ScenarioNet: An Interpretable Data-Driven Model for Scene Understanding

Zachary A. Daniels[1], **Dimitris Metaxas**[1]
[1] Department of Computer Science, Rutgers University
zad7@cs.rutgers.edu, dnm@cs.rutgers.edu

Abstract

The ability for computational agents to reason about the high-level content of real world scene images is important for many applications. Existing attempts at complex scene understanding lack representational power, efficiency, and the ability to create robust meta-knowledge about scenes. We introduce **scenarios** as a new way of representing scenes. The scenario is an interpretable, low-dimensional, data-driven representation consisting of sets of frequently co-occurring objects that is useful for a wide range of scene understanding tasks. Scenarios are learned from data using a novel matrix factorization method which is integrated into a new neural network architecture, the **ScenarioNet**. Using ScenarioNet, we can recover semantic information about real world scene images at three levels of granularity: 1) scene categories, 2) scenarios, and 3) objects. Training a single ScenarioNet model enables us to perform scene classification, scenario recognition, multi-object recognition, content-based scene image retrieval, and content-based image comparison. ScenarioNet is **efficient because it requires significantly fewer parameters than other CNNs** while achieving similar performance on benchmark tasks, and it is **interpretable because it produces evidence in an understandable format** for every decision it makes. We validate the utility of scenarios and ScenarioNet on a diverse set of scene understanding tasks on several benchmark datasets.

1 Introduction

For many applications (e.g., robotics, human-machine teaming, surveillance, and autonomous vehicles), an agent must reason about the high-level content of real world scene images in order to make rational, grounded decisions that can be trusted by humans. It is often also necessary to have models that are able to be interpreted by humans in order to further encourage trust and allow humans to understand the failure modes of the autonomous agent. For example, if a self-driving car makes an error, it is important to know what caused the error to prevent future situations where similar errors might arise. Recently, a lot of progress has been made in constructing algorithms and systems that address fundamental scene understanding tasks such as scene classification, object detection, and semantic segmentation as well as more complex scene understanding tasks such as visual question-answering, automatic relationship extraction, scene graph generation, and learning how to visually reason about objects in simple scenes (e.g., [Johnson *et al.*, 2016]). While existing methods for solving such tasks are impressive, they often lack the interpretability and semantic grounding needed to make them trustworthy for safety-critical tasks and tasks involving human-machine teaming.

In this paper, we present a novel interpretable data-driven model for scene understanding. Explainable machine learning models rely on two properties: 1) features should be low-dimensional and human-interpretable and 2) models should be simple (with few parameters), easy for humans to inspect, and operate in a principled, well-understood way. We introduce a low-dimensional, semantically-grounded, object-based representation for scene understanding called the "scenario" which addresses the first property. We then show how scenarios can be used to make convolutional neural networks (CNNs) more transparent, thus addressing the second property.

We introduce **scenarios**, an interpretable, data-driven representation for scene understanding. Scenarios are based on *sets of frequently co-occurring objects*. Scenarios should satisfy a few key properties:

1. Scenarios are composed of one or more objects.
2. The same object can appear in multiple scenarios, and this should reflect the context in which the object appears, e.g., {keyboard, screen, mouse} and {remote control, screen, cable box} both contain the "screen" object, but in the first scenario, the screen is a computer monitor, and in the second scenario, it is a television screen.
3. Scenes can be decomposed as combinations of scenarios, e.g., a bathroom scene instance might decompose into: {shower, bathtub, shampoo} + {mirror, sink, toothbrush, toothpaste} + {toilet, toilet paper}.
4. Scenarios are flexible and robust to missing objects. A scenario can be present in a scene without all of its constituent objects being present.

We propose **Pseudo-Boolean Matrix Factorization (PBMF)** to identify scenarios from data. PBMF takes a binary *Object-Scene* matrix and decomposes it into 1) a dictionary matrix where each basis vector is a scenario and 2) an encoding matrix that expresses a scene instance as a combination of scenarios. We integrate PBMF into a novel convolutional neural network architecture (CNN), the **ScenarioNet**.

ScenarioNet replaces the final convolutional layers in standard CNNs with the **scenario block** (see Fig. 1) which consists of three parts: 1) global pooling layers that identify the parts of an image ScenarioNet attends to when recognizing whether each scenario is present in an image, 2) layers that use a PBMF-based loss function to learn a dictionary of scenarios and predict the presence and strength of each scenario for a given image, and 3) layers equivalent to a multinomial logistic regression model that use scenarios as low-dimensional features for predicting the scene category. During training, ScenarioNet only requires information about the *presence* (but not location) of objects in an image. For scene classification, class labels are also needed during training. During testing, only images are given.

Using ScenarioNet, we can recover semantic information about scene images at three levels of granularity: 1) scene categories, 2)

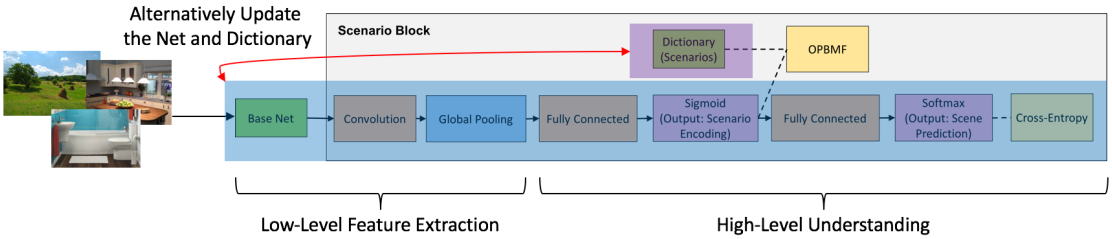

Figure 1: The **scenario block** replaces the final fully connected layers of a standard CNN and consists of: 1) global pooling layers that identify which parts of an image ScenarioNet attends to when recognizing whether a scenario is present in a given image, 2) layers that use a PBMF-based loss function to finetune a dictionary of scenarios and predict the presence of each scenario for a given image, and 3) layers equivalent to multinomial logistic regression that use scenarios as low-dimensional, interpretable features for scene classification.

scenarios, and 3) objects. This allows us to train a single ScenarioNet model capable of performing 1) scene classification, 2) scenario recognition, 3) multi-object recognition, 4) content-based scene image retrieval, and 5) content-based image comparison.

ScenarioNet has several advantages over other CNNs. It is computationally efficient because it requires significantly fewer parameters than other CNNs in order to achieve similar performance on benchmark tasks, and it is interpretable because it produces semantically- and visually-grounded evidence when making decisions. For example, for scene classification, predicted scenarios are used as low-dimensional semantic features; humans can verify the presence of each predicted scenario in an image by examining the scenario-localizing attention maps produced by the network; and humans can inspect how much influence each scenario exerts when assigning a class. This helps us to understand how a network arrives at specific decisions.

We evaluate the utility of ScenarioNet using the SUNRGBD [Song *et al.*, 2015], ADE20K [Zhou *et al.*, 2017b], and MIT 67 Indoor Scenes [Quattoni and Torralba, 2009] datasets. We perform quantitative experiments on multi-object recognition, scene classification, and content-based image retrieval. We also show examples demonstrating the interpretability and expressiveness of ScenarioNet.

2 Related Work

2.1 Learning Meaningful Groups of Objects

Discovering meaningful groups of objects is not a new idea. The simplest object-based representations are those that utilize pairwise co-occurrence relationships between objects (e.g., [Rabinovich *et al.*, 2007]). Scenarios go one step further by efficiently learning groups of objects of varying size. Many works focus on hierarchical models relating objects and scenes. [Feng and Bhanu, 2016] constructs a tree-based hierarchy of concepts based on object co-occurrence graphs. Objects sharing an ancestor node can be grouped into scene concepts, an idea similar to our scenarios. Several issues exist with using a tree structure for specifying scene concepts. To compute explicit scenarios, one must identify where to cut the tree. Additionally, while individual concepts can belong to multiple scene concepts by cutting the tree at different ancestor nodes, it becomes hard to properly place objects in the hierarchy that serve different functions within different groups, e.g., a screen with a keyboard and mouse is different from a screen with a cable box and remote. Our scenarios address these issues and provide additional information, e.g., how important each object is to a given scenario and how to decompose scene instances into combinations of scenarios. Other tree-based and hierarchical models for scene understanding exist. [Choi *et al.*, 2012] introduces a tree structure where nodes represent objects and latent variables and edges represent positive and negative correlations between nodes. These trees implicitly capture scenarios while our work learns explicit scenarios. [Fan *et al.*, 2008] exploit hierarchies of concepts to build ontologies for content-based image retrieval. [Lan *et al.*, 2013] investigate context at three levels: individual objects, parts of objects, and visual composites.

Other groups focus on using sets of objects to aid object detection. [Li *et al.*, 2012] discovers groups of objects of arbitrary size, model these groups using deformable parts models, and directly detects these groups in images. [Cinbis and Sclaroff, 2012] constructs classifiers that operate over sets of objects using object-object and object-scene relations to re-score and remove noisy detections. ScenarioNet differs from these methods because it jointly learns to group objects and coarsely localize them in images.

2.2 Explainable Models for Visual Recognition

The AI community has placed greater importance on learning explainable models and making complex models more interpretable. This is especially important for visual recognition tasks where many state-of-the-art models rely on deep neural networks. Several solutions have been proposed to solve this problem. [Ribeiro *et al.*, 2016] proposes a general framework for making complex models interpretable by looking at local linear approximations of the model's behaviour. Other works focus on generating visual explanations of CNN features, e.g., [Oquab *et al.*, 2015], [Zhou *et al.*, 2016], [Selvaraju *et al.*, 2017], and [Lengerich *et al.*, 2017], but these methods do not generate semantic explanations. [Hendricks *et al.*, 2016] trains a CNN to recognize objects and then trains an RNN to generate natural language explanations for the recognition decisions. They extend the system to work with visual question answering systems and also generate attention maps [Park *et al.*, 2016]. ScenarioNet generates less sophisticated, yet still human-interpretable semantic descriptions but doesn't require training language models which require large databases of image-caption pairs.

3 Proposed Method

3.1 Identifying Scenarios from Data: Pseudo-Boolean Matrix Factorization

We begin our discussion of the technical details of our model by asking: how do we identify which sets of objects naturally group together to form scenarios? We start with a training set of scene instances and a finite set of predetermined objects. We have ground-

truth annotations for the presence (or lack thereof) of every object in every scene instance given by either humans or object detectors. For each training instance, we create a vector of object presences where each element corresponds to a specific object, and the element is 1 if the object is present and 0 otherwise. We concatenate these vectors to form a matrix A where each row corresponds to a specific object and each column is a training instance. After specifying the number of desired scenarios k (which can be estimated from the data), we decompose A into two smaller approximately binary matrices: a dictionary matrix W representing a set of scenarios and an encoding matrix H that expresses scene instances as combinations of scenarios. Each column of W represents a single scenario and each row represents an object. If element W_{ij} is 0 or very small, object i is not present in scenario j. As W_{ij} approaches 1, object i exerts more influence on scenario j. Each column of H represents a specific scene instance and each row represents a specific scenario. If element H_{ij} is 0 or very small, then scenario i is not present in scene instance j. As H_{ij} approaches 1, scenario i exerts more influence on scene instance j.

Formulation of PBMF

We propose identifying scenarios using an approximation of Boolean matrix factorization (BMF) [Miettinen *et al.*, 2008]. In BMF, A, W, and H are binary matrices and the matrix multiplication is Boolean (denoted as $\circ$):

$$\min_{W,H} ||(A - W \circ H)||_1 \; s.t. \; W \in \{0,1\}, H \in \{0,1\} \tag{1}$$

BMF is well-suited for identifying scenarios from data because: 1) it efficiently compresses and preserves information using low-dimensional representations; 2) the basis vectors are easy to interpret; 3) it discovers meaningful interactions between objects; and 4) the encoding vectors are sparse, so each instance is expressed by a small subset of scenarios.

We use a gradient descent-based approach to solve the optimization problem. The formulation in Eq. 1 is not continuous, so we approximate Boolean matrix multiplication as $W \circ H \approx \min(WH, 1)$ and relax the constraints to lie in $[0, 1]$. Using $\min(WH, 1)$ results in cases where the gradient vanishes, so we further approximate $\min(WH, 1) \approx \min(WH, 1 + 0.01WH)$. Our basic **Pseudo-Boolean Matrix Factorization (PBMF)** formulation becomes:

$$\min_{W,H} ||(A - \min(WH, 1 + 0.01WH))||_F^2 \; s.t. \; W \in [0,1], H \in [0,1] \tag{2}$$

(Eq. 2) is still not perfectly suited for discovering scenarios. We add three additional terms: an orthogonality penalty to encourage diversity between scenarios and sparse penalties on the scenario dictionary and encoding to push W and H closer to binary matrices and improve interpretability. We introduce a weight matrix Ω that decreases the importance of common objects and increases the importance of rare objects during the factorization.

$$\begin{aligned} &\min_{W,H} ||\Omega \bullet (A - \min(WH, 1 + 0.01WH))||_F^2 \\ &+ \alpha_1 ||W^\intercal W - diag(W^\intercal W)||_F^2 + \alpha_2 ||W||_1 + \alpha_3 ||H||_1 \\ &s.t. \; W \in [0,1], H \in [0,1], \\ &\Omega_{ij} = \max\left(A_{ij} * \left(1 + \log\left(\frac{N_{instances}}{N_{objects}}\right)\right), 1\right) \end{aligned} \tag{3}$$

$\bullet$ denotes element-wise matrix multiplication. The αs represent tradeoff parameters.

3.2 ScenarioNet: Updating and Recognizing Scenarios from Visual Data

So far we've assumed we have perfect knowledge of all ground-truth object data. This means that if we're given a previously unseen scene instance, we can hold the scenario matrix constant and directly solve for the encoding matrix. In practice, we'll not have object data at test time. We need to learn how to recover the scenario encoding for a specific scene instance entirely from visual data. To do this, we integrate PBMF with CNNs. We propose **ScenarioNet**, a CNN that learns to identify and recognize scenarios from real-world visual data, performs scene classification using the predicted scenario encoding, and generates attention maps that highlight the regions the net focuses on when predicting whether a specific scenario is present in a given image. **ScenarioNet learns to predict an estimated scenario encoding matrix $\hat{H}$ and finetunes the dictionary W to adapt to the noisier $\hat{H}$. W also incorporates feedback from the scene classification task to improve discriminability**. The key architectural difference between ScenarioNet and other CNNs is the **scenario block** (see Fig. 1) which replaces the final fully connected layers used for classification in standard CNNs.

We now describe the rationale behind the scenario block. The final convolutional layers of a neural net such VGGNet are fed into a global average pooling (GAP) layer. This layer in combination with the class activation mapping technique [Zhou *et al.*, 2016] allows us to identify which parts of an image ScenarioNet attends to when determining if a scenario is present in the image. The output of the GAP layer is fed into a fully connected layer followed by a sigmoid transformation layer. The sigmoid layer outputs the **scenario encoding vector** and enforces each element of the vector is between 0 and 1. This vector tells us how present each scenario is in a given image. The scenario encoding layer feeds into a PBMF loss layer which finetunes the scenario dictionary and provides feedback to the network. The scenario encoding is also fed into a sequence of layers equivalent to a multinomial logistic regression model that uses scenarios as low-dimensional, interpretable features for scene classification.

Training ScenarioNet

During training, ScenarioNet only requires information about the *presence* (not location) of objects in an image. For scene classification, class labels are also needed during training. During testing, only images are given. First, the scenario dictionary is learned using ground-truth object presence data. Then, the net is trained to predict the scenario encodings while the dictionary is finetuned. Next, we train a softmax classifier for scene classification on top of a frozen net. Finally, we jointly finetune the net for scenario recognition and scene classification while once again finetuning the dictionary. It is useful to finetune only the last few layers of networks that have been previously trained for scene classification (e.g., on the Places dataset [Zhou *et al.*, 2017a]) since scenario recognition and scene classification are closely related. Each step of the finetuning process takes between 10 and 20 epoches. To finetune the dictionary while training the net, we use alternating projected gradient descent. During training, we hold the scenario dictionary constant and finetune the network using back-propagation in mini-batches to predict the encoding coefficients. After every four iterations, we hold the network constant and perform a full pass through the data to reconstruct $\hat{H}$ and finetune the scenario dictionary W using projected gradient descent. Alternatively, W can be efficiently finetuned using mini-batches by noting that the gradient of the PBMF loss w.r.t. W is able to be decomposed as a sum of gradients over sub-batches of $\hat{H}$; thus, we never have to compute the full $\hat{H}$ at any point in time.

Generating Evidence: Interpreting the Output of ScenarioNet

We now discuss how to interpret the output of ScenarioNet and by doing so understand how ScenarioNet makes interpretable decisions. Given an input image, ScenarioNet provides us with 1) a probabilistic scene class assignment, 2) a vector of scenario encoding coefficients, 3) the dictionary of scenarios, and 4) activation maps that can be used to localize the discriminative parts of each scenario. In Fig. 2, we show an example of decomposing a scene instance into its top-3 strongest detected scenarios using ScenarioNet. We see that ScenarioNet correctly predicts with high confidence that the scene category is "dining room". The top-3 scenarios support this: one focuses on dining areas, one on kitchen appliances, and one on decorative flowers. The encoding coefficient denotes the strength of each scenario. Note that all of the encoding coefficients are close to one since these are the strongest detected scenarios. As this coefficient decreases, the scenarios become less present. Encoding coefficients tend to cluster around 0 and 1. Recall that ScenarioNet uses scenarios as features for scene classification. We can define a scenario's *influence score* for a specific class to be the corresponding weight in the multinomial logistic regression model. If the influence is a large positive number, the scenario provides strong evidence for the specified class. If it is a large negative number, the scenario is strong evidence against a specific class. For this image, scenario 1 is very indicative of the scene class, while scenarios 2 and 3 are weakly indicative. We can also see how much influence each object exerts on each scenario. For example, in scenario 1, the "chandelier" and "chair" objects exert more influence when defining the scenario than the "buffet counter" object. By examining the scenario activation maps, we see that each predicted scenario is present and net attends to regions of the image containing objects present in the scenarios.

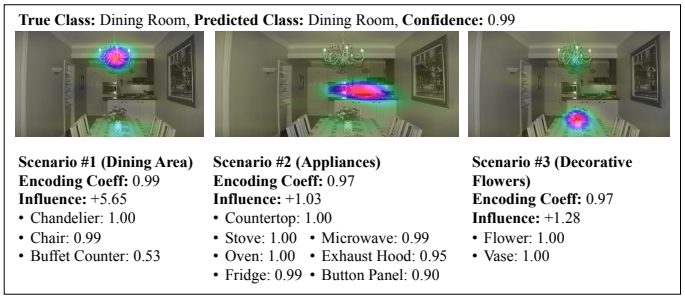

Figure 2: We demonstrate the explainability of ScenarioNet. We show the top-3 predicted scenarios for a dining room scene along with the corresponding activation maps. Please view in color.

Efficiency

Interestingly, our network has substantially fewer parameters than equivalent base architectures. For example, the final convolutional layers of VGG-16 typically consist of a 4096-by-4096 matrix followed by a 4096-by-#classes matrix for a total of $4096(4096 + \#classes)$ parameters. Our net uses a 512-by-#scenarios matrix followed by a #scenarios-by-#classes matrix for a total of $\#scenarios(512 + \#classes)$ parameters. Since #scenarios $<<$ 4096 (we use between $k = 25$ and $k = 70$ scenarios in our experiments), this results in over a 100x reduction in the number of parameters in the final layers, reduces the memory footprint of the *total* net by a factor of ˜10, and the net is ˜15% faster during testing.

4 Experimental Results and Analysis

In this section, we evaluate the reconstruction ability of PBMF, and also analyze the performance of ScenarioNet on three common scene understanding tasks: multi-object recognition, scene classification, and content-based scene image retrieval. We first explain the general experimental setup.

4.1 Experimental Setup

We conduct experiments on the SUNRGBD [Song *et al.*, 2015], ADE20K [Zhou *et al.*, 2017b], and MIT 67 Indoor Scenes [Quattoni and Torralba, 2009] datasets. We divide each dataset into separate training and test sets using the recommended splits for the SUNRGBD and MIT67 datasets and a random split for the ADE20K dataset. For each dataset, we only consider objects that appear in at least 1% of the training instances resulting in 55 objects for SUNRGBD, 193 for ADE20K, and 166 for MIT67. We use random cropping and horizontal mirroring to augment the training examples. For the SUNRGBD dataset, we use the 15 most frequently occurring scene classes, reserving 100 samples per class for test data, and generating 1000 samples per class for the training data. For the ADE20K dataset, we use the 31 most frequently occurring scene classes, reserving 25 samples per class for test data, and generating 500 samples per class for training data. For the MIT67 dataset, we use 67 scene classes, reserving 20 samples per class for test data, and generating 800 samples per class for training data. **We learn 25 scenarios for SUNRGBD, 70 for ADE20K, and 70 for MIT67. We use VGG-16 as our base CNN architecture**, replacing the final fully-connected layers with the scenario block. **For the MIT dataset, we only have object annotation data for about one-fifth of the training data, the amount of annotated data is very imbalanced between classes, and the annotations are much noisier than for the other datasets. These properties make learning scenarios on the MIT dataset much more difficult than for the other datasets, but we are still able to achieve relatively good results.** For this dataset, we learn the scenarios using the annotated portion of the training set and train a scene classifier on top of these scenarios for the full training set.

4.2 Reconstruction Error of PBMF

PBMF is a lossy factorization. We want to determine how much information about object presence is lost as a result of the decomposition. **For this experiment, we assume perfect, ground-truth knowledge of the object presences.** We consider three cases of PBMF: PBMF-Basic (Eq. 2), PBMF-Full (Eq. 3) with uniform weighting, and PBMF-Full using the proposed weight matrix. We compare to the SVD, NNSVD [Ding *et al.*, 2006], NMF [Paatero and Tapper, 1994], Greedy Boolean MF [Miettinen *et al.*, 2008], and Binary MF [Zhang *et al.*, 2007] as well as all-zeros and all-mean values baselines. We initialize the basis and encoding matrices using a procedure similar to [Zhang *et al.*, 2007]. Results are plotted in Fig. 3. PBMF-Basic works exceptionally well for reconstruction, generally losing to the much less constrained SVD. However, if we only focus on optimizing reconstruction error, we will overly prioritize common objects and might learn bases that lack diversity. As Fig. 3 demonstrates, adding orthogonality constraints and reweighing rare classes impacts the reconstruction error; however, we found adding these constraints results in dictionaries that are better suited for higher-level tasks such as scene classification and image retrieval.

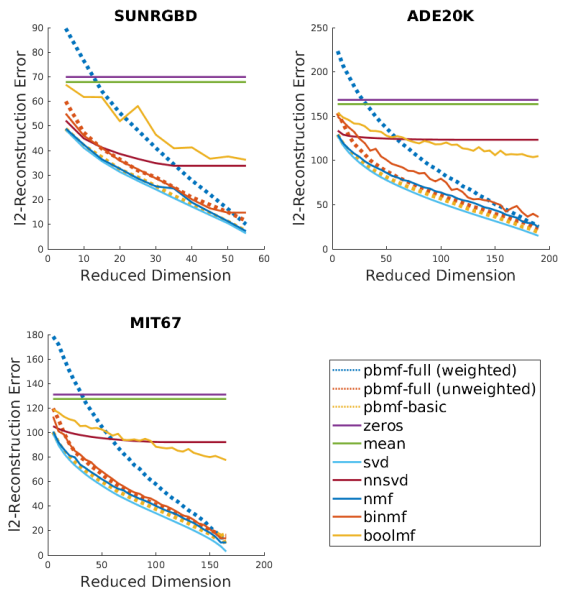

Figure 3: Reconstruction error between a recovered and groundtruth matrix as the dimensionality of the reduced representation is varied

4.3 Multi-Object Recognition from Scene Images

In addition to noise from the lossy PBMF, we also need to consider noise resulting from mapping visual data to objects. We consider the task of tagging images with their constituent objects. We use ScenarioNet to predict the scenario encoding matrix $\hat{H}$ and try to recover *an approximate hypothesis* about which objects are present in a scene by recovering the object-scene data matrix A using the learned scenario dictionary W and predicted encoding matrix $\hat{H}$: $A \approx W\hat{H}$. This recovery gives us a list of the *possible* objects in a scene. We compare against a finetuned object detector [Redmon and Farhadi, 2017] and VGG-16 finetuned for multi-object recognition. We use the macro (averaged) area under the precision-recall curve as our metric since the data is very imbalanced for rare objects. In Table 1, we show results for when we consider objects that appear in at least 1% of data (very rare) and 5% of data (rare) to show how imbalance affects both methods. The object detector works well when we have large, complete objects, but ADE20K tends to contain smaller objects and parts-of-objects, so the VGG-based nets outperform the object detector on this data, and perform worse on SUNRGBD. Likewise, there is greater labor cost associated with training the YOLO net because it requires bounding box information, which the other methods do not. VGG-Objects and ScenarioNet perform similarly despite PBMF being lossy and the output of ScenarioNet being 2-3 times smaller in dimensionality. Interestingly, ScenarioNet performs better than VGG-Objects on the SUNRGBD-5% task. We believe ScenarioNet performs well because it excels at capturing context and because it is easier to recognize scenarios (defined by a few key objects) than individual objects. ScenarioNet has several advantages over individual object-based methods: it finds relationships between objects and captures global scene information.

4.4 Scene Classification

We now consider the task of scene classification where we care more about global scene information than local objects. In the following sections, we compare ScenarioNet to other object-based representations, baseline CNNs, compressed CNNs, and other mid-level features. Results are reported in Table 2. For experiments not involving a CNN, we train a logistic regression model on top of the given features.

Method	SUNRGBD		ADE20K	
	1% (55 Objs)	5% (16 Objs)	1% (193 Objs)	5% (50 Objs)
Random	0.066	0.152	0.070	0.171
Object Detection (YOLOv2)	0.442	0.633	0.379	0.587
VGG-Objects	0.369	0.574	0.475	0.696
ScenarioNet	**0.356**	**0.585**	**0.452**	**0.683**

Table 1: Macro-AUPRC for multi-object recognition

Method	Dimens.	SUNRGBD	ADE20K	MIT
Object-Based Representations				
Object Bank + PCA	8000	0.296	0.511	0.39
Object Detection (YOLOv2)	55/193/166	0.399	0.639	0.517
VGG-Objects	55/193/166	0.483	0.726	0.6187
Baseline CNNs				
AlexNet	4096	0.469	0.786	0.687
GoogLeNet	2048	0.541	0.796	0.737
VGG-16	4096	0.531	0.809	0.792
ResNet-50	1024	0.509	0.777	0.687
Dimensionality-Reducing and Lower-Parameter CNNs				
VGG-Reduced	25/70/70	0.458	0.787	0.722
VGG-GAP	512	0.486	0.767	0.779
VGG-GMP	512	0.463	0.786	0.723
Attribute-Based Representations				
SUN-Attribute	102	0.429	0.705	0.655
Classemes	2659	0.309	0.581	0.448
Meta-Classes	15232	0.36	0.635	0.525
Learned Mid-Level Visual Representations				
Mid-Level Patches	14070	N/A	N/A	0.381*
Mid-Level Vis. Elem.	67000	N/A	N/A	0.64*
DPM	N/A	N/A	N/A	0.304*
RBoW	N/A	N/A	N/A	0.379*
BoP	3350	N/A	N/A	0.461*
Discriminative Parts	4926	N/A	N/A	0.514*
Proposed Model				
ScenarioNet	**25/70/70**	**0.520**	**0.794**	**0.725**

Table 2: Scene classification accuracy; * denotes reported results

Comparison to Other Object-Based Representations

We first consider object-based representations. These include the same models as in Sec. 4.3 (using the object probabilities as features) and also Object Bank features [Li *et al.*, 2010] compressed to 8000 dimensions using PCA. ScenarioNet is better than all other object-based representations for scene classification despite its lower dimensionality. This suggests that scenarios are better at capturing global scene information than individual object-based approaches. This is partly because ScenarioNet is trained to jointly recognize objects and scenes, a key difference to the other methods.

Comparison to Baseline CNNs

CNNs are currently a very popular method for scene classification. We finetune AlexNet [Krizhevsky *et al.*, 2012], GoogLeNet [Szegedy *et al.*, 2015], and VGG-16 [Simonyan and Zisserman, 2014] models that have been pre-trained on the Places dataset [Zhou *et al.*, 2017a] as well as a ResNet-50 CNN [He *et al.*, 2016] pre-trained on ImageNet [Deng *et al.*, 2009]. Since ScenarioNet extends VGG-16, we focus on how these two nets compare. ScenarioNet tends to slightly underperform VGG-16 by about 1-2%. The most significant drop in performance is on the MIT dataset, but ScenarioNet is forced to learn scenarios on a smaller, noisier, and imbalanced subset of the training data (see Sec. 4.1). ScenarioNet has several advantages over VGG-16; it has fewer pa-

rameters, has the ability to explain its decisions, and can produce scenario encodings which are useful for tasks beyond scene classification. In the next section, we see that ScenarioNet generally performs better than VGG-16 nets that compress the feature space to the same dimensionality as ScenarioNet.

Comparison to Dim-Reducing and Low-Param. CNNs
We modify VGG-16 so the output of the final feature layer is the same dimensionality as our scenario-based representation (by shrinking the FC layers). ScenarioNet matches or outperforms the compressed VGG-16 net in all cases. This might be because ScenarioNet constrains the intermediate representation to have high-level meaning while the compressed-VGG net lacks such guidance, making it susceptible to finding worse local minimum. We also compare ScenarioNet to VGG nets which replace the double fully-connected layers with global average pooling (GAP) and global max pooling (GMP) layers. These nets contain roughly the same number of parameters as ScenarioNet. In five of six cases, we outperform or match the low-parameter nets.

Comparison to Mid-Level Representations
Finally, we compare against three other types of mid-level representations: attributes, mid-level visual patches, and parts-based models. Attributes are high-level semantic properties shared between multiple classes [Farhadi *et al.*, 2009]. We consider three attribute-like representations: SUN Attributes [Patterson and Hays, 2012], Classemes [Torresani *et al.*, 2010], and Meta-Classes [Bergamo and Torresani, 2012]. Several representations consider visually-distinct, meaningful mid-level patches [Singh *et al.*, 2012] and mid-level visual elements [Doersch *et al.*, 2013]. Finally, we consider parts-based models including the deformable parts model (DPM) [Pandey and Lazebnik, 2011], reconfigurable bags-of-words (RBoW) [Parizi *et al.*, 2012], bags-of-parts (BoP) [Juneja *et al.*, 2013], and discriminative parts [Sun and Ponce, 2013]. We outperform all of these methods, but it should be noted that for the non-attribute-based features, we use the reported results on the MIT dataset because the code to generate these features is either unavailable or prohibitively expensive to run on our machines. It should also be noted that these methods pre-date CNNs, and not all of the reported results include the use of training data augmentation while ScenarioNet does.

4.5 Content-Based Querying and Comparison

ScenarioNet is useful for content-based scene image retrieval because it can retrieve images satisfying a set of high-level criteria based on the scene category, scenarios, and objects present in an image (e.g., find images of scene category A OR B THAT CONTAIN scenarios X AND Y but EXCLUDE object Z). Often, we want to query for broad concepts and not individual objects. Scenarios offer a nice compromise between global (scene category) and local (object) information. It is easy for humans to examine the scenario dictionary and form complex queries because scenarios are low-dimensional and interpretable. Scenarios can also act as an efficient hashing mechanism because they are low-dimensional and approximately binary, so memory requirements are low and retrieval can be performed in an efficient manner.

In Table 3, we evaluate ScenarioNet for complex content-based scene image retrieval. We form 500 random queries, each consisting of a desired scene class, two objects that should be present, and one object that should be absent but frequently co-occurs with the other two objects, i.e. $(SC \cap O_1 \cap O_2 \cap \neg O_3)$. We do not consider querying against scenarios for this task because no other method is capable of recognizing scenarios. We measure the relevance of a returned image as the proportion of query terms that are satisifed. We compute the normalized discounted cumulative gain for the top-5 result images for each query. ScenarioNet is very competive with the other methods, matching VGG-Objects for the best performance on ADE20K, and coming very close to both baselines on SUNRGBD.

Method	SUNRGBD	ADE20K
Random	0.302	0.313
Object Detection (YOLOv2)	0.679	0.760
VGG-Objects	0.686	0.799
ScenarioNet	**0.652**	**0.799**

Table 3: NDCG@5 for retrieving images of a given class containing 2 specific objects and not containing a third highly-correlated object

Query: How are the Images Similar?
Result:
-Both images are "park" scenes.
-They share the following scenarios: (person) and (grass, path, earth, tree)

Query: How do the Images Differ?
Result:
-Image 1 contains the scenario (head, arms, legs, bag)
-Image 2 contains the scenario (building, lamp, road, sidewalk, window)

Figure 4: Using ScenarioNet to find high-level similarities and differences between two images.

ScenarioNet is also useful for generating a quick overview of the similarities and differences between two scene images without relying on (often unnecessary) information about individual objects. Fig. 4 shows an example.

5 Conclusions

We introduced scenarios as a new way of representing scenes. The scenario is a simple data-driven representation based on sets of frequently co-occurring objects. We provided a method for learning scenarios from data by combining PBMF with CNNs to form the ScenarioNet. Our experiments showed that a single ScenarioNet model can perform scene classification, scenario recognition, multi-object recognition, content-based scene image retrieval, and content-based image comparison with performance comparable to or better than existing models. We showed that scenarios have several advantages over individual object-based representations; specifically, they are lower-dimensional, capture global scene context, and find relationships between objects. We also discussed and demonstrated the computational efficiency and interpretability of ScenarioNet compared to traditional CNNs. We believe ScenarioNet provides a strong first step towards constructing explainable and trustworthy models for safety-critical applications related to scene understanding (e.g., robotics, human-machine teaming, surveillance, and self-driving cars). However, much work remains, including evaluating the utility of scenarios in human studies and figuring out how ScenarioNet can be used in dynamic and interactive settings.

Acknowledgments

This work is partly supported by the Air Force Office of Scientific Research (AFOSR) under the Dynamic Data-Driven Application Systems (DDDAS) program. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1433187.

References

[Bergamo and Torresani, 2012] Alessandro Bergamo and Lorenzo Torresani. Meta-class features for large-scale object categorization on a budget. In *CVPR*, 2012.

[Choi *et al.*, 2012] Myung Jin Choi, Antonio Torralba, and Alan S Willsky. A tree-based context model for object recognition. *IEEE TPAMI*, 2012.

[Cinbis and Sclaroff, 2012] Ramazan Cinbis and Stan Sclaroff. Contextual object detection using set-based classification. *ECCV*, 2012.

[Deng *et al.*, 2009] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*. IEEE, 2009.

[Ding *et al.*, 2006] Chris Ding, Tao Li, Wei Peng, and Haesun Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *SIGKDD*, 2006.

[Doersch *et al.*, 2013] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Mid-level visual element discovery as discriminative mode seeking. In *NIPS*, 2013.

[Fan *et al.*, 2008] Jianping Fan, Yuli Gao, and Hangzai Luo. Integrating concept ontology and multitask learning to achieve more effective classifier training for multilevel image annotation. *IEEE TIP*, 2008.

[Farhadi *et al.*, 2009] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In *CVPR*, 2009.

[Feng and Bhanu, 2016] Linan Feng and Bir Bhanu. Semantic concept co-occurrence patterns for image annotation and retrieval. *IEEE TPAMI*, 2016.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[Hendricks *et al.*, 2016] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In *ECCV*, 2016.

[Johnson *et al.*, 2016] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. *arXiv*, 2016.

[Juneja *et al.*, 2013] Mayank Juneja, Andrea Vedaldi, CV Jawahar, and Andrew Zisserman. Blocks that shout: Distinctive parts for scene classification. In *CVPR*, 2013.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[Lan *et al.*, 2013] Tian Lan, Michalis Raptis, Leonid Sigal, and Greg Mori. From subcategories to visual composites. In *ICCV*, 2013.

[Lengerich *et al.*, 2017] Benjamin J Lengerich, Sandeep Konam, Eric P Xing, Stephanie Rosenthal, and Manuela Veloso. Visual explanations for convolutional neural networks via input resampling. *ICML Workshop on Visualization in Deep Learning*, 2017.

[Li *et al.*, 2010] Li-Jia Li, Hao Su, Li Fei-Fei, and Eric P Xing. Object bank: A high-level image representation for scene classification. In *NIPS*, 2010.

[Li *et al.*, 2012] Congcong Li, Devi Parikh, and Tsuhan Chen. Automatic discovery of groups of objects for scene understanding. In *CVPR*, 2012.

[Miettinen *et al.*, 2008] Pauli Miettinen, Taneli Mielikäinen, Aristides Gionis, Gautam Das, and Heikki Mannila. The discrete basis problem. *IEEE TKDE*, 2008.

[Oquab *et al.*, 2015] Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Is object localization for free? In *CVPR*, 2015.

[Paatero and Tapper, 1994] Pentti Paatero and Unto Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 1994.

[Pandey and Lazebnik, 2011] Megha Pandey and Svetlana Lazebnik. Scene recognition and weakly supervised object localization with deformable part-based models. In *ICCV*, 2011.

[Parizi *et al.*, 2012] Sobhan Naderi Parizi, John G Oberlin, and Pedro F Felzenszwalb. Reconfigurable models for scene recognition. In *CVPR*, 2012.

[Park *et al.*, 2016] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. Attentive explanations. *arXiv*, 2016.

[Patterson and Hays, 2012] Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*, 2012.

[Quattoni and Torralba, 2009] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *CVPR*, 2009.

[Rabinovich *et al.*, 2007] Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. Objects in context. In *ICCV*, 2007.

[Redmon and Farhadi, 2017] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. *CVPR*, 2017.

[Ribeiro *et al.*, 2016] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you? In *SIGKDD*, 2016.

[Selvaraju *et al.*, 2017] Ramprasaath Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *ICCV*, 2017.

[Simonyan and Zisserman, 2014] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*, 2014.

[Singh *et al.*, 2012] Saurabh Singh, Abhinav Gupta, and Alexei A Efros. Unsupervised discovery of mid-level discriminative patches. In *ECCV*. 2012.

[Song *et al.*, 2015] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*, 2015.

[Sun and Ponce, 2013] Jian Sun and Jean Ponce. Learning discriminative part detectors for image classification and cosegmentation. In *ICCV*, 2013.

[Szegedy *et al.*, 2015] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[Torresani *et al.*, 2010] Lorenzo Torresani, Martin Szummer, and Andrew Fitzgibbon. Efficient object category recognition using classemes. *ECCV*, 2010.

[Zhang *et al.*, 2007] Zhongyuan Zhang, Tao Li, Chris Ding, and Xiangsun Zhang. Binary matrix factorization with applications. In *ICDM*, 2007.

[Zhou *et al.*, 2016] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016.

[Zhou *et al.*, 2017a] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE TPAMI*, 2017.

[Zhou *et al.*, 2017b] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017.