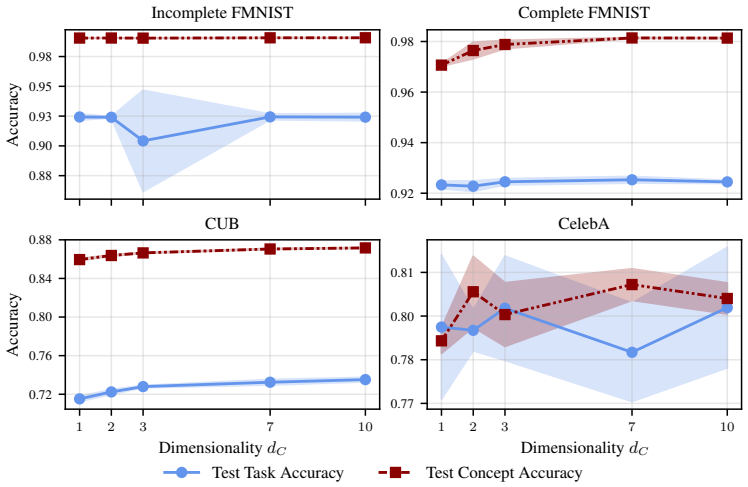


Figure 14: Concept and Task accuracies across different d_C values for the exogenous of the concepts. Values are presented as mean $\pm$ standard deviation. Increasing $\mathbf{z}_Y$, generally improves performance.

in the form of increased concept accuracy, task accuracy, or both at the same time. However, the relative boost in accuracies can sometimes be negligible.

F.3 Effect of the depth of Concept-Concept Block

In Fig. 15 we study the effect of increasing the depth d of the Concept-Concept Block to model performance. For each model we keep the same configuration as the main text, but now $d \in \{0, 1, 2, 3, 5\}$. We notice a trend across all models; increasing the depth leads to equal or worse performance. These findings further support the notion that human-interpretable concepts are often linearly encoded in the latent space of neural networks [45].

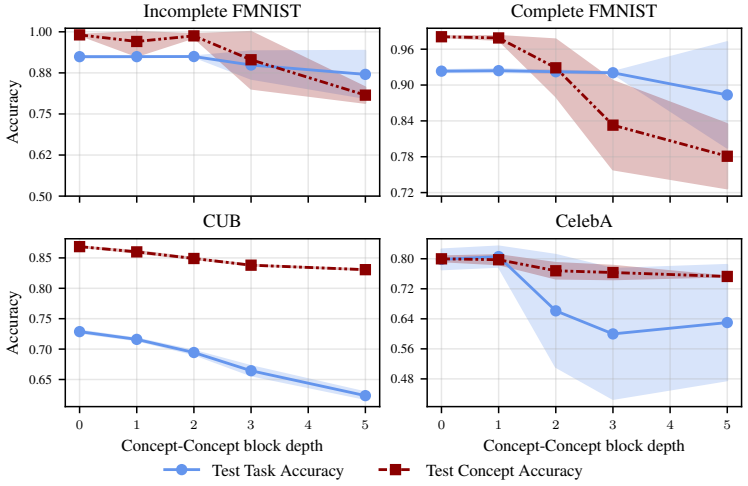


Figure 15: Concept and Task accuracies across different d values in the Concept-Concept block. Values are presented as mean $\pm$ standard deviation. Increasing d of the `C-C` block degrades performance.

G Hard models

In this section we will study CREAM with hard concept representations, i.e., $\hat{C} \in \{0, 1\}$. To ensure trainability in the hard case, we leverage the straight-through estimator (STE) [6], which approximates gradients during the backward pass. The representation of hard and soft concepts is given as:

$$\hat{C} := \begin{cases} \sigma(\hat{l}_C) & \text{S-CREAM} \\ round(\sigma(\hat{l}_C)) & \text{H-CREAM} \end{cases} \tag{17}$$

where H-CREAM and S-CREAM are for hard and soft concepts respectively. We also handle mutually exclusive concepts similarly. The hyperparameter configurations of H-CREAM, can be found in Table 7. Hard CBM models are sometimes preferred due to them not suffering from leakage [21, 58] and being easier to intervene on.

G.1 Concept and Task Accuracy

S-CREAM consistently outperforms its hard variant across all datasets and evaluation metrics. This suggests that soft concept representations provide greater flexibility and are easier optimize, allowing the model to capture useful variations in concept values while maintaining structured reasoning.

Table 10: Performance comparison between CREAM variants. Reported values represent the mean and standard deviation over five seeds. S-CREAM outperforms H-CREAM performance and interpretability in both FMNIST variants. The best-performing model variant is highlighted in **bold**.

Dataset	**Model**	**Test** ACC_Y	**Test** ACC_C
iFMNIST	Standard	92.7 $\pm$ 0.07	-
	S-CREAM	**92.43** $\pm$ 0.23	**99.07** $\pm$ 0.03
	H-CREAM	92.05 $\pm$ 0.29	98.99 $\pm$ 0.02
cFMNIST	S-CREAM	**92.38** $\pm$ 0.16	**98.08** $\pm$ 0.06
	H-CREAM	88.18 $\pm$ 4.92	94.97 $\pm$ 2.83

G.2 Importances

Table 3 demonstrates that, CCI remains consistently above the critical 0.5 threshold, and PCI exceeds PSI, for the selected hyperparameter settings. This indicates that the hard variant can also fulfill our concept importance desiderata. Furthermore, soft models report a higher CCI value compared to their hard counterparts, indicating that their decisions are more strongly influenced by the concepts.

G.3 Interventions

We compare task accuracy between the two CREAM variants, after individual concept interventions. We follow the same process as the one mentioned in Section 5. For H-CREAM we do not have to find the 5th and 95th percentiles, and we instead directly use the ground truth concepts. As seen in Fig. 16, H-CREAM shows the same behavior as S-CREAM. However, the latter slightly outperforms the former in both presented cases.

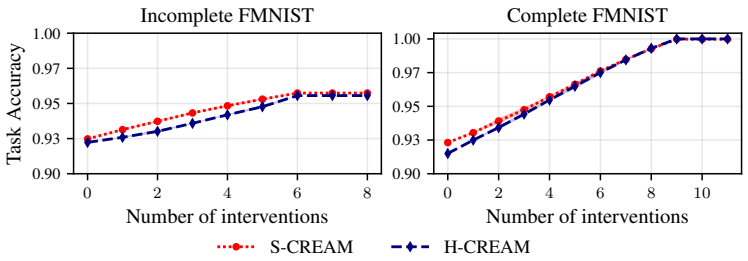


Figure 16: Intervenability comparison of H-CREAM and S-CREAM. The latter slightly outperforms the latter. Both models' task accuracy peaks after 6 and 9 interventions in iFMNIST and cFMNIST respectively.

G.4 Correlations

We also investigate the correlations between the concept exogenous variables and the input to the side-channel in the hard variant. As illustrated in Fig. 17, H-CREAM does not exhibit the block correlations that the soft model does (Fig. 10). Also, in H-CREAM more concept exogenous variables have become constants. This indicates that the hard variant requires a lower number of exogenous variables, which is also in line with the best hyperparameters seen in Table 7.

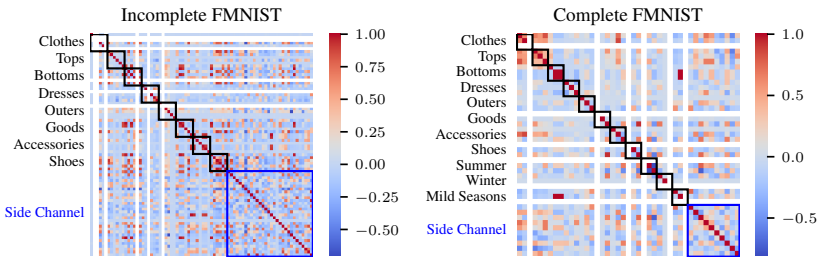


Figure 17: Correlation matrices of the output of the representation splitter ($\mathbf{z}$) for H-CREAM. We notice that more concept exogenous variables have become constants.

H Logic viewpoint of CREAM

In this section we will express the relationships in G using description logic [2]. Note that in CREAM the concepts are calculated using the exogenous variables, and the tasks using the concepts and their corresponding side-channel input. Table 11 shows some of the logic rules that match CREAM's calculations, with a slight abuse of notation.

Table 11: Some logic rules based on description logic for CREAM. The $\mathbf{z}$ variables correspond to the concept and side-channel, and $\bot$ denotes the empty set.

Dataset	
iFMNIST	Clothes $\leftarrow \mathbf{z}_{Clothes}$ Goods $\leftarrow \mathbf{z}_{Goods}$ Clothes $\sqcap$ Goods $\sqsubseteq \bot$ Tops $\sqcap$ Bottoms $\sqsubseteq \bot$ Tops $\sqcap$ Dresses $\sqsubseteq \bot$ $\vdots$ Accessories $\sqcap$ Shoes $\sqsubseteq \bot$ Tops $\leftarrow \mathbf{z}_{Clothes} \sqcap \mathbf{z}_{Tops}$ $\vdots$ Shoes $\leftarrow \mathbf{z}_{Goods} \sqcap \mathbf{z}_{Shoes}$ T-shirt $\sqcap$ Pullover $\sqsubseteq \bot$ T-shirt $\sqcap$ Shirt $\sqsubseteq \bot$ $\vdots$ Sneaker $\sqcap$ Ankle Boot $\sqsubseteq \bot$ T-shirt $\leftarrow$ Tops $\sqcap \mathbf{z}_{T-shirt}$ $\vdots$ Ankle Boot $\leftarrow$ Shoes $\sqcap \mathbf{z}_{AnkleBoot}$
CelebA	Bags Under Eyes (BUE) $\leftarrow \mathbf{z}_{BUE}$ High Cheekbones (HC) $\leftarrow \mathbf{z}_{HC}$ Mouth Slightly Open (MSO) $\leftarrow \mathbf{z}_{MSO}$ Rosy Cheeks (RC) $\leftarrow \mathbf{z}_{RC}$ Arched Eyebrows (AE) $\leftarrow \mathbf{z}_{AE}$ Double Chin (DC) $\leftarrow \mathbf{z}_{DC}$ Narrow Eyes (NE) $\leftarrow \mathbf{z}_{MSO} \sqcap \mathbf{z}_{BUE} \sqcap \mathbf{z}_{HC} \sqcap \mathbf{z}_{NE}$ Smiling $\leftarrow BUE \sqcap HC \sqcap MSO \sqcap NE \sqcap RC \sqcap DC \sqcap AE \sqcap z_{Smiling}$