VCR: A Tabular Data Slicing Approach to Understanding Object Detection Model Performance [Scalable Data Science]

Jie Jeff Xu
Georgia Institute of Technology
jxu680@gatech.edu

Saahir Dhanani
Georgia Institute of Technology
saahir@gatech.edu

Jorge Piazentin Ono
Bosch Research North America
jorge.piazentinono@us.bosch.com

Wenbin He
Bosch Research North America
wenbin.he2@us.bosch.com

Liu Ren
Bosch Research North America
liu.ren@us.bosch.com

Kexin Rong
Georgia Institute of Technology
krong@gatech.edu

ABSTRACT

Slice discovery methods (SDMs) are valuable tools for identifying semantically meaningful subsets of data where machine learning models may exhibit systematic errors. While extensively explored in image classification tasks, existing SDMs face limitations when applied to object detection tasks. Object detection demands a finer understanding of images at the segment level rather than the entire image and involves additional detection-specific metadata such as bounding box size and spatial relationships with nearby objects.

This paper introduces VCR, the first automated slice discovery framework designed for object detection tasks. Leveraging the capabilities of vision foundation models, VCR generates segment-level visual concepts, which serve as explanation primitives for diagnosing smaller, domain-specific vision models. The visual concept extraction process is model-agnostic and unsupervised, facilitating its application across diverse scenarios. VCR integrates visual concepts with metadata in a tabular format and uses a scalable frequent itemset mining-based technique to identify common patterns associated with model performance. We evaluate VCR through a comprehensive evaluation benchmark involving 1713 slice discovery settings, as well as a user study with six industry machine learning scientists and engineers. Together, these evaluations demonstrate the usability and scalability of VCR.

PVLDB Reference Format:
Jie Jeff Xu, Saahir Dhanani, Jorge Piazentin Ono, Wenbin He, Liu Ren, and Kexin Rong. VCR: A Tabular Data Slicing Approach to Understanding Object Detection Model Performance [Scalable Data Science]. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/d2i-lab/VCR.

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

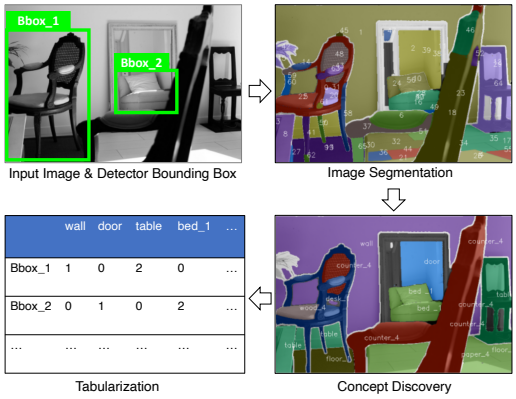


Figure 1: VCR extracts structured information from input images describing interactions between visual concepts and bounding boxes, and uses a tabular data slicing approach to summarize the common patterns that affect the object detection model's performance.

1 INTRODUCTION

Systematic errors made by computer vision models, where models perform significantly worse on a subset (or slice) of the data despite having good average performance, are becoming an increased concern for practitioners [38]. For example, variations in object recognition accuracy of up to 20% have been observed between images taken in countries with different income levels, due to objects (*e.g.*, a toothbrush) appearing in different contexts (*e.g.*, inside versus outside a bathroom) [18]. Similar performance gaps have been observed in various applications including object recognition [8, 51], image classification [12, 47], and medical diagnosis [10, 19]. Detecting these systematic errors could help guide practitioners to update training datasets and mitigate unwanted biases in models [31, 55].

While data slices in structured, tabular datasets can be easily defined by attribute-value pairs such as $\{age < 20, gender = Female\}$, identifying coherent and semantically meaningful slices in unstructured image datasets is considerably more challenging. While prior works have explored various explanation primitives for image classification tasks, they fall short in object detection tasks for two reasons. First, unlike image classification which assigns a single label to an entire image, object detection locates and identifies individual objects or regions within the image, which requires fine-grained explanation primitives capable of understanding individual image segments. Therefore, existing methods that use image-level

embeddings [23] or rely on internal states of models (*e.g.*, activation layers of convolutional neural networks) [17, 20, 43, 49] do not suffice. Second, object detection contains a rich set of metadata attributes, such as the size and position of bounding boxes, that can be useful in explaining model behaviors, but are absent in image classification tasks. For example, small and occluded objects could cause object detection models to perform poorly.

In this work, we introduce VCR, an automated slice discovery framework that identifies and summarizes data slices where object detection models underperform in a human-interpretable manner. VCR capitalizes on recent advances in vision foundation models, such as Meta's Segment Anything Model (SAM) [33], that have significantly improved the granularity and quality of our image understandings. Inspired by the success of slice finding methods for tabular datasets, VCR leverages these pre-trained foundation models to extract structured information from images, which can augment the metadata attributes and be used as dimensions to define data slices. Figure 1 shows an overview of VCR's workflow.

Specifically, VCR uses foundation models to extract structured data in the form of visual concepts, which are semantically meaningful image segments such as objects (*e.g.*, car) and object parts (*e.g.*, wheel). VCR explains model behaviors using interactions between visual concepts and object detection models' bounding box predictions. For example, VCR may explain a set of poor detection results for the car class with the presence of the visual concept "pole" that occludes the view. Both the extraction and labeling of the visual concepts are unsupervised and model-agnostic, making them widely applicable. VCR then combines visual concepts and additional metadata in a tabular format, which allows it to identify problematic data slices using a frequent itemset mining-based method. We additionally introduce pruning optimizations that significantly enhance the scalability of mining for concept absence, allowing VCR to handle hundreds of concepts at interactive time.

In summary, our key contributions are the following:

- We develop VCR, the first automated slice discovery framework for object detection tasks. VCR novelly combines visual concept discovery with tabular data slicing techniques to enable the discovery of interpretable data slices at scale.
- We introduce a visual concept discovery mechanism that utilizes pre-trained foundational vision models for unsupervised concept extraction, accompanied with a concept exploration interface to aid users' understandings and elicit supplementary user feedback as necessary.
- We create a large-scale evaluation benchmark that includes 1713 slice discovery settings across three widely used datasets. This benchmark enables quantitative comparisons of slice discovery methods in object detection tasks, where VCR consistently outperforms baselines. Through interviews with six industry machine learning scientists and engineers, we also demonstrate VCR's usability and effectiveness for understanding the performance of real-world object detection models.

2 BACKGROUND AND RELATED WORK

2.1 Background: Object Detection

Given images, object detection models provide predictions in the form of bounding boxes along with predicted class labels for each detected object. These predicted bounding boxes are evaluated against ground-truth bounding boxes to assess class prediction accuracy as well as bounding box alignment. The quality of bounding boxes is crucial in object detection models and is assessed using the Intersection over Union (IoU) metric that measures the overlap between two bounding boxes by dividing the area of intersection by the area of their union.

Bounding box matching, the process of associating predicted and ground-truth bounding boxes, is also a critical task in object detection. Different systems implement this task differently; some prioritize maximizing IoU (*e.g.*, using the Hungarian algorithm), while others give preference to class label confidence.

In contrast to image classification, where the focus is mainly on correct classification, object detection models can make errors in multiple ways, including:

- *Classification error:* The bounding box is localized correctly (IoU greater than some threshold) but classified incorrectly.
- *Localization error:* The bounding box is classified correctly, but the IoU with ground truth is low, indicating poor localization.
- *Background error (false positive):* The model incorrectly detects the background as objects.
- *Missed ground truth (false negative):* Ground truth objects are undetected, not covered by classification or localization errors.

These error types are essential considerations in evaluating and improving object detection models [11].

2.2 Related Work

We discuss related work in slice discovery methods and visual concepts. Table 1 summarizes the main features offered by representative frameworks that help discover and explain systematic errors in image classification tasks.

Slice Finding using Tabular Attributes. Our work is inspired by the success of slice finding methods for uncovering systematic errors in tabular datasets [7, 15, 16, 41, 42, 45]. These methods primarily work on predefined columns of tabular data, identifying problematic data subgroups defined by predicates such as (age=25-40, gender=Male). For example, DivExplorer [42], SliceLine [45], and Macrobase [7] use optimized frequent itemset mining algorithms such as Apriori [5] and FP-growth [26] for slice discovery, while Slice Finder [15, 16] uses decision trees and lattice search techniques. VCR uses a frequent itemset mining-based slice discovery algorithm that is implemented efficiently to support concept absences.

Researchers have applied similar methods to image datasets, by using image metadata attributes as dimensions for evaluating vision models' performances. For example, SliceTeller [56] identifies under-performing image slices by applying frequent itemset mining techniques on predefined image metadata (*e.g.*, annotated attributes in the CelebA dataset [36]). Uni-Evaluator [13] evaluates classification, detection, and segmentation models using both discrete (*e.g.*, class labels) and continuous metadata attributes (*e.g.*, sizes, aspect ratios). VCR also exploits metadata but introduces additional slicing dimensions via visual concepts.

Slice Finding for Vision Models. A number of automated slice discovery methods seek to evaluate performance of image classification models beyond predefined metadata [17, 20, 23, 43, 49, 50, 53]. For example, Spotlight [20] identifies problematic slices of images

	VCR (Ours)	SliceTeller [56]	Domino [23]	POEM [17]	ESCAPE [6]
Model Agnostic	✓	✓	✓	✗	✓
Segment-level Concepts	✓	✗	✗	✓	✓
Leverage Metadata	✓	✓	✗	✗	✗
Automated Discovery	✓	✓	✓	✓	✗
Object Detection	✓	✗	✗	✗	✗

Table 1: Feature comparison between VCR and representative frameworks for identifying systematic errors in image classification tasks.

by searching for contiguous regions in the final layer representation space of a neural network that align with errors. Domino [23] fits an error-aware gaussian mixture model on cross-modal embeddings such as CLIP [44] to generate slices (clusters). FACTS [53] first learns a model that separates bias-aligned and bias-conflicting slices for each class, and then fits a mixture model in the bias-amplified feature space using a similar approach to Domino. POEM [17] uses a pre-trained semantic segmentation model on Unified Perceptual Parsing [52] to label visual concepts and identifies a filter activation map in image classifier CNNs that overlaps with the visual concepts to use for explanations. In contrast, VCR's visual concepts are not limited to predefined labels. EAC [50] uses SAM [33] to generate segments for each image as visual concepts and uses Shapley values to explain each concept's contribution to the model's prediction, whereas VCR's notion of visual concepts is a cluster of semantically related segments across images.

Visual Concepts. Visual concepts have been widely used to understand internal states and to explain the performance of computer vision models. Some methods require users to provide labeled examples for the concept of interest [9, 32, 58] or collaborate with the system interactively [6, 29, 48, 57]. For example, ESCAPE [6] provides a workflow that allows users to select a set of semantically coherent segments to be defined as a visual concept. Similarly, VL-Slice [48] provides a human-in-the-loop visual analytics tool that recommends similar and counterfactual visiolinguistic clusters to help users form slices for vision-language models. While VCR can also incorporate user feedback, its focus is to support automated and unsupervised concept extraction.

Others seek to extract concepts automatically [24, 28, 40]. For example, Visual Concept Programming [28] learns the embedding of each pixel using a self-supervised representation learning approach to segment images into semantically meaningful regions. NeuroCartography [40] groups neurons in deep neural networks based on their activation maps and visualizes concepts detected by such neuron groups. ACE [24] extracts concepts in an unsupervised fashion by clustering on the activation space of final layers in CNNs. VCR uses a similar idea but leverages the embedding space of input images generated by vision transformers and is therefore agnostic to the architecture of the vision model used.

3 OVERVIEW

VCR is an automated slice discovery framework that enables users to validate object detection models by identifying semantically coherent data subgroups that the model performs poorly on. VCR is agnostic to the model architecture, operates without requiring any user input, and scales to large datasets. VCR consists of three main components:

Concept Extraction. The first component takes an input image dataset, its corresponding ground-truth annotations (bounding boxes and labels), the target object detection model, and its predictions. It extracts visual concepts and identifies whether they have a significant pixel overlap with the model predictions. The concept extraction is performed in an unsupervised fashion to avoid constraints on a predefined label set. The output of this component summarizes the interaction between every bounding box and visual concept in a tabular format, such as shown in Figure 1.

Concept Pattern Mining. The second component takes as input the concept-bounding box interactions and error metrics and outputs common patterns that are associated with poor model performance via frequent pattern mining techniques [25]. After mining, users will be presented with patterns that describe how the presence and/or absence of certain concepts relates to the model's performance. For example, one may find that the presence of "pole" and the absence of "road" lead to poor localization of car bounding boxes. VCR also combines bounding box metadata such as sizes and positions along with the concept interactions in the mining.

Concept Exploration and Refinement. The third component is responsible for helping users understand, explore, and manually label the visual concepts as needed. This provides the flexibility for users to inject domain knowledge into the otherwise automated workflow. For example, users might want to look into the "road" concept that surfaced from the mining results. Upon exploration, the user finds that there are three separate road concepts and decides to merge them. The updated visual concepts can then be used to generate new results from the miner.

4 EXTRACTION AND EXPLORATION OF VISUAL CONCEPTS

In this section, we introduce VCR's default unsupervised visual concept extraction pipeline (§4.1), as well as the interactive concept exploration interface for soliciting additional user feedback (§4.2).

4.1 Unsupervised Concept Extraction

Step 1: Segment Embeddings. First, we use an image segmentation model to extract segments for each input image. We use the state-of-the-art Segment Anything Model (SAM) [33] from Meta, which can produce fine-grained segmentation for unfamiliar objects and images without additional training. We filter out segments that are less than one percent of the total image area.

Next, we incorporate semantic information into the SAM segments using MaskCLIP [21], which distills CLIP representation from a full image to a masked image. Specifically, we use MaskCLIP to generate CLIP embeddings at the pixel level and resize the shape of the MaskCLIP embeddings to match the shape of the output from SAM using bilinear interpolation. We can then derive segment-level embeddings by averaging the pixel embeddings that intersect with each segment. This process essentially assigns a single CLIP embedding to each SAM segment, which helps semantically related segments be closer in CLIP's embedding space. For instance, embedding vectors representing different types of dogs would be closer to each other than those representing different types of cats. In § 6.3, we evaluate an alternative embedding DinoV2 [39].

Step 2: Concept Clustering and Labeling. Similar to how ACE [24] forms concepts by clustering on the activation space of CNNs, we perform clustering on the segment-level embeddings generated from the previous step to derive visual concepts. As a result, segments grouped into the same cluster are considered to be representative of the same visual concept. For the choice of clustering method, we used K-means clustering after finding that other methods like DBSCAN, HDBSCAN, and agglomerative clustering are either too sensitive to outliers and hyperparameters or too computationally expensive for large datasets. Specifically, we use the efficient K-Means clustering implementation from the Faiss library [22]. In § 6.3, we show that the VCR's performance is not sensitive to the number of clusters.

We also leverage the cross-modality nature of the CLIP embedding space to automatically assign labels to concept clusters. We achieve this by pairing the cluster center with the text label having the smallest CLIP embedding distance. The input labels utilized for this pairing process can be sourced independently from the input images. For instance, one approach is to compile a list of frequently used non-abstract English nouns. Alternatively, users can specify labels in an open vocabulary manner. For example, practitioners working with images taken by autonomous vehicles may provide labels such as "car window," "pedestrian," and "green light."

Since different concepts may be mapped to the same label, we resolve these conflicts by appending numbers to conflicting labels, which can be thought of as subcategories of the same concept. For example, we may see labels like "tree_1", "tree_2" and "tree_3," suggesting that they were either different enough to separate into different categories or that the clustering granularity is too fine.

Step 3: Concept Interaction. Finally, we identify the set of visual concepts that interact with the model's prediction. We use a simple greedy bounding box matching algorithm, as we found different pairing strategies did not noticeably change results. After the matching process, we identify all concepts that have a non-trivial area overlap with each pair of predicted and ground truth bounding boxes. Since bounding boxes are tightly fitted around the object, we additionally apply a fixed amount of pixel padding (*e.g.*, 50) around the bounding boxes to help capture additional concept interactions that are in the immediate vicinity of the object. Having identified the concept interactions, we can generate the following concept-bounding box interaction table I, whose rows correspond to different predicted and ground truth bounding box pairs and whose columns correspond to different visual concepts. $I[i][j]$ represents the number of segments belonging to the j^{th} concept that interacts with the i^{th} bounding box pair.

4.2 Concept Exploration and Refinement

In addition to the unsupervised extraction, VCR provides a concept exploration interface to help users better understand extracted concepts and make adjustments as needed, shown in Figure 2.

The top panel shows an overview of all extracted visual concepts projected into a 2D space using UMAP [37], where each color represents a different concept. When a user hovers over a dot in this 2D view, the automatically generated concept label will appear. This 2D overview allows users to explore the concept space. For example, the bottom right shows a collection of concepts relating to nature elements such as flower, tree and sky.

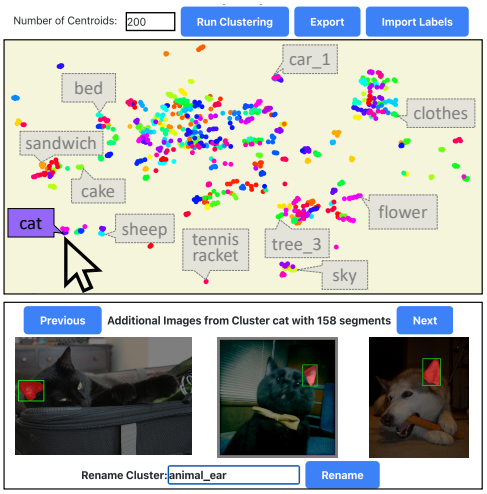


Figure 2: VCR's concept exploration interface allows users to explore and understand extracted concepts and also perform additional refinement, such as relabeling a concept and merging clusters.

By clicking on a specific concept cluster, users can see a sample of image segments belonging to the concept in the bottom panel. For example, Figure 2 shows example segments from the concept that were automatically labeled as "cat". Upon inspection, the user realizes that this concept cluster is more about "animal_ear" than cats. The user can then rename this cluster on the interface. Using this rename feature, users can also merge different concept clusters by renaming them to the same label. We do not support splitting clusters, but users have the option to tune the total number of clusters to improve the granularity. At the end of the exploration, users can export their customized concepts to an output file, which can then be consumed by the miner.

5 CONCEPT PATTERN MINING

Given the error metrics and the visual concept interactions and metadata in tabular format, VCR identifies patterns that correlate with model's performance via frequent itemset mining techniques.

5.1 Supported Itemsets

VCR considers classification error, localization error, background error (false positive), and missed ground truth (false negative) as discussed in § 2.1. Given an error type and an IoU threshold which is default at 0.5 (as in [11, 27]), VCR marks a subset of the bounding box pairs as problematic according to the error metrics.

VCR outputs the common patterns among problematic predictions in the form of frequent itemsets. Users can specify a minimum slice size (minimum support) parameter. VCR identifies all data slices above the support threshold and ranks them according to the average error metrics of the slice. We support itemsets that are a conjunction of predicates with the following types of attributes:

- *Bounding box stats:* relative size of the bounding box as a percentage of the image, aspect ratio of the bounding box, and relative position (*e.g.*, left, right, top, bottom) of the predicted bounding box with respect to the ground truth.
- *Crowding:* the number of nearby bounding boxes that overlap with the predicted and ground truth bounding box pair.

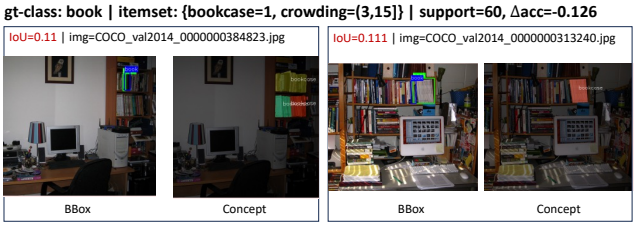


Figure 3: Example output from VCR's concept pattern mining. Concepts are highlighted in bright colors. Green bounding box indicates ground truth and blue bounding box indicates prediction.

- *Image metadata*: additional image annotations such as class labels, time of day, and location.
- *Counts of visual concepts:* the count of each type of visual concept that overlaps with the predicted and ground truth bounding box pair. This can also be used as a binary attribute that represents the presence and the absence of a concept.

Numeric attributes such as the bounding box area are automatically discretized into 10 bins using quartiles.

Figure 3 shows an example itemset $\{bookcase = 1, crowding = (3, 15]\}$ generated by VCR, with a support count of 60 and an accuracy difference of -0.126. This means that this data slice contains 60 total bounding box predictions, and the average IoU for these predictions is 0.126 lower than the average of the entire dataset. The itemset further summarizes the commonality among these bounding boxes: they all interact with the "bookcase" concept and overlap with between 3 to 15 nearby bounding boxes.

5.2 Improving Mining Scalabilty and Utility

Concept Absence and Pruning. Classic frequent itemset mining focuses on identifying items that occur frequently. Accounting for the absence of items is equally important in our application. For example, a model reliant on the presence of car wheels in order to predict a cars may fail when wheels are absent.

The challenge in efficiently mining concept absence arises from the sparsity of concepts: for a dataset with many concepts, most concepts do not interact with a given predicted and ground truth bounding box pair. Mining under these circumstances leads to a combinatorial explosion of results dominated by concept absences. Therefore, naïvely supporting concept absence does not scale; for example, POEM experienced significant slowdown in mining beyond 15 concepts [17].

To address this challenge, we introduce a pruning optimization that significantly improves the mining performance for concept absences. Our observation is that for any pair of concepts c_i and c_j, the absence of c_i is only meaningful if there exist cases in which c_i and c_j are both present. For example, the "sea" and the "boat" concepts frequently co-occur, so scenarios when the sea is present but the boat is absent could be worth investigating; in contrast, "carrots" seldom appear with "sea", so itemsets $\{carrot = 0, sea = 1\}$ and $\{sea = 1\}$ might contain almost identical sets of bounding boxes. We leverage this insight to apply a pruning rule when generating itemset candidates of length 2. Note that since all frequent itemsets of length 2 contain at most one absence according to this pruning rule, this implies that all itemsets generated of arbitrary length will contain at most one absence according to the apriori principle.

Result Duplication. We allow users to optionally de-duplicate the results generated from the mining. De-duplication is useful since mining can produce many itemsets that represent almost the same data and confuse the user–an issue that concept absences further complicates. For example, $\{sky = 1, sea = 0\}$ might represent a similar set of bounding-boxes as $\{sky = 1, land = 0\}$.

In tabular data slicing settings, DivExplorer [42] uses Shapley value [46] to quantify the contribution of single-attribute patterns to the itemset as a basis for deduplication, but we found that it adds non-trivial overheads to the mining. We instead use a lightweight, greedy deduplication algorithm that aims to preserve interesting itemsets while maximizing the diversity. Our method involves iterating through the itemsets in order of decreasing divergence in error metrics. We mark the bounding box pairs covered by each itemset. We only include a new itemset in the results when it contains a significant portion (e.g., $\delta = 50\%$) of bounding box pairs not covered by previous itemsets.

6 EVALUATION

In this section, we evaluate the empirical performance of VCR. Our experiments show that:

- VCR's automated slice discovery method consistently outperforms baselines in identifying problematic subgroups across different datasets and settings in object detection tasks. Both visual concepts and metadata attributes contribute meaningfully to VCR's final performance (§6.2).
- VCR can support hundreds of concepts at interactive time, and is robust to variations of main parameters. (§6.3).
- VCR is easy to use and can help ML practitioners understand the performance of object detection models (§6.4).

6.1 Evaluation Methodology

Few real-world object detection datasets specify data slices where a model systematically underperforms. Indeed, knowing the true problematic slices apriori for a given dataset and model is difficult unless we can artificially control the ground truth. Following existing practices [23, 30, 43], we programmatically generate 1713 slice discovery settings to enable quantitative performance comparison. Table 2 summaries our evaluation benchmark.

Slice discovery settings. We use three widely-used datasets in object detection tasks: COCO 2014 [35], Visual Genome [34], and BDD 100K [54]. We consider two types of reasons that cause the model to underperform: metadata-based and content-based. Slices derived from metadata can exhibit better visual consistency, while those derived from contents will have better semantic coherency. Accordingly, we create four error scenarios based on:

- *Color*: Ground truth bounding boxes are resized into 50x50 squares, followed by K-Means clustering on raw pixel values to assign them into different color clusters. We use 20 color clusters for each class of objects, and each cluster forms a slice.
- *Size and Aspect Ratio*: Bounding boxes are sorted into bins based on width*height for size and width/height for aspect ratio. We use a randomly chosen number of bins between 5 and 15 for each object class, and each bin forms a slice.
- *Semantic*: Inspired by [30], we extract semantically coherent slices such as "[object class] next to [setting/object]" (*e.g.*, "person next to water"). For COCO and Visual Genome, we use

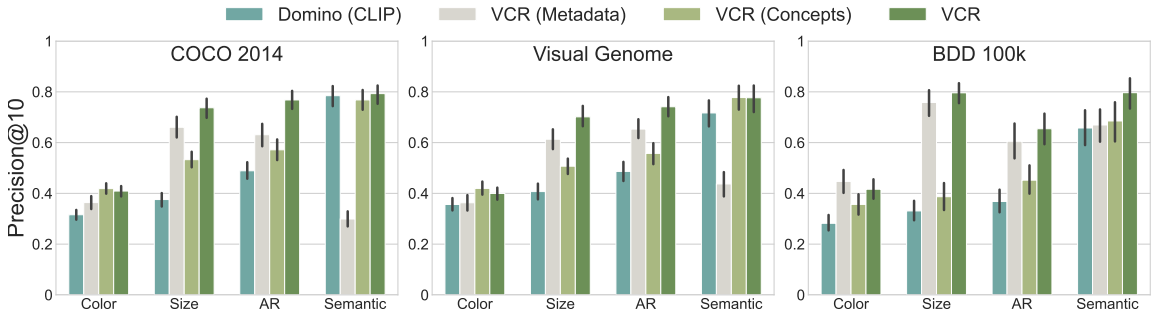


Figure 4: VCR leads to consistent improvement in slice discovery performance compared to baselines across three datasets and four error scenarios, with a total of 1713 total slice discovery settings.

their image captions, embed them with CLIP embeddings, and retrieve up to 500 images closest in the embedding space to our target prompts. For BDD, we use the provided annotations to define slices based on weather condition, time of day, and scene (*e.g.*, highway, residential); VCR does not directly use these metadata as slicing dimensions.

We discard slices with fewer than 25 samples. Once a slice is generated, we synthetically increase its localization error rate by perturbing the predicted bounding box location to lower the IoU. We model the IoU errors as a Gaussian distribution centered on 0.4, just below the standard 0.5 IoU error threshold. This allows us to create ground truth problematic slices for evaluation. While we focus on localization error, other error types can be supported similarly.

Our default object detector is Faster-RCNN from MMDetection [2, 14] trained on the MS-COCO Dataset's 2017 train split. We provide additional details of the slice discovery setting generation, as well as samples of ground truth slices generated in the Appendix A of the technical report [4].

Baselines. We evaluate VCR against three baselines:

- Domino [23]: Similar to VCR, Domino also uses an external, pretrained model to generate image embeddings used for explanations. We use 100 clusters and $\gamma = 40$ after hyperparameter tuning. We also provide Domino IoU values as its error metric.
- VCR (concept): VCR using concept interactions as the only slicing dimension (i.e. without metadata). This baseline is similar to POEM [17], which also only uses visual concepts as explanation primitives.
- VCR (metadata): VCR with only metadata attributes in the itemsets. This baseline is similar to SliceTeller [56], which slices datasets based on predefined metadata attributes.
- VCR: By default, we use 500 concepts, a support count threshold of 10, limit the maximum itemset length to 3, and the greedy de-duplication algorithm with overlap threshold $\delta = 50\%$. For models, VCR uses SAM's ViT-L for segments and MaskCLIP Vit-B/16 for pixel-level embeddings.

Since previous works do not support object detection tasks, we simplify the tasks by cropping images from ground-truth bounding-box annotations and treating them as image classification tasks for baselines. We apply additional pixel padding to enhance the quality of image crops, which provides important contextual information and improves the performance of baselines.

Datasets	Color	Aspect Ratio	Size	Semantic	Total Settings
COCO	287	156	153	184	780
VG	217	143	140	113	613
BDD	100	65	75	80	320

Table 2: Overview of our evaluation benchmark.

Metric. We adopt Domino's *Precision@k* metric for evaluation, which is the proportion of the topk elements in the discovered slice that are also present in the ground truth slice. We use $k = 10$ by default. Domino uses a Gaussian mixture model to estimate the probability of an element belonging to each slice; the top k elements within each slice are those with the highest membership probability. VCR ranks elements within each slice based on their error metrics. Both Domino and VCR rank slices by error rate, and we select the top 10 slices from each system. The Precision@k metric is calculated as the maximum precision value among the top slices.

6.2 Quantitative Comparison

Figure 4 summarizes the performance of VCR and baselines across all slice discovery settings. Overall, VCR consistently outperforms Domino, and both visual concepts and metadata contribute meaningfully to VCR's final performance. For the three metadata tests, VCR's mean difference with Domino in precision score is 0.244 (60.9% increase), 0.206 (48.2% increase) and 0.292 (87.8% increase) in the COCO, Visual Genome (VG), and BDD datasets, respectively. For the semantic tests, VCR's mean difference with Domino in precision score is .008, 0.100 (14.0% increase) and 0.170 (25.9% increase) in the three datasets respectively.

Domino sees its best performance on the semantic slice tests and is not far behind VCR. This is expected as Domino uses CLIP embeddings of images to form semantically coherent clusters. Understandably, VCR (metadata) can not differentiate semantic attributes and experiences decreased precision in most semantic settings.

However, in BDD's semantic test cases, we find VCR (metadata) can perform relatively well. While semantic tests in COCO and VG were derived from classes' relations to other objects or settings (*e.g.*, indoors), BDD's semantic tests featured both concrete dimensions with "scene" as well two abstract ones, "timeofday" and "weather." We find that VCR (metadata) achieves higher precision that Domino in "timeofday" and "weather", only losing in the scenes test. When combined with VCR (concepts), metadata enables a significant increase in precision. This highlights the importance of metadata attributes even in the semantic setting.

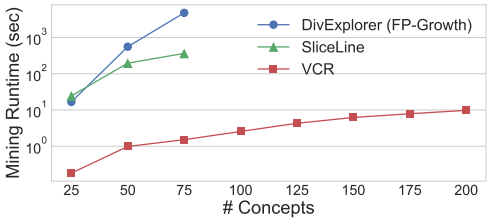


Figure 5: VCR's absence pruning optimization enables significant speedups compared to alternative mining approaches.

Configurations	Color	Aspect Ratio	Size	Semantic
k=500	0.409	0.737	0.767	0.792
k=400	0.408	0.728	0.746	0.762
k=300	0.409	0.722	0.732	0.749
k=200	0.405	0.750	0.764	0.751
δ = 50%	0.409	0.737	0.767	0.792
δ = 25%	0.411	0.740	0.768	0.790
δ = 0% (No Dedup)	0.324	0.664	0.674	0.687
MaskCLIP	0.409	0.737	0.767	0.792
DinoV2	0.412	0.748	0.767	0.762

Table 3: Effect of number of clusters (k), embedding choice, and deduplication threshold (δ) on VCR's precision on the COCO dataset. The rows in bold represent the default experiment configuration.

For metadata slices on bounding box size and aspect ratio, VCR sees the biggest improvements over Domino, outperforming by 0.255-0.465 precision points in the three datasets. This is because VCR utilizes bounding box statistics as slicing dimensions, while Domino's semantic-based slicing is a poor fit for these object detection specific error scenarios. For color clusters, all methods exhibit relatively poor performance, since neither metadata nor visual concepts explicitly capture the concept of color. Color clusters also tend to be more noisy compared to other test cases, particularly as we directly utilized raw pixel values. However, for specific classes like traffic lights and umbrellas, coherent color clusters (examples in Appendix A [4]) could be formed. VCR still outperforms Domino by 0.04 to 0.13 precision points across datasets.

6.3 Detailed Analysis

Concept Mining Scalability. We compare VCR against two representative tabular slice discovery frameworks, DivExplorer [42] and SliceLine [45], using their open-source Python implementations [1, 3]. DivExplorer supports Apriori and FP-growth, while SliceLine uses a linear-algebra-based method. We use a binary concept presence/absence data extracted from the COCO dataset with 100,000 rows and 200 columns (concepts). All methods use a support threshold of 0.03 and generate itemsets with presence/absence up to length 3. Figure 5 reports the mining runtime versus concept count in log scale, averaged over 5 runs. DivExplorer with FP-Growth takes over an hour at 75 concepts, and SliceLine triggers the out-of-memory killer at 100 concepts. In contrast, VCR finishes within 10 seconds even at 200 concepts, indicating at least two orders of magnitude speedup, mainly due to the absence pruning optimization.

Sensitivity Analysis. Table 3 presents VCR's performance across varying deduplication thresholds (δ) and concept cluster counts (k). While deduplication enhances VCR's performance, VCR is not sensitive to the overlap threshold δ. VCR is also robust to changes in the number of clusters. Moreover, users can dynamically adjust concept counts and labels through our concept explorer interface.

Furthermore, we evaluate an alternative segment embedding approach derived from DinoV2 [39]. DinoV2 is a pre-trained vision transformer model that provides embedding at the patch-level, where fixed-sized regions of pixels are assigned the same embedding values. To obtain DinoV2's segment-level embeddings, we follow a similar procedure as before and extract the last layer's patch embeddings, resize them using bilinear interpolation to create pixel-level embeddings, and finally average the pixel embeddings for each segment's overlap with the pixels. Overall, MaskCLIP has a slight performance edge over DinoV2. In addition, MaskCLIP's cross-modal embedding capability allows for automated generation of concept labels, further enhancing its utility.

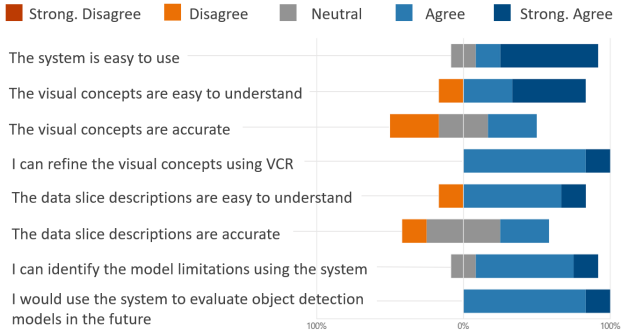


Figure 6: Expert perception of VCR, using a 5-point Likert Scale questionnaire.

6.4 Domain Expert Feedback

We also conducted interviews with six expert users, where they were asked to use VCR to evaluate an object detection model trained on the COCO dataset and provide feedback.

Expert User Demographics. We interviewed six expert users (industry ML scientists and engineers). All experts have two or more years of experience with Machine Learning (5.67 ± 2.49 years) and are familiar with object detection (four have trained detection models before, and two have used but not trained these models). Five of the experts have a PhD in STEM, and one has a Masters Degree. The experts are not authors of this paper.

Interview Protocol. Each interview lasted for 45 minutes and proceeded as follows: first, the user filled out a demographics questionnaire (5 min). Next, we demoed the system capabilities using the object "car" , allowing the experts to identify the model problems in the data slices, and answered any questions they had (10 min). We then asked the experts to use VCR to identify and understand the model limitations, *i.e.*, problematic data slices, in other objects (20 min). Finally, we asked them for feedback, including positive aspects, negative aspects and points of improvement (10 min). We also used a 5-point Likert scale questionnaire to assess the perception of the system's functionality and usability.

Expert user analysis and insights. The experts evaluated two to three objects per session. A shared strategy involved initially examining the worst-performing slices. This allowed them to pinpoint where the models struggled with object detection. Next, they

inspected the slices where the models outperformed the average, indicating scenarios where the model could easily detect the objects. Here, we list some of their findings:

1. Occlusion was frequently listed as a fundamental reason for subpar detection performance. This issue was identified in the objects "car" and "chair". During the system demo using the "car" object, experts immediately found that the poorest performing data slice was defined by the itemset $\{pole = 1, road = 0\}$ ($\Delta_{acc} = -0.218$), indicating the presence of a pole near the undetected object in the image. Upon further examination, it was observed that the pole was obstructing the car in the picture. Similarly, when the experts were inspecting the "chair" object, they found that undetected chairs were frequently due to occlusion by babies or toddlers, as indicated in the slice $\{baby = 1, diningtable = 0\}$ ($\Delta_{acc} = -0.248$).

2. Crowding was also a frequent error identified by the experts. For example, when exploring the data slices belonging to "book" object, they noticed that the slice $\{bookshelf = 1\}$ ($\Delta_{acc} = -0.141$) had poor detection performance. Upon inspecting the images and detections, they noticed that the model had trouble identifying an object among objects of the same type. Similarly, when exploring the "boat" object data slices, they found that most of the mistakes in the boat class arrived from object crowding, *e.g.*, multiple boats next to each other.

3. Ground truth errors were also identified. The first type of error found in this category are related to crowding errors: experts found that slices containing multiple boats often had inconsistent annotations: sometimes, the bounding box would contain a single boat. Other times, multiple boats would be included in the box. This issue is also present in other objects, such as "broccoli" (multiple pieces in a plate) and "book" (multiple books in a bookshelf). A second type of error occurred when only part of an object is included in the bounding box. For example, in the chair detection, one slice corresponded to chairs where only part of the object was included in the box.

4. Easily detected objects. Experts also explored the data slices with the best performance, gaining additional insight into the model. For example, when exploring the "chair" object, experts found that the model performs best when the chair is large or looks like a sofa ($\{floor = 0, sofa = 1\}$, $\Delta_{acc} = 0.195$). Another interesting case comes from the "boat" object. When experts investigated the best performing slice ($\{water = 1, boat = 1\}$, $\Delta_{acc} = 0.339$), they noticed that kayak, a particular type of boat, was easy to detect.

5. Spurious correlations arise due to the slice finding method's reliance on co-occurrence to identify problematic slices. This can sometimes lead to the identification of slices that are the result of coincidental correlations rather than actual problems. A common example of this issue was found when exploring the "boat" class. The slice that had the poorest performance was characterized by the $\{clouds = 1\}$ ($\Delta_{acc} = -0.518$) itemset. However, the experts found the association of missed detection with clouds to be incorrect. Upon further investigation, they discovered that the errors were not due to the presence of clouds. Instead, the errors were the result of overcrowding issues (multiple boats in close proximity to each other) and labeling problems (boxes containing more than one boat). Similar spurious correlations happened with the slices containing sky ($\{sky = 1\}$). Despite the slice itemset indicating a false correlation between clouds and incorrect detections, the grouping of these similar errors together still allowed users to identify the mistake with relative ease.

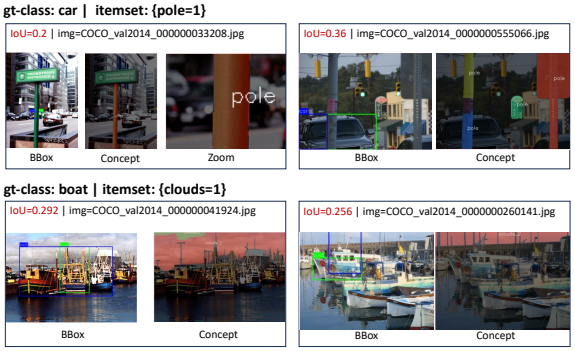


Figure 7: Example data slices investigated by expert users. Concepts are highlighted in bright colors. Green bounding box indicates ground truth and blue bounding box indicates prediction. Top: data slice identified for the object "car", where poles obstruct the detection of the object. Bottom: data slice identified for the object "boat", where crowding hampers the detection of the object.

Expert feedback. The expert users generally expressed positive feedback about the system. They valued how the data slices could provide potential reasons for a model's errors, aiding them in considering various contributing factors. The simplicity of the interface was also well-received, along with the four views used to display of results (original images, detection boxes, semantic segmentation, and a zoomed-in segmentation). Users also appreciated how segmentations facilitated their understanding of the model's mistakes. However, some experts noted occasional inaccuracies in the segmentation labels. While they valued the ability to alter the labels of visual concepts, they also expressed a desire to refine segments in real-time, such as splitting a cluster of segments containing multiple objects. Additionally, they found data slices with absent concepts occasionally difficult to comprehend, for example, the detection of the object "chair" performing poorly when no wall was present. At the end of the interview, the expert users were asked to fill out a Likert scale questionnaire about their experience with the system. Figure 6 shows the user's responses. Overall, the responses to this questionnaire coincide with the other feedback provided.

7 CONCLUSION

As practitioners increasingly seek insights beyond a model's overall test performance, there is a growing need for better tools to identify systematic errors, where models perform significantly worse on a semantically meaningful subset of the data. In response, we introduce VCR, the first automated slice discovery framework for object detection tasks. VCR is model-agnostic and scalable, thanks to the novel combination of unsupervised visual concept discovery and tabular data slicing techniques. Through our large-scale evaluation benchmark with 1713 slice discovery settings, we show that VCR consistently outperforms alternatives in identifying problematic data slices. Through in-depth interviews with six expert users, we demonstrate the overall effectiveness of VCR for identifying and explaining errors in real-world object detection models.

REFERENCES

[1] [n.d.]. DivExplorer Github. https://github.com/divexplorer/divexplorer. Accessed Februrary, 2024.

[2] [n.d.]. MMDetection Github. https://github.com/open-mmlab/mmdetection. Accessed Februrary, 2024.

[3] [n.d.]. SliceLine Github. https://github.com/DataDome/sliceline. Accessed Februrary, 2024.

[4] [n.d.]. VCR: A Tabular Data Slicing Approach to Understanding Object Detection Model Performance (Technical Report). https://github.com/d2i-lab/VCR/blob/master/docs/tr.pdf. Accessed March, 2024.

[5] Rakesh Agrawal, Ramakrishnan Srikant, et al. 1994. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, Vol. 1215. Santiago, Chile, 487–499.

[6] Yongsu Ahn, Yu-Ru Lin, Panpan Xu, and Zeng Dai. 2023. ESCAPE: Countering Systematic Errors from Machine's Blind Spots via Interactive Visual Analysis. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 834, 16 pages. https://doi.org/10.1145/3544548.3581373

[7] Peter Bailis, Edward Gan, Samuel Madden, Deepak Narayanan, Kexin Rong, and Sahaana Suri. 2017. Macrobase: Prioritizing attention in fast data. In *Proceedings of the 2017 ACM International Conference on Management of Data*. 541–556.

[8] Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. 2019. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. *Advances in neural information processing systems* 32 (2019).

[9] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. 2017. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 6541–6549.

[10] Alceu Bissoto, Eduardo Valle, and Sandra Avila. 2020. Debiasing skin lesion datasets and models? not so fast. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 740–741.

[11] Daniel Bolya, Sean Foley, James Hays, and Judy Hoffman. 2020. Tide: A general toolbox for identifying object detection errors. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer, 558–573.

[12] Joy Buolamwini and Timnit Gebru. 2018. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*. PMLR, 77–91.

[13] Changjian Chen, Yukai Guo, Fengyuan Tian, Shilong Liu, Weikai Yang, Zhaowei Wang, Jing Wu, Hang Su, Hanspeter Pfister, and Shixia Liu. 2024. A Unified Interactive Model Evaluation for Classification, Object Detection, and Instance Segmentation in Computer Vision. *IEEE Transactions on Visualization and Computer Graphics* 30, 1 (2024), 76–86. https://doi.org/10.1109/TVCG.2023.3326588

[14] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. 2019. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155* (2019).

[15] Yeounoh Chung, Tim Kraska, Neoklis Polyzotis, Ki Hyun Tae, and Steven Euijong Whang. 2019. Automated data slicing for model validation: A big data-ai integration approach. *IEEE Transactions on Knowledge and Data Engineering* 32, 12 (2019), 2284–2296.

[16] Yeounoh Chung, Tim Kraska, Neoklis Polyzotis, Ki Hyun Tae, and Steven Euijong Whang. 2019. Slice finder: Automated data slicing for model validation. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 1550–1553.

[17] Vargha Dadvar, Lukasz Golab, and Divesh Srivastava. 2023. POEM: Pattern-Oriented Explanations of Convolutional Neural Networks. *Proceedings of the VLDB Endowment* 16, 11 (2023), 3192–3200.

[18] Terrance De Vries, Ishan Misra, Changhan Wang, and Laurens Van der Maaten. 2019. Does object recognition work for everyone?. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 52–59.

[19] Alex J DeGrave, Joseph D Janizek, and Su-In Lee. 2021. AI for radiographic COVID-19 detection selects shortcuts over signal. *Nature Machine Intelligence* 3, 7 (2021), 610–619.

[20] Greg d'Eon, Jason d'Eon, James R Wright, and Kevin Leyton-Brown. 2022. The spotlight: A general method for discovering systematic errors in deep learning models. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*. 1962–1981.

[21] Xiaoyi Dong, Jianmin Bao, Yinglin Zheng, Ting Zhang, Dongdong Chen, Hao Yang, Ming Zeng, Weiming Zhang, Lu Yuan, Dong Chen, et al. 2023. Maskclip: Masked self-distillation advances contrastive language-image pretraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10995–11005.

[22] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library. *arXiv preprint arXiv:2401.08281* (2024).

[23] Sabri Eyuboglu, Maya Varma, Khaled Saab, Jean-Benoit Delbrouck, Christopher Lee-Messer, Jared Dunnmon, James Zou, and Christopher Ré. 2022. Domino: Discovering systematic errors with cross-modal embeddings. *arXiv preprint arXiv:2203.14960* (2022).

[24] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. 2019. Towards automatic concept-based explanations. *Advances in neural information processing systems* 32 (2019).

[25] Jiawei Han, Hong Cheng, Dong Xin, and Xifeng Yan. 2007. Frequent pattern mining: current status and future directions. *Data mining and knowledge discovery* 15, 1 (2007), 55–86.

[26] Jiawei Han, Jian Pei, and Yiwen Yin. 2000. Mining frequent patterns without candidate generation. *ACM sigmod record* 29, 2 (2000), 1–12.

[27] Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. 2012. Diagnosing error in object detectors. In *European conference on computer vision*. Springer, 340–353.

[28] M. Hoque, W. He, A. Shekar, L. Gou, and L. Ren. 2023. Visual Concept Programming: A Visual Analytics Approach to Injecting Human Intelligence at Scale. *IEEE Transactions on Visualization and Computer Graphics* 29, 01 (jan 2023), 74–83. https://doi.org/10.1109/TVCG.2022.3209466

[29] Jinbin Huang, Aditi Mishra, Bum Chul Kwon, and Chris Bryan. 2022. ConceptExplainer: Interactive explanation for deep neural networks from a concept perspective. *IEEE Transactions on Visualization and Computer Graphics* 29, 1 (2022), 831–841.

[30] Siddharth Joshi, Yu Yang, Yihao Xue, Wenhan Yang, and Baharan Mirzasoleiman. 2023. Towards Mitigating Spurious Correlations in the Wild: A Benchmark & a more Realistic Dataset. *arXiv preprint arXiv:2306.11957* (2023).

[31] Kimmo Karkkainen and Jungseock Joo. 2021. Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 1548–1558.

[32] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. 2018. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*. PMLR, 2668–2677.

[33] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. *arXiv preprint arXiv:2304.02643* (2023).

[34] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision* 123 (2017), 32–73.

[35] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 740–755.

[36] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*. 3730–3738.

[37] Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).

[38] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A survey on bias and fairness in machine learning. *ACM computing surveys (CSUR)* 54, 6 (2021), 1–35.

[39] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. 2023. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193* (2023).

[40] Haekyu Park, Nilaksh Das, Rahul Duggal, Austin P Wright, Omar Shaikh, Fred Hohman, and Duen Horng Polo Chau. 2021. Neurocartography: Scalable automatic visual summarization of concepts in deep neural networks. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 813–823.

[41] Eliana Pastor, Elena Baralis, Luca de Alfaro, et al. 2023. A Hierarchical Approach to Anomalous Subgroup Discovery. In *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, California, USA, April 3–7, 2023*. IEEE.

[42] Eliana Pastor, Luca De Alfaro, and Elena Baralis. 2021. Looking for trouble: Analyzing classifier behavior via pattern divergence. In *Proceedings of the 2021 International Conference on Management of Data*. 1400–1412.

[43] Gregory Plumb, Nari Johnson, Angel Cabrera, and Ameet Talwalkar. 2023. Towards a More Rigorous Science of Blindspot Discovery in Image Classification Models. *Transactions on Machine Learning Research* (2023).

[44] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark,

et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.
[45] Svetlana Sagadeeva and Matthias Boehm. 2021. Sliceline: Fast, linear-algebra-based slice finding for ml model debugging. In *Proceedings of the 2021 International Conference on Management of Data*. 2290–2299.
[46] Lloyd S Shapley et al. 1953. A value for n-person games. (1953).
[47] Rakshith Shetty, Bernt Schiele, and Mario Fritz. 2019. Not Using the Car to See the Sidewalk–Quantifying and Controlling the Effects of Context in Classification and Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8218–8226.
[48] Eric Slyman, Minsuk Kahng, and Stefan Lee. 2023. VLSlice: Interactive Vision-and-Language Slice Discovery. In *International Conference on Computer Vision (ICCV)*. https://arxiv.org/pdf/2309.06703.pdf
[49] Nimit Sohoni, Jared Dunnmon, Geoffrey Angus, Albert Gu, and Christopher Ré. 2020. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *Advances in Neural Information Processing Systems* 33 (2020), 19339–19352.
[50] Ao Sun, Pingchuan Ma, Yuanyuan Yuan, and Shuai Wang. 2023. Explain Any Concept: Segment Anything Meets Concept-Based Explanation. *arXiv preprint arXiv:2305.10289* (2023).
[51] Kai Xiao, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. 2020. Noise or signal: The role of image backgrounds in object recognition. *arXiv preprint arXiv:2006.09994* (2020).
[52] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. 2018. Unified perceptual parsing for scene understanding. In *Proceedings of the European conference on computer vision (ECCV)*. 418–434.
[53] Sriram Yenamandra, Pratik Ramesh, Viraj Prabhu, and Judy Hoffman. 2023. FACTS: First Amplify Correlations and Then Slice to Discover Bias. In *IEEE/CVF International Conference in Computer Vision (ICCV)*.
[54] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. 2020. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2636–2645.
[55] Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. *arXiv preprint arXiv:1808.05326* (2018).
[56] Xiaoyu Zhang, Jorge Piazentin Ono, Huan Song, Liang Gou, Kwan-Liu Ma, and Liu Ren. 2022. SliceTeller: A data slice-driven approach for machine learning model validation. *IEEE Transactions on Visualization and Computer Graphics* 29, 1 (2022), 842–852.
[57] Zhenge Zhao, Panpan Xu, Carlos Scheidegger, and Liu Ren. 2021. Human-in-the-loop extraction of interpretable concepts in deep learning models. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 780–790.
[58] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. 2018. Interpretable basis decomposition for visual explanation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 119–134.

A DETAILED DESCRIPTION OF DATASETS AND SLICE SETTINGS

COCO 2014. The COCO 2014 Validation dataset is a subset of the larger Microsoft Common Objects in Context (COCO) dataset, covering a wide range of objects and scenes for a total of 40504 images. Furthermore, each image is annotated with five descriptive captions, which we leverage in the semantic slice generation process. We chose the 2014 split over the 2017 split since our detection model was directly trained on the 2017 split.
For generating semantic slices, we focus on the top 15 object classes in the dataset, ranked by the frequency of their annotations. Specifically, among these top classes, we generate semantic slices by:

- **Filtering Images:** For each class, we filter down to images containing at least one ground truth instance of that object class.
- **Generating Image Representations:** We represent each image by embedding its captions in the CLIP embedding space. For a single image, we average the embeddings for each caption and normalize the resultant vector, denoted as v_i.
- **Generating Contexts:** We use the template "[object class] next to [setting]" to create categories for the slices. We extract settings by considering the most frequent words in the captions of the object-filtered dataset, removing stop words (e.g., "the," "a," "of") and choosing a selection of the top nouns as settings.
- **Generating Context Representations:** We format each context string into template strings (e.g., "a photo of a big ", "a photo of a small ") and generate embeddings using CLIP. By using these templates, we can generate a variety of contexts that describe the object class in different ways, such as its size, quality, or appearance. We average the CLIP embeddings and normalize the resultant vector, denoted as c_j, representing one context.
- **Finding Closest Semantic Slice:** To categorize the image representations, we find the context vector, c_j, with the smallest distance to the image vector, v_i, indicating the closest semantic slice. Once the closest semantic slice is determined, every instance of the object class in that image is assigned to that category.

Figure 8 shows a few examples of semantic ground truth slices generated via this approach. For the three metadata-based settings, we again use the 15 classes with the most number of labels. For both size and aspect ratio bounding box metadata settings, we bucket bounding boxes into 5-15 different bins according to their size/aspect ratio. For color clusters, we extract 50x50 square crops of ground-truth bounding boxes and run them through the K-Means clustering algorithm for k=20 total clusters. Figure 9 shows examples of color slices generated for this dataset.

Visual Genome. The Visual Genome (VG) dataset is a large-scale image dataset covering a wide range of everyday scenes and objects, with more than 108,000 images, each annotated with dense object annotations, attributes, and relationship graphs. Instead of using all 108,000 images, we start with the first half of the dataset VG_100K (where the second half is VG_100K_2) and filter it down the images further to include only images that have any of the 80 COCO labels. Additionally, VG is known to have image overlaps with COCO. As such, we ignore any of the images that have an associated COCO "id" with it.
In order to generate semantic slices we perform the following steps:

- **Filtering Images:** For each class, we filter down to images containing at least one ground truth instance of that object class.
- **Image Representation:** We generate a set of captions for each image by selecting a diverse set of regions within the image. This selection includes:
  - A subset of the largest regions by bounding box area, representing the most prominent objects.
  - Middle-sized regions, providing contextual information and additional details.
  - A random sample of the smallest regions, offering diversity and capturing less prominent elements.

  We concatenate the phrases from these regions in groups of three to generate a set of captions for the image. This helps to form captions that encapsulate the image's coarse and fine

details. We embed these captions in the CLIP embedding space and take the average of the embeddings to represent the image. This average serves as the image representation v_i.

- **Generating Contexts:** We use the text template "[object class] next to [settings]" to generate contexts for the slice categories. We find the settings by selecting from the most common nouns found in the phrases associated with each region in an image after filtering out stop words.
- **Context Representation:** We format each context into template strings (e.g., "a photo of ", "a photo of a small ") to generate a variety of contexts that describe the object class in different ways, such as its size, quality, or appearance. We generate embeddings using CLIP for each context template, average the CLIP embeddings, and normalize the resultant vector, denoted as c_j, representing one context.
- **Finding Closest Semantic Slice:** To categorize the image representations, we find the context vector c_j with the smallest distance to the image vector v_i, indicating the closest semantic slice. This is done by calculating the L2 distance $||v_i - c_j||^2$ and selecting the context vector c_j that minimizes this distance. Once the closest semantic slice is determined, every instance of the object class in that image is assigned to that category.

We display a few of these semantic slices in Figure 10. For the three metadata based settings, we do the same as with the COCO dataset, bucketing the different sizes and aspect ratios for the bounding boxes, and using K-Means to form color clusters.

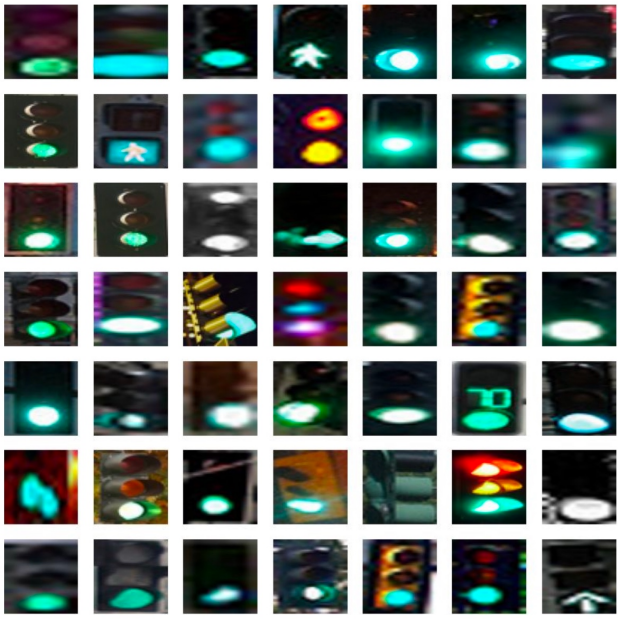
BDD100K. The BDD100K dataset consists of 100,000 images taken from the perspective of a car, featuring diverse scenes across various times and conditions. Unlike COCO and VGG, BDD100K has a very limited number of classes, most of which overlap with COCO's. To make sure BDD100K's detection annotations align with those of our object detector, we remove the "traffic sign" label and merge "rider" and "pedestrian" into "person." We further created our own 10K split of BDD100K after filtering for images that contain object detections after finding that the provided BDD10K did not always have detection labels.

From this subset of BDD data, we then created semantic slices based directly on metadata provided by BDD (no CLIP needed). Specifically, we used the following metadata attributes to create semantic slices:

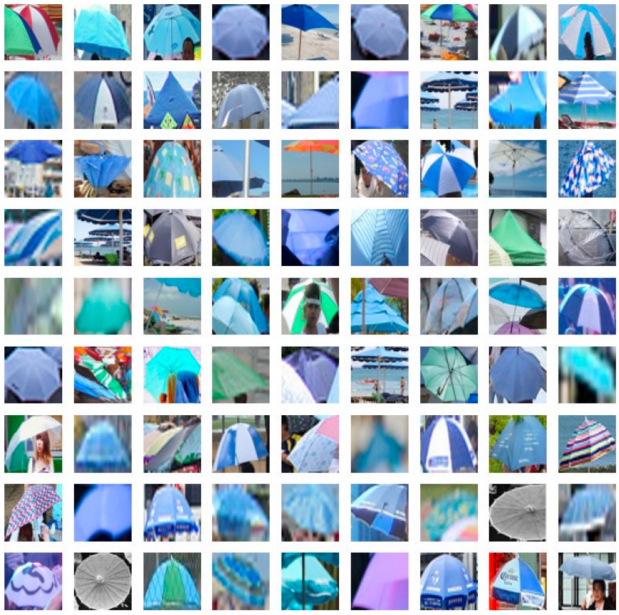
- timeofday: daytime, night, dawn, dusk, undefined
- weather: rainy, snowy, clear, overcast, partly cloudy, foggy, undefined
- scene: tunnel, residential, parking lot, city street, gas stations, highway, undefined

This leads to slices of the form "[object class] in [metadata attribute]." We display a few of these semantic slices in Figure 11. For the three metadata based settings, we do the same as with the COCO dataset, bucketing the different sizes and aspect ratios for the bounding boxes, and using K-Means to form color clusters.

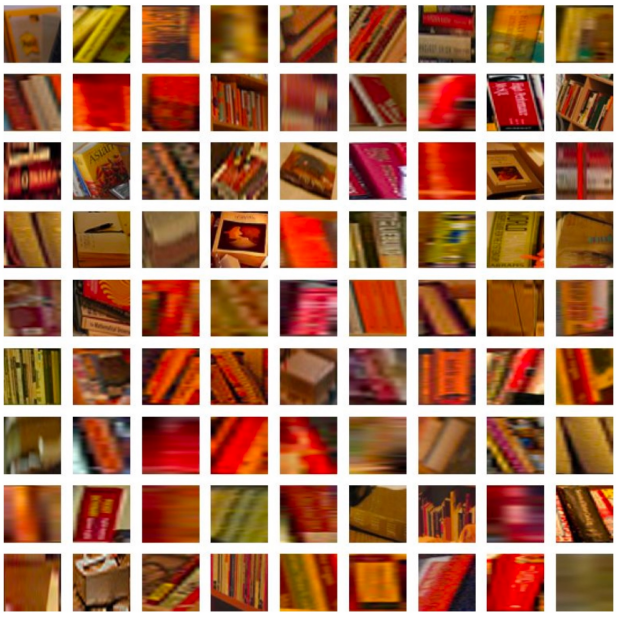
Figure 8: Example semantic ground truth slices generated from the COCO 2014 dataset.



Class Traffic Light Cluster 13



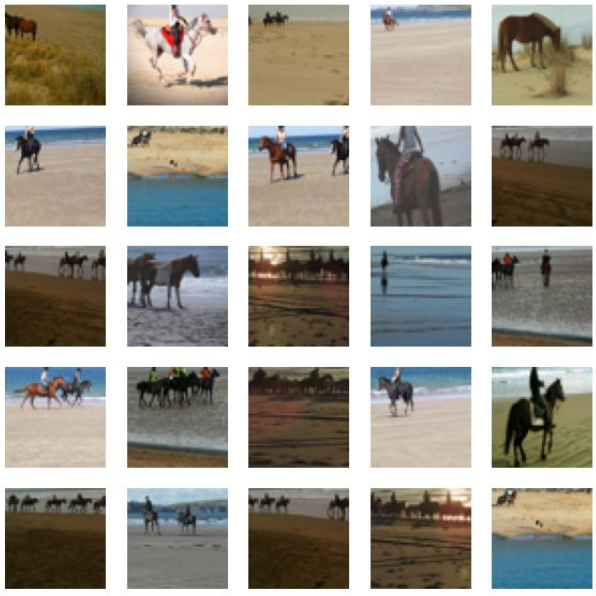
Class Umbrella Cluster 6



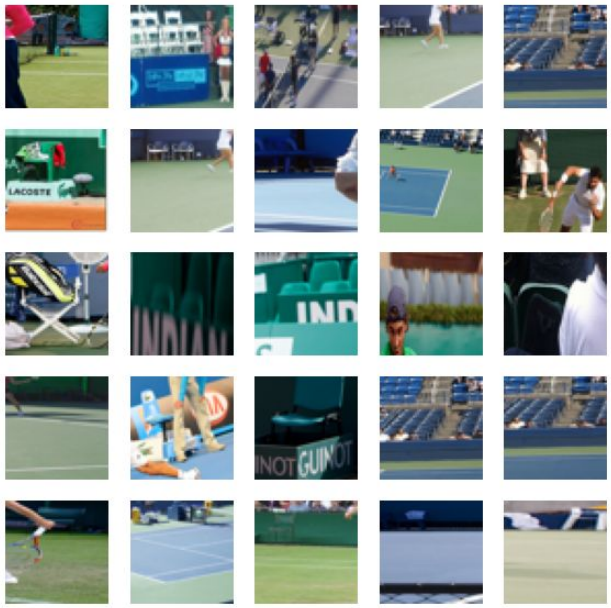
Class Book Cluster 14

Class Bowl Cluster 17

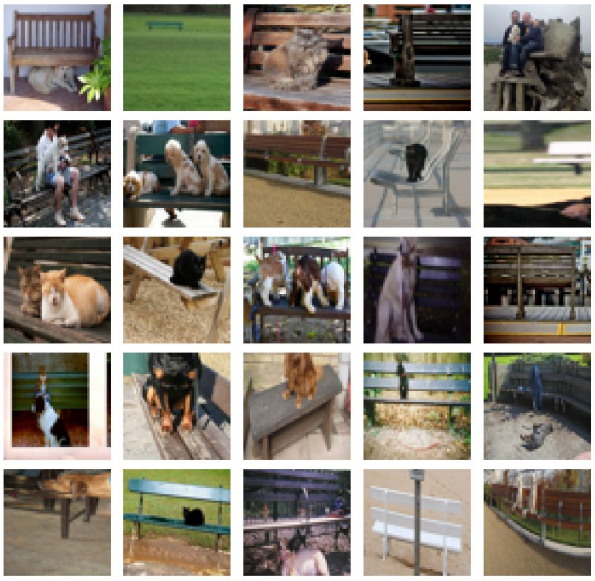
Figure 9: Example color slices generated from the COCO 2014 dataset.



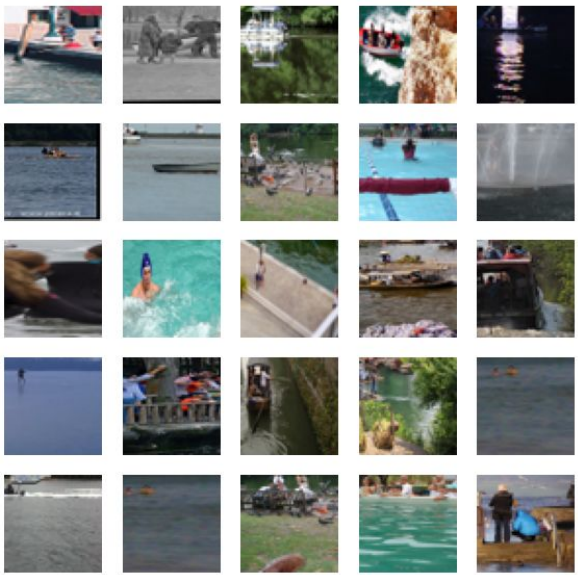
horse next to sand



chair next to tennis court

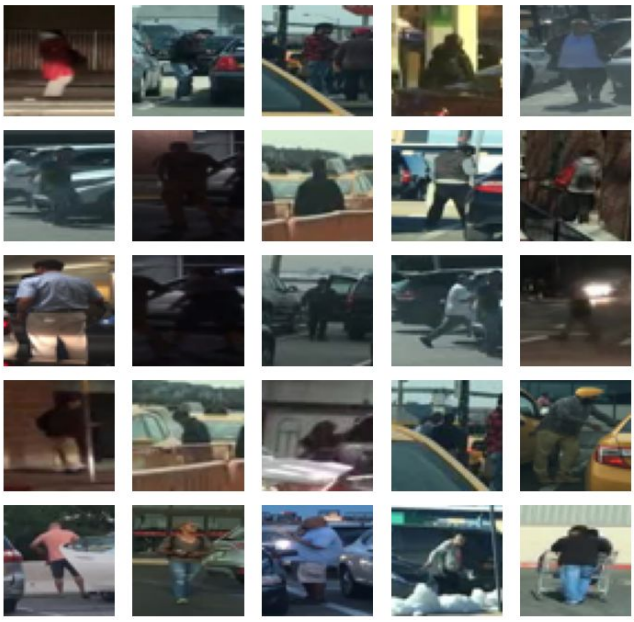


bench next to dog

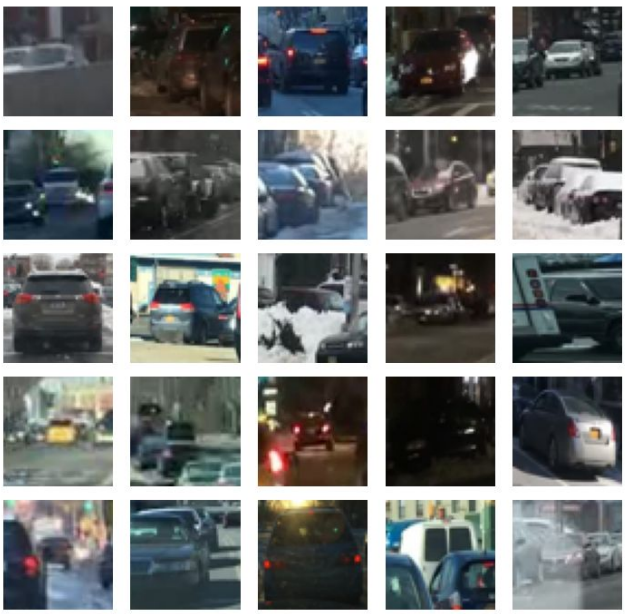


person next to water

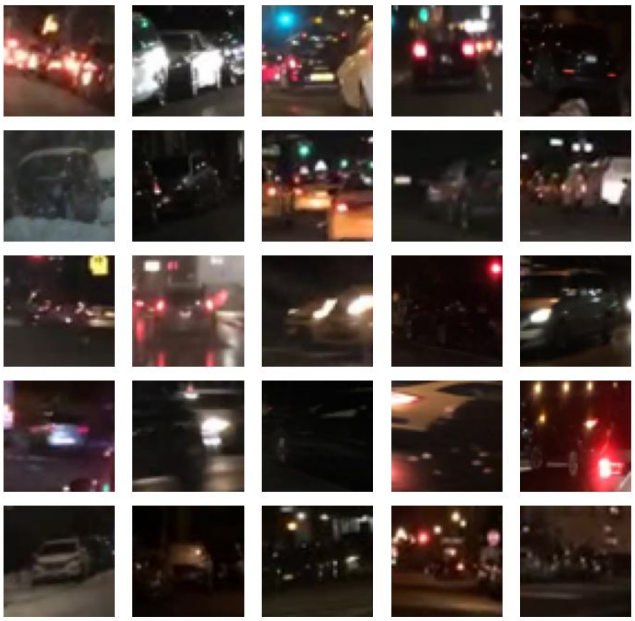
Figure 10: Example semantic ground truth slices generated from the Visual Genome dataset.



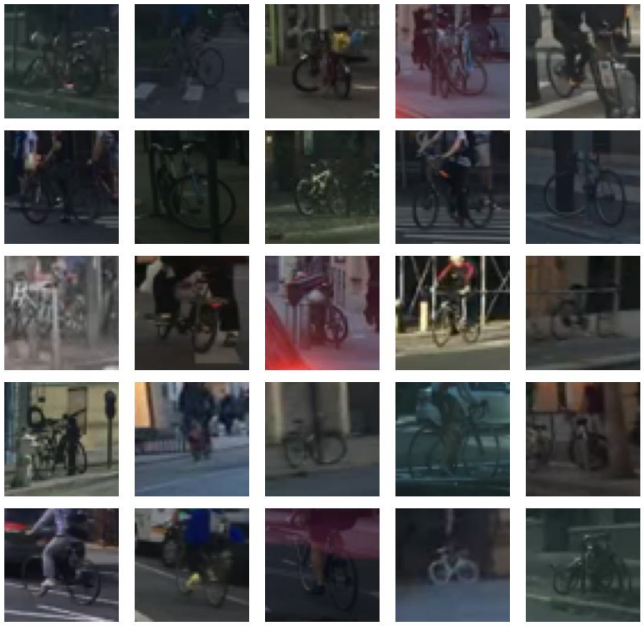
scene: parking lot
label: person



weather: snowy
label: car



timeofday: night
label: car



timeofday: dawn/dusk
label: bike

Figure 11: Example semantic ground truth slices generated from the BDD dataset. The first line under each image represents the semantic setting, the second line represents the target class.