ABSTRACT
Unknown unknowns represent a major challenge in reliable image recognition. Existing methods mainly focus on unknown unknowns identification, leveraging human intelligence to gather images that are potentially difficult for the machine. To drive a deeper understanding of unknown unknowns and more effective identification and treatment, this paper focuses on unknown unknowns characterization. We introduce a human-in-the-loop, semantic analysis framework for characterizing unknown unknowns at scale. We engage humans in two tasks that specify what a machine should know and describe what it really knows, respectively, both at the conceptual level, supported by information extraction and machine learning interpretability methods. Data partitioning and sampling techniques are employed to scale out human contributions in handling large data. Through extensive experimentation on scene recognition tasks, we show that our approach provides a rich, descriptive characterization of unknown unknowns and allows for more effective and cost-efficient detection than the state of the art.

CCS CONCEPTS
• Computing methodologies → Machine learning; Knowledge representation and reasoning; • Human-centered computing → Human computer interaction (HCI).

KEYWORDS
Unknown unknowns, humans in the loop, semantic analysis

1 INTRODUCTION
Machine learned image recognition models are rapidly deployed in many high-stakes contexts [35]. While largely accelerating and aiding the decision-making process, such models suffer from a severe issue of reliability—they can just as easily fail and generate errors that can eventually lead to drastic consequences [41]. Being able to understand and detect such errors has become a key demand for both, model developers to debug and improve the model [4], and for the users to decide when to trust the model output [11, 12, 50].

Among image recognition errors, a specific type known as unknown unknowns is of particular interest [5, 29]. Unknown unknowns refer to the images for which a model is highly confident about its predictions but is wrong. Identifying such errors is challenging due to the overconfidence of the model. Recent efforts resort to human-in-the-loop approaches that ask humans to gather data instances that are potentially difficult for a model to handle [5, 27, 29]. An important finding reveals that unknown unknowns often come with internal consistency, making them particularly suitable to be described by natural language building on top of conceptual knowledge [5, 17, 29]. We bring to fore the notion of characterizing unknown unknowns to allow us to gain a deeper understanding of when the model fails. This lies in contrast to previous work that has focused largely on identifying unknown unknowns.

For effective characterization of unknown unknowns, two types of knowledge are needed: knowledge of what a model has learned, that we henceforth refer to as REALLY-KNOWS, and what a model should have learned, referred to as SHOULD-KNOW. Recent work on human-in-the-loop machine learning interpretability [6] has shown the important role of humans as computational agents to describe REALLY-KNOWS, by annotating salient image areas in image recognition with semantic concepts. In this paper, we advocate another view to the role of humans as contributors who can shed light on SHOULD-KNOW. We envision that eliciting SHOULD-KNOW from the perspective of human understanding of a given task, can lead to a complete and usable characterization of unknown unknowns. Consider the example of indoor scene recognition in Figure 1, where the model incorrectly classifies a kitchen image as a Conference Room.
Room: knowing that the model fails by focusing on the chair and ceiling only tells half the story; knowing that the model should have focused on the microwave, oven, counter, for instance, presents a deeper understanding that would allow further identification of similar errors which the model produces by missing such concepts.

With this in mind, we introduce Scalpel-HS, a human-in-the-loop, semantic analysis framework for unknown unknowns characterization in image recognition. Drawing inspiration from cognitive psychology literature [16, 20, 30, 36], Scalpel-HS is designed with two human computation tasks— for SHOULD-KNOW specification and REALLY-KNOWS description—that both engage human contributors to operate at the conceptual level. In the SHOULD-KNOW task, human contributors identify a set of objects (with attributes) and relations that are relevant for a given image. In the REALLY-KNOWS task human contributors annotate areas of an image, shown to be relevant for model prediction, with semantic concepts (i.e., visual objects, attributes, and relations). Leveraging the outcome from both SHOULD-KNOW and REALLY-KNOWS tasks, model unknowns can be characterized by comparing those concepts that a model should have learned with what the model actually learned.

Scalpel-HS builds upon a computational pipeline to provide input to the human computation tasks with minimized cognitive load imposed to human contributors. For the SHOULD-KNOW task, we leverage state-of-the-art information extraction techniques to pre-identify objects and relations in the images, allowing human contributors to primarily focus on adjudicating the relevance of concepts to a given scene. This cognitively simplifies the task at hand, in comparison to explicitly synthesizing relevant concepts [25], and results in a more structured vocabulary. For the REALLY-KNOWS task, we leverage machine learning interpretability methods to highlight important pixels of an image for model prediction [45, 46]. To minimize human effort, Scalpel-HS employs a semantic data partitioning and sampling method that identifies representative images for the human tasks. To do so, Scalpel-HS starts off by first learning semantically-rich image representations.

We demonstrate the effectiveness, informativeness, and cost-efficiency of Scalpel-HS on several state-of-the-art machine learning models for scene recognition [26, 55], a task that is considered to be complex in image recognition for machines, as well as for humans as it requires the understanding of context [31, 54]. We show that Scalpel-HS provides informative, easy-to-understand characterizations of unknown unknowns that significantly boost the state of the art in unknown unknown detection by 31%, and is able to detect 2x to 3x the sizes of unknown unknowns compared to the number of annotated images.

In summary, we make the following key contributions:

- We introduce a human-in-the-loop framework that orchestrates both automatic and human computation components for cost-efficient characterization and identification of unknown unknowns;
- We present the design of human computation tasks for both model should-know and actually-knows description at the conceptual level, with a set of design choices made to account for the cognitive load, and fault-tolerance of human work;
- We introduce computational methods for learning semantically rich image representations and for image sampling by partitioning the semantic data space, for scaling out human contributions.

2 THE SCALPEL-HS FRAMEWORK

Figure 2 presents an overview of Scalpel-HS. Given an image set and a trained image recognition model, it first extracts the scene graphs of the images and then partitions the image set and sample representative images for the human tasks. The scene graphs and the saliency maps of the sampled images are then respectively fed to the human tasks published in a crowdsourcing platform, learns the representation of the images combining both the visual and semantic features and based on that, partitions the image set and sample representative images for the human tasks. The scene graphs and the saliency maps of the sampled images are then respectively fed to the human tasks.

In the following, we describe the components in more detail.

1a Scene Graph Extraction. Understanding a natural scene image usually requires reasoning about the relationship between objects in the image. For example, in the recognition of rooms, a sink next to an oven indicates a kitchen while a sink next to a mirror more likely indicates a bathroom. To help humans specify the required knowledge in scene recognition, i.e., SHOULD-KNOW, we extract scene graphs. A scene graph is a structured representation of objects and the relationships between them present in an image. It consists of a set of relationships, each represented as \([o_i, r_{ij}, o_j]\), where \(o_i\) and \(o_j\) refer to two objects (image patches usually captured by bounding boxes) in the image, and \(r_{ij}\) represents the relationship between two objects. Given a scene image, we generate the visual scene graph using state-of-the-art methods Neural Motifs [53].

1b Salience Map Extraction. Understanding machine behavior in scene recognition is the machine learning interpretability problem. The most extensively studied interpretability approach for image classification is saliency, a local interpretability post-hoc method that highlights the most important pixels of an image for model decisions in what is called a saliency map [45]. We choose this method to help humans describe what a model REALLY-KNOWS.

We opted for SmoothGrad [46], which is sensitive to the parameters of a model (thus catering for more accurate capturing of a model behavior) while minimizing noisy results (i.e., highlighting irrelevant pixels). Our framework is though agnostic to the employed local interpretability method. To minimize human effort, we sample a subset of representative images for human annotation. Data sampling is performed via a data partitioning method based on a new type of image representation.

1c Representation Learning. Due to the complexity of the scene recognition task, representative images should be diverse in terms of both the semantic information contained and the visual appearance. Existing methods generally rely on pre-trained models for visual feature extraction only, which is suboptimal in our context. We propose to fuse in also the semantic information and introduce a self-supervised learning approach for learning semantically rich image representations. We describe the details in Section 3.1.

1d Data Partitioning and Sampling. Prior work has shown that unknown unknowns are caused by systematic biases in training data, reside in certain partitions (i.e., blind spots) of the feature
Figure 2: The Scalpel-HS framework. It takes as input an image set and a trained image recognition model; as output, it produces a characterization of unknown unknowns and identifies the corresponding unknown unknown images. To do so, it extracts the (1a) scene graph and (1b) saliency map of model classification of a subset of images—sampled by (2a) representation learning and (2b) data partitioning—, feeds them to the (3a) SHOULD-KNOW and (3b) REALLY-KNOWS human computation tasks published on a crowdsourcing platform, and (4) aggregates the output for unknown unknowns characterization and detection of more corresponding unknown unknown images.

space [5, 29]. Following this, we propose a data partitioning method for sampling, Semantic Space Partitioning (SSP), that identifies the optimal subset of representative images in the semantic space. Our method partitions the semantic space and selects candidate images in such a way that (the weighted sum of) cosine distances from the candidate data points to others in the same region are minimized. As the result, semantically similar images will be grouped in partitions, centered around the representative images. Those representative images are then sampled for human annotations. We describe the details of our SSP method in Section 3.2.

3a The SHOULD-KNOW Task. For a given set of valid objects and relations pertaining to a scene, it is important to understand the salience of each object and relation in identifying the scene in the given image. For example, from a human perspective, a bed when compared to a carpet can be deemed to be relatively more salient in identifying the scene as a bedroom. In this task, human workers identify the salient objects, their attributes, and the relations between objects for identifying a given scene in an image. We describe details of the task design in Section 4.1.

3b The REALLY-KNOWS Task. The goal of the task is to find out which objects in the scene influence the prediction of the machine learning model, and whether this is congruent with the human mental model. Human workers identify objects and relations found by the machine, and rate their relevance in identifying the scene. Details of the task are described in Section 4.2.

4 Aggregation & Detection. Results of the two human tasks are aggregated to obtain a characterization of unknown unknowns. Denoting the true class and wrongly predicted class as \( y \) and \( y' \), respectively, the characterization is represented in the form of the triple \( (y, +), (y', -) \) for False Positive (in terms of \( y' \)) and \( (y, -, y') \) for False Negative (in terms of \( y \)). For example, \( \langle \text{Conference Room}, (+)\text{sofa}, \text{Living Room} \rangle \) indicates that the model wrongly classifies a conference room image to be a living room because of the focus on the spurious concept sofa; \( \langle \text{Kitchen}, (-)\text{oven}, \text{Conference Room} \rangle \) indicates that the model wrongly classifies a kitchen image to be a conference room by missing the concept of oven in the kitchen.

Apart from the characterization, this component detects more unknown unknowns of the same characteristics utilizing the data partitions: images in the same partition as the human annotated (representative) one are likely to be unknown unknowns sharing the same missing or spurious concepts. The component therefore, identifies more images as unknown unknowns those on which the model confidence is greater than a threshold.

3 IMAGE REPRESENTATION AND SAMPLING

This section describes our methods for semantically rich image representation learning and semantic space partitioning.

3.1 Representation Learning

Our representation learning model comprises components for visual feature extraction, semantic feature extraction, multi-modality fusion, and image representation generation. Together those components make a representation learning model that can be trained in an end-to-end fashion. We describe the details in the following. Visual Feature Extraction. We use the pre-trained model Faster-RCNN [38] to generate feature vectors for nodes and relationships in the generated scene graph. For each object node \( o_i \) in the scene graph, a visual feature vector \( V_{o_i} \) is extracted from its corresponding image region. For each relationship node \( r_{ij} \), its visual feature vector \( V_{r_{i,j}} \) is extracted from the union region of \( o_i \) and \( o_j \) on the image. Semantic Feature Extraction. Each node, either object node or relationship node, has a text description generated by the scene graph generator. From such text description, we obtain the initial semantic features using the pre-trained GloVe embeddings [34]. Those embeddings are trainable parameters in our representation method. We denote the semantic feature of the node and relationship as \( E_{o_i} \) and \( E_{r_{i,j}} \), respectively. Multi-modality Fusion. We fuse the visual and semantic features into a joint multi-modal representation. Inspired by [18], the visual
feature vector and label feature vector are first concatenated, then fused as follows:

\[ \begin{align*}
Z_{o_i} &= \tanh(W_1^T V_{o_i} + W_2 E_{o_i}) \\
Z_{r_{ij}} &= \tanh(W_1^T V_{r_{ij}} + W_2 E_{r_{ij}})
\end{align*} \tag{1} \]

where \(Z_{o_i}\) and \(Z_{r_{ij}}\) are joint feature vectors for object and relation nodes, respectively. \(W_1\) and \(W_2\) are the shared parameters.

**Image Representation Generation.** To obtain a single vector representation of an image, we combine the visual-semantic features of all the objects and relationships. Considering the fact that those objects and relations together make a graph, we employ a multi-layer graph convolutional networks (GCN) [24] to capture the graph structure while combining the object and relationship features. Through multiple layers of linear and non-linear transformation layers, GCN generates new node features that contain structural information of the graph. To obtain a global representation of the image, we aggregate the node representations learned by GCN in different layers—denoted as \(U_{o_i}\) and \(U_{r_{ij}}\) for object and relationship nodes respectively—into a graph representation using a graph pooling operation, known as readout [32, 52].

The entire representation learning model contains parameters of embedding, the multi-modality fusion layer, and those of the GCN. We describe the training details in Appendix A.1.

### 3.2 Semantic Space Partitioning

Formally, we denote the distance between the feature vectors \(f_i, f_j\) of two images \(i, j\) as following,

\[ \text{dist}(f_i, f_j) = 1 - \frac{f_i^T f_j}{\|f_i\| \cdot \|f_j\|}. \tag{3} \]

Given a budget \(B\), i.e., the number of representative images to be sampled for human tasks, our partitioning method finds the representations images by minimizing the following objective function:

\[ \begin{align*}
\min & \sum_{i \in \mathcal{D}} \sum_{j \in \mathcal{D}} \text{dist}(f_i, f_j) X_{ij} \\
\text{s.t.} & \sum_{j \in \mathcal{D}} X_{ij} = 1 \\
& X_{ij} \leq Y_j \\
& \sum_{j \in \mathcal{D}} Y_j = B
\end{align*} \tag{4} \]

where \(X_{ij}\) indicates the decision of whether image \(i\) is assigned to partition \(j\); \(Y_j\) indicates if image \(j\) is selected as a representative sample (note that the index \(j\) is overloaded to represent both the partition and the representative image of the partition).

Due to the large number of possible solutions that are associated with the problem of finding an optimal set of representative samples, it is very challenging to provide a deterministic solution. We employ a meta-heuristic approach based on genetic algorithms (GA). Details of our algorithm are described in Appendix A.2.

### 4 THE HUMAN COMPUTATION TASKS

We now describe the human computation tasks for specifying what an image recognition model SHOULD–KNOW and what it REALLY–KNOWS.

#### 4.1 The SHOULD–KNOW Task

In this task, human workers are presented with a sampled image, and the corresponding scene graph, to identify concepts (salient objects, their attributes, and relations) in scene recognition. The procedure is shown in Figure 3 (zoomed figures in Appendix A.3). Workers are first asked to add missing concepts, and then determine the indispensable concepts for identifying the scene in the given image. To add missing concepts, workers need to specify the concept by entering the name and drawing a bounding box in the image. In the case of the concept being an object attribute or a relationship, the worker needs to further specify the relations. Indispensable concepts are selected using checkboxes from the concept list.

#### 4.2 The REALLY–KNOWS Task

While a scene in the human mind is composed of objects possessing clear boundaries and having intelligible locations in space relative to each other [44], to the machine it is the composition of pixels rather than objects. To understand machine behavior, we map the pixels highlighted in warm colors by the saliency map, to actual concepts that humans can understand. The procedure is shown in Figure 4 (zoomed figures in Appendix A.3). Workers are asked to draw bounding boxes to annotate objects highlighted by the saliency map, name the objects and assign the attributes (e.g., color), define relations among the objects, and add all the objects and relations highlighted by the saliency map to the list.

By comparing to their own mental models in scene recognition, workers then rate the relevance these objects/relations are in identifying the scene. Using a slider (ranging from 1 to 20, meaning not relevant at all to highly relevant). They are also encouraged to give reasons. Note that with annotations from this task, a characterization of false positive prediction due to the wrong focus on the spurious concept can already be obtained. We compare in our experiments unknown unknowns detection using only the characteristic obtained from this REALLY–KNOWS task and that from also the SHOULD–KNOW task.

### 5 EXPERIMENTAL SETUP AND RESULTS

We evaluate the performance of **Scalpel–HS** by investigating the following questions:\footnote{Source code and data are available at https://sites.google.com/view/www22-scalpel-hs} **Q1:** How effective is it in detecting and characterizing unknown unknowns? **Q2:** How informative are those characteristics provided by our framework? and **Q3:** How cost-efficient is our framework under a limited budget? For these questions, we further evaluate the contribution of the individual components of **Scalpel–HS** and compare to the state of the art whenever possible.
Rating the relevance of relations and giving reasons

Defining the minimum relation set for scene identification

Adding missing relations

Defining the minimum object set for scene identification

Figure 3: The procedure of the SHOULD-KNOW task for specifying what a model should know in scene recognition. (Zoomed in Appendix A.3)

Drawing bounding boxes to annotate objects highlighted by the heatmap

Naming objects and assigning attributes

Defining relations using the objects annotated

Adding all the objects and relations highlighted by the heatmap

Rating the relevance of objects/relations for identifying the scene

Figure 4: The procedure of the REALLY-KNOWS task for describing what a model really knows in scene recognition. (Zoomed in Appendix A.3)

5.1 Experimental Setup

Datasets. We use two image datasets: (1) PLACES: it contains 10 million images divided into over 400 unique scene classes with 5000 to 30,000 training images and over 100 test images per class. As not all classes are about scenes, we select a subset of data containing nine indoor scene classes. (Details provided in the Appendix A.4.)

The subset contains 60,000 training and 1000 test images equally distributed across the nine classes. (2) MIT67: this dataset contains 15,620 images in 67 indoor classes. Unlike the PLACES dataset, the number of images varies across classes, however, there are at least 100 images per class. We filter images of the same set of scene classes as PLACES and select a subset consisting of 3224 test images with at least 100 images per class. Note that due to the limited number of images in MIT67, there are only 3216 images left after filtering; we therefore only use the training set of PLACES for model training. Test sets from PLACES and MIT67 allow us to experiment with model unknowns exposed in test data of different distributions.

Unknown Unknowns Creation. We consider the unknown unknowns characterization effective in two senses: it exposes the reasoning of an image recognition model in a high-confidence yet wrong prediction, and it allows for the detection of unknown unknowns of False Negative by removing concepts from the training images of the class of interest (not other classes). To make sure the concepts are distributed in several classes, we select 15 most frequent concepts (objects and relations) and then those that are distributed across at least three classes. A co-occurrence matrix between concepts and scene classes is provided in the companion page. The induced unknown unknowns are summarized in Table 1.

Apart from evaluating the effectiveness of our framework in exposing the incorrect reasoning and in detecting unknown unknowns that are manually induced, we further look into the informativeness of the characterization for “natural” unknown unknowns, i.e., those unknown unknown images without the chosen concepts (or missed by the scene graph extractor). Note that while we cannot make sure that the model makes high-confidence errors on all images with the identified characteristics, we are sure about the errors when they occur given the ground truth labels and about model rationales for images annotated by our framework.

Scene Recognition Models. We conduct our experiment with two state-of-the-art convolution neural networks, ResNet[19] and DenseNet[23], which have shown superior performance on various classification tasks [48]. We train the two scene classifiers for 50 epochs on the biased training data and confirm that biases are successfully injected into the scene classifier by observing overfitting of the performance metrics during the training phase.

<table>
<thead>
<tr>
<th>Type</th>
<th>Index</th>
<th>Class of Interest</th>
<th>Concept</th>
</tr>
</thead>
<tbody>
<tr>
<td>False Positive</td>
<td>FP1</td>
<td>Kindergarten</td>
<td>person</td>
</tr>
<tr>
<td></td>
<td>FP2</td>
<td>Bedroom</td>
<td>bed</td>
</tr>
<tr>
<td></td>
<td>FP3</td>
<td>Conference Room</td>
<td>chair</td>
</tr>
<tr>
<td>False Negative</td>
<td>FN1</td>
<td>Kitchen</td>
<td>oven</td>
</tr>
<tr>
<td></td>
<td>FN2</td>
<td>Bathroom</td>
<td>sink</td>
</tr>
<tr>
<td></td>
<td>FN3</td>
<td>Dining Room</td>
<td>wine glass</td>
</tr>
<tr>
<td></td>
<td>FN4</td>
<td>Living Room</td>
<td>couch/sofa</td>
</tr>
<tr>
<td></td>
<td>FN5</td>
<td>Conference Room</td>
<td>woman at table</td>
</tr>
<tr>
<td></td>
<td>FN6</td>
<td>Kindergarten</td>
<td>boy wearing shirt</td>
</tr>
</tbody>
</table>
Table 2: Performance (P = Precision, R = Recall, and F = F1-score) comparison with baseline methods on detecting unknown unknowns images. We highlight the best performance for each metric in bold.

<table>
<thead>
<tr>
<th>Type</th>
<th>Comparison Method</th>
<th>Places ResNet</th>
<th>Places DenseNet</th>
<th>MIT67 ResNet</th>
<th>MIT67 DenseNet</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>P  R  F</td>
<td>P  R  F</td>
<td>P  R  F</td>
<td>P  R  F</td>
</tr>
<tr>
<td>False Positive</td>
<td>Random</td>
<td>0.383 0.187 0.251</td>
<td>0.311 0.158 0.209</td>
<td>0.39 0.161 0.228</td>
<td>0.336 0.120 0.177</td>
</tr>
<tr>
<td></td>
<td>Least Average Similarity</td>
<td>0.558 0.272 0.366</td>
<td>0.318 0.161 0.214</td>
<td>0.558 0.272 0.366</td>
<td>0.663 0.218 0.329</td>
</tr>
<tr>
<td></td>
<td>Least Maximum Similarity</td>
<td>0.379 0.185 0.249</td>
<td>0.232 0.118 0.156</td>
<td>0.379 0.185 0.249</td>
<td>0.616 0.209 0.312</td>
</tr>
<tr>
<td></td>
<td>Most Uncertain</td>
<td>0.348 0.170 0.228</td>
<td>0.44 0.223 0.296</td>
<td>0.351 0.183 0.240</td>
<td>0.53 0.190 0.279</td>
</tr>
<tr>
<td></td>
<td>UUB</td>
<td>0.629 0.378 0.472</td>
<td>0.755 0.383 0.509</td>
<td>0.617 0.394 0.480</td>
<td>0.702 0.282 0.402</td>
</tr>
<tr>
<td></td>
<td>Scalpel-HS</td>
<td>0.855 0.522 0.648</td>
<td>0.915 0.465 0.816</td>
<td>0.874 0.716 0.787</td>
<td>0.766 0.521 0.620</td>
</tr>
<tr>
<td>False Negative</td>
<td>Random</td>
<td>0.21 0.08 0.11</td>
<td>0.28 0.152 0.197</td>
<td>0.293 0.271 0.281</td>
<td>0.33 0.09 0.141</td>
</tr>
<tr>
<td></td>
<td>Least Average Similarity</td>
<td>0.542 0.279 0.368</td>
<td>0.663 0.218 0.329</td>
<td>0.542 0.279 0.368</td>
<td>0.589 0.155 0.246</td>
</tr>
<tr>
<td></td>
<td>Least Maximum Similarity</td>
<td>0.372 0.185 0.247</td>
<td>0.616 0.209 0.312</td>
<td>0.372 0.185 0.247</td>
<td>0.495 0.135 0.212</td>
</tr>
<tr>
<td></td>
<td>Most Uncertain</td>
<td>0.585 0.219 0.319</td>
<td>0.452 0.246 0.319</td>
<td>0.549 0.237 0.331</td>
<td>0.44 0.12 0.188</td>
</tr>
<tr>
<td></td>
<td>UUB</td>
<td>0.551 0.634 0.589</td>
<td>0.480 0.376 0.422</td>
<td>0.553 0.649 0.597</td>
<td>0.456 0.271 0.340</td>
</tr>
<tr>
<td></td>
<td>Scalpel-HS</td>
<td>0.711 0.525 0.604</td>
<td>0.653 0.435 0.522</td>
<td>0.577 0.678 0.624</td>
<td>0.704 0.364 0.480</td>
</tr>
</tbody>
</table>

Baseline Methods. Following previous work [27, 29], we compare the performance of our pipeline against the following methods: 1) Random Sampling: Randomly selects instances from the test data to be queried with humans. 2) Least Average Similarity [8]: Computes the average Euclidean distance for each test instance to all training instances, and chooses the instances with the highest distances. 3) Least Maximum Similarity [8]: Computes the minimum Euclidean distance of test data instances to all training data instances and chooses instances with the highest distances. 4) Most Uncertain [42]: Ranks the instances in the test dataset by increasing order of the prediction confidence as assigned by the scene classification model. 5) UUB [27]: Combines clustering and the bandit algorithm to query an Oracle. Least Average Similarity and Least Maximum Similarity are popular outlier detection methods; Most Uncertain is similar to the uncertain sampling strategy used in active learning; and UUB is the state of the art unknown unknowns detection method. Apart from those, we further compare with variations of our own framework considering only output from the REALLY-KNOWS task, and those with other baseline representation learning and data sampling methods.

Evaluation Metrics. For effectiveness evaluation, we use Precision and Recall to measure the performance on unknown unknowns identification. We consider detection performance in both cases when we are sure the unknown unknowns happen due to the injected data biases and in the entire test set.

Crowdsourcing. We crowdsourced 300 images from each dataset, selected by our feature space partitioning method. We present the same set of images for both the SHOULD-KNOW and REALLY-KNOWS tasks. For each task, we recruited 300 workers on Prolific.

5.2 Scalpel-HS Performance

Effectiveness. Table 2 reports the performance of comparison methods on detecting unknown unknown images.

We observe that among the baselines, Most Uncertain, which is widely used in detecting known unknowns, yields low performance (similar to Random), providing evidence for the important difference between the problems of detecting known unknowns and unknown unknowns. Among the two outlier detection methods, Least Average Similarity generally outperforms Least Maximum Similarity, indicating that unknown unknowns are distant to the general image population in the feature space. UUB, which considers model confidence, gives better performance than all the other baseline methods, showing that unknown unknowns are related to not only the data but also what models have learned from the data. Most importantly, our proposed framework Scalpel-HS achieves the best performance across all settings (unknown unknown types, datasets, and metrics), and outperforms UUB by a significant margin of 31% in F1-score, a strong evidence demonstrating the effectiveness of our framework in unknown unknowns detection.

The relative detection performance of Scalpel-HS on the two types of unknown unknowns (False Positive vs. False Negative) is consistent across datasets given the same model; similarly, it is consistent across models given the same dataset. Those results show the robustness of our framework in unknown unknowns detection.

Informativeness. To gain a deeper understanding of the informativeness of unknown unknowns characterization provided by Scalpel-HS, we report in Table 3 its performance on uncovering the exact reasoning of the model on unknown unknown images. We observe that our framework successfully exposes the characteristics of all manually created unknown unknowns (except FP3 on MIT67, which corresponds to 13 images only), showing the strong characterizing power of our framework for unknown unknowns. We find a large variability of the performance in detecting unknown unknown images with different characteristics, showing the specificity of unknowns characteristics for detection. As a remark, we note there is a discrepancy between the overall performance in Table 3 and Table 2; this is due to the presence of "natural" unknown unknowns that are not manually induced.
We now evaluate the contribution of our representation learning with 300 images annotated, accounting for 3% of the overall test. With the identified characteristics, we reach a recall of 42%. Those results show that our framework allows to detect 19% extra False Positive natural unknown unknowns, and 38% extra False Negative natural unknown unknowns.

We show samples of manually induced unknown unknowns detected by our framework in Appendix A.5 and more (including natural ones) in the companion page.

Cost-Efficiency. Figure 5 depicts the performance of Scalpel-HS under different budgets. As expected the precision decreases and recall increases when the budget increases; yet, we observe that the decrease in precision is much slower than the increase in recall. With 300 images annotated, accounting for 3% of the overall test images in PLACES and 20% of the unknown unknowns with the identified characteristics, we reach a recall of over 60% of all the unknown unknowns; on MIT67, the 300 annotated images account for 9% of the overall test images and 19% of the unknown unknowns with the identified characteristics, we reach a recall of 42%. Those results show that our framework allows to detect 2x to 3x unknown unknowns w.r.t. a given budget, demonstrating that our framework is highly cost-efficient.

5.3 Contribution by Automatic Components

We now evaluate the contribution of our representation learning and image sampling methods. Figures 6 (a,b) compare the performance of our framework with our proposed semantically-rich representation learning method and the method with visual features only for representation learning (ResNet152 pre-trained on ImageNet and fine-tuned on our dataset, the rest components of the framework kept the same). The result shows that our proposed representation learning method is a better approach in both precision and recall across almost all budgets, signifying the utility of semantic features in the images for image sampling, and ultimately for unknown unknowns characterization and detection.

Figures 6 (c,d) compare the performance of our framework with our proposed semantic space partitioning (SSP) and other baseline data partitioning or sampling methods. These include: 1) Random Sampling; 2) DSP [27]: optimizes the overall distances within data partitions (minimization) and across partitions (maximization), and then randomly sample representative images. 3) K-means: generates clusters and the nearest instance to the center of each cluster (mean) is selected as the anchor of that partition. We observe that SSP achieves much higher precision with comparable recall (higher when the budget increases), showing the superiority of our partitioning methods in sampling representative unknown unknowns and the effectiveness of joint partitioning and sampling.

5.4 Impacts of SHOULD-KNOW and REALLY-KNOWS

We evaluate the effect of including human annotations, and in particular, human specification of what a model should know, by comparing the following configurations of our framework: 1) no human task, 2) including only the REALLY-KNOWS task, and 3) including both the REALLY-KNOWS and SHOULD-KNOW tasks.

Figure 7 compares the performance of those configurations under different budgets. We observe that involving human annotations has a big impact on the precision under any budget. In addition, we observe that integrating human annotations has more impact on the recall as the budget increases. Compared to the version of no human tasks, our framework improves by 26% and 10% in precision and recall, respectively (budget=300). Compared to REALLY-KNOWS only, with SHOULD-KNOW the performance of Scalpel-HS improves by 5% in both precision and recall. We also notice that the SHOULD-KNOW task is especially useful for precise identification of unknown unknowns when the budget is low.
and machine learning behavior interpretation, supported by automatic computational methods for scaling out human contributions.

**Automatic and Human Methods.** Unknown unknowns arise from biases in the training data. As such, methods developed for outlier detection are relevant for unknown unknowns detection. Typical methods can be characterized as either parametric [1, 2] or non-parametric [13–15] i.e., with or without assumptions on the underlying data distributions. In unknown unknowns detection, outlier detection methods are limited in that 1) they assume the accessibility to the reference data (i.e., training data) which is not necessarily available as in our setting, and 2) they do not take into account what a model has learned thus is limited to identifying model unknowns as we have shown in our experiment.

Another closely related line of work is human-in-the-loop (HiTL) machine learning, where human intelligence has been leveraged to address inherent limitations of ML such as reliability and interpretability. Early work in HiTL methods mainly focuses on leveraging human intelligence for data labeling [10, 37]. More recent work has investigated the advantage of human computation in debugging ML system components [33] and in identifying biases and noisy labels in the data [22, 51]. The most closely related work, as we discussed, is Lakkaraju et al. [27] and Liu et al. [29] that use HiTL methods for unknown unknowns detection. Recent work that has directly inspired ours is Balayn et al. [6] that propose to use human computation to interpret the behavior of image classifiers by attaching semantic concepts to the saliency maps of classification. We employ this method for unknown unknowns characterization in image recognition, and take a step further to show that by including human specified requirements of what a model should know, we can significantly improve unknown unknowns characterization.

**7 CONCLUSION**

We presented Scalpel-HS, a human-in-the-loop, semantic analysis framework for characterizing and detecting unknown unknowns of image recognition models. It involves human contributors to specify both what the model should know and describe what it really knows, while minimizing the cognitive load of the tasks leveraging scene graph extraction and machine learning interpretability techniques. It scales out human contributions through both a new semantically-rich representation learning and data sampling method. Our extensive evaluation on multiple models and datasets with different types of unknown unknowns demonstrates that characterizations provided by Scalpel-HS are not only informative but also highly effective and cost-efficient for unknown unknowns detection.
A APPENDIX

A.1 Learning Image Representations

To obtain graph-level representations, we train the graph encoder end-to-end using the approach proposed by Sun et al. [47], by maximizing the mutual information between graph-level and patch-level representations. The patch-level representation refers to the representation of nodes learned by aggregating the features of their neighborhood nodes, at the last GCN layer, while the graph-level representation is a fixed length vector obtained by pooling all patch-level representations. To train our graph encoder, we define the following objective function:

$$
\max \frac{1}{|G|} \sum_{g \in G} \left[ \frac{1}{|g|} \sum_{i=1}^{|g|} D \left( \tilde{U}_i, \tilde{U}_g \right) \right]
$$

where $|G|$ is the number of graphs in train set, $|g|$ is the number of nodes in graph $g$, and $\tilde{U}_i, \tilde{U}_g$ are representations of node $i$ and graph $g$, respectively. $D$ denotes a mutual information estimator which is modeled as a discriminator to score the agreement between patch-level and graph-level representations. The agreement score is obtained by simply computing the dot product between two representations.

Hyperparameter Setting. The number of GNN layers are chosen from {4, 8, 12}. Initial learning rate is chosen from the set {0.01, 0.001, 0.0001}. We set the batch size to 64. The number of epochs are chosen from {30, 60, 90}.

A.2 Semantic Space Partitioning Algorithm

The input of our GA-based method is a set of data points $\mathcal{D} = \{d_1, d_2, ..., \}$ where each $d_i$ consists of (feature vector, weighted concept) pairs, and the number of representative samples $\mathcal{B}$ to be found. Note that the number of generated partitions is equal to the number of representative samples. We initialize the genetic algorithm by constructing a population of random chromosomes $\mathcal{P} = \{p_1, p_2, ..., \}$, where each chromosome $p_i$ consists of $\mathcal{B}$ candidate samples. Algorithm 1 implements the fitness function, which corresponds to the proposed objective function (See Eq.4). The fitness function guides the exploration through the search space towards an optimal solution.

GAs are prone to premature convergence to local optima. Inspired by [40], we address this problem by adjusting the mutation rate (Pm) while the algorithm explores the search space. To avoid the generation of invalid solutions and to improve the performance of the GA, we use a greedy approach to the mutation process, which mutates the offspring only if the mutated solution gains lower fitness value.

Hyperparameter Setting. Population, elitism, mutation-rate and cross-over rate are regarded as hyper parameters, and therefore can be found via grid search ($N \in \{50, 100, 150, 200\}$; elitism $E \in \{5, 10, 15, 20, 30\}$; mutation rate $Pm \in \{0.0005, 0.001, 0.005, 0.01, 0.05\}$; and cross-over rate $Pc \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$). We set $N = 100$; $E = 20\%$; $Pm = 0.005$ and $Pc = 0.7$. We used a stagnation based termination criterion; following [3], we terminate the algorithm after $\sqrt{N}$ generations, where $N$ denotes the number of representative samples to be found, and $\mathcal{D}$ is the number of data points.

Algorithm 1 Fitness algorithm

1: procedure Fitness($\mathcal{D}, F, W$)
2: fitness $\leftarrow 0$
3: for $d \in \mathcal{D}$ do
4: max_similarity $\leftarrow \infty$
5: for $r \in R$ do
6: $f_r \leftarrow GetWeightedFeatures(F, W, r)$
7: $f_d \leftarrow GetFeatures(F, d)$
8: similarity $\leftarrow \cos(f_r, f_d)$
9: if similarity $\leq$ max_similarity then
10: max_similarity $\leftarrow$ similarity
11: end if
12: end for
13: fitness $\leftarrow$ fitness + max_similarity
14: end for
15: return fitness
16: end procedure

Figure 8: Example of unknown unknowns characterized and detected by Scalpel-HS: (upper-row) False Negative <Living room, (-)sofa, Dorm room> and (lower-row) False Positive <Conference room, (+)person, Kindergarten>. For each case, we show the sampled representative image with relevant concepts on the left and an additional similar unknown unknown image on the right. All images are shown together with a corresponding saliency map showing where the model is attending to in making the incorrect prediction. Note that in the False Negative case, the sofa leads to False Negative w.r.t. Living Room yet False Positive w.r.t. Dorm Room.

A.3 Annotation Workflow in Large Figures

Figures 9 and 10 show the zoomed version of our SHOULD-KNOW and REALLY-KNOWS human computation tasks.

A.4 Experimental Setup Details

We filter the two datasets by keeping images of the following scene categories: dining room, bathroom, conference room, kindergarten, hospital room, kitchen, living room, bedroom, dorm room.

A.5 Examples of Unknown Unknowns Identified by Scalpel-HS

We show a few examples of unknown unknown images Scalpel-HS identified and characterized, for both False Positive and False Negative. Figure 8 illustrates two characterizations of each type for both the sampled images (i.e., anchors) two similar ones detected by our framework.
Validating relations (correct or wrong)
Rating the relevance of relations and giving reasons
Adding missing relations
Defining the minimum relation set for scene identification
Defining the minimum object set for scene identification

Figure 9: The procedure of the SHOULD-KNOW task zoomed.

Drawing bounding boxes to annotate objects highlighted by the heatmap
Naming objects and assigning attributes
Defining relations using the objects annotated
Adding all the objects and relations highlighted by the heatmap
Rating the relevance of objects/relations for identifying the scene

Figure 10: The procedure of the REALLY-KNOWS task zoomed.