NOD-CC: A Hybrid CBR-CNN Architecture for Novel Object Discovery

JT Turner[1], Michael W. Floyd[2], Kalyan Gupta[2], and Tim Oates[1]

[1] University of Maryland Baltimore County, Baltimore, MD, USA
jturner1@umbc.edu, oates@umbc.edu
[2] Knexus Research Corporation, National Harbor, MD, USA
first.last@knexusresearch.com

Abstract. Deep Learning methods have shown a rapid increase in popularity due to their state-of-the-art performance on many machine learning tasks. However, these methods often rely on extremely large datasets to accurately train the underlying machine learning models. For supervised learning techniques, the human effort required to acquire, encode, and label a sufficiently large dataset may add such a high cost that deploying the algorithms is infeasible. Even if a sufficient workforce exists to create such a dataset, the human annotators may differ in the quality, consistency, and level of granularity of their labels. Any impact this has on the overall dataset quality will ultimately impact the potential performance of an algorithm trained on it. This paper partially addresses this issue by providing an approach, called **NOD-CC**, for discovering novel object types in images using a combination of Convolutional Neural Networks (CNNs) and Case-Based Reasoning (CBR). The CNN component labels instances of known object types while deferring to the CBR component to identify and label novel, or poorly understood, object types. Thus, our approach leverages the state-of-the-art performance of CNNs in situations where sufficient high-quality training data exists, while minimizing its limitations in data-poor situations. We empirically evaluate our approach on a popular computer vision dataset and show significant improvements to objects classification performance when full knowledge of potential class labels is not known in advance.

Keywords: Deep Learning · Novel Object Discovery · Computer Vision · Convolutional Neural Networks.

1 Introduction

Deep Learning has seen rapid advancement in recent years, setting benchmarks for many machine learning tasks in the areas of computer vision, natural language processing, and game AI. While these deep neural networks are fundamentally the same as the perceptrons [14] of the late 1960's, they leverage dramatic improvements in the availability of computational resources and training data to significantly outperform their predecessors. In particular, the field of computer vision has benefited from the application of *Convolutional Neural Networks* (CNNs) [6] that are able to use massive image datasets to learn relevant

image features rather than relying on hand-engineered feature sets. Additionally, this field has been able to utilize a seemingly endless streams of crowdsourced labeled images from sources like Facebook, Instagram, Twitter, and Reddit.

However, the ability of these deep learning architectures to learn is directly tied to the availability of high-quality, human-labeled data to use during training [16]. If training data is either rare or low-quality, deep learning systems will have difficulty accurately learning from the data. In the case of rare data, it may be possible to gather more data over time as more images become available (e.g., as a new model mobile phone is released, when a new species is discovered and documented). The more difficult long-term problem is the quality of data, as demonstrated by the age-old idiom *"garbage in, garbage out"*. In some situations, this can be erroneous labels being given to training instances. For example, if an annotator labels an image of a *car* as a *tree*, the learning system will attempt to learn based on that erroneous data. Similarly, an annotator may only label a subset of objects in complex scenes. In a bedroom scene, the annotator may label *bedclothes*, *pillows*, and *nighstands*, but treat other objects as background, like *alarm clocks* or *lamps*. Furthermore, the system's learning will be constrained by the level of granularity used during labeling and the annotator's term preference. For example, the choice of whether to use a high-level label such as *animal* or *pet*, limit the classification granularity of a vision system compared to using lower-level labels such as *cat*, *European cat*, or *Russian Blue cat*. These issues are compounded by the fact that, given the scale of datasets used by Deep Learning systems, it is impractical for a single annotator to label an entire dataset. Instead, the annotation work is generally crowdsourced from hundreds or thousands of human annotators. It is unlikely that all of these annotators will be consistent, error-free, and complete in the labels they provide. Thus, the overall quality of the labeled datasets, and ultimately the potential performance of a machine learning system trained on the datasets, is bound by the quality of the human annotators.

We propose a method, called *Novel Object Discovery Using Convolutional Neural Networks and Case-Based Reasoning (NOD-CC)*, for object discovery and classification in images that leverages the high-end performance of CNNs while reducing its reliance on large sources of pre-labeled training data. Instead, NOD-CC attempts to classify an input image using a trained CNN, but can dynamically switch to using a case-based classification approach if the CNN isn't confident in its prediction. The primary motivation of this approach is that while CNNs require a large collection of training images of each object type to learn successfully, a CBR can be used to learn using as few as one training instance. Thus, the CBR component can be used to discover novel object types and provide classification of those types until such time as there are sufficient training examples to retrain the CNN.

In our previous work [23], we demonstrated how CBR can leverage the automated feature extraction capabilities of CNNs, and perform novel object discovery and classification. In that work, which we will refer to as *Novel Object Discovery using Case-Based Reasoning (NOD-CBR)*, the convolutional layers of

a CNN (i.e., the CNN architecture excluding the fully-connected neural network layers) are used to convert input images into a feature vector representation. That feature vector representation, and optionally any detectable *object parts* that are visible, is used to retrieve similar cases and determine if an object of that type has been encountered previously. NOD-CC significantly extends NOD-CBR and provides the following key contributions:

- A hybrid architecture that includes both the NOD-CBR system as well as a fully functional CNN (i.e., a CNN that performs object classification rather than purely feature extraction).
- An architecture that provides both the high-end performance of CNNs as well as the lazy, data-poor learning capabilities of CBR.
- A series of decision algorithms that can dynamically select whether to use the CNN or CBR components of our architecture to perform object classification.
- An online method for object classification, novel object discovery, novel object labeling, and learning.
- An empirical evaluation that demonstrates the utility of NOD-CC when the full set of object types is not known in advance.

The remainder of this paper describes how NOD-CC combines Convolutional Neural Networks and Case-Based Reasoning to classify images while also performing only novel object discovery. Section 2 provides an overview of similar research in both deep learning and CBR. Section 3 describes our hybrid architecture that combines CNNs and CBR for object classification and discovery. Our empirical evaluation is presented in Section 4, and provides evidence to support our claims of the utility of NOD-CC. Finally, in Section 5 we summarize our findings and identify important future research directions.

2 Related Work

The intersection of CBR and CNNs can often be seen in the domain of signal analysis known as Human Activity Recognition (HAR), or in medical settings. In many HAR settings, usage of high-fidelity wearable sensors for movement are used for feature extraction [20], and to further train classifiers for new users [19]. Using multi-channel medical device EEG signals, researchers have also conducted analysis on patterns of electrical signals from the brain to set a baseline for seizure detection [24], and then using a case base of seizure-like data to classify unseen patients [15]. The usage of actigraphy sensors provided a plethora of medical data regarding sleep patterns [21] to attempt to predict sleep quality and activity using Deep Learning techniques such as multilayer perceptrons, convolutional neural networks, and many variants of recurrent neural network. By leveraging this high quality sensor data, it is possible to preserve existing patient privacy of medical information, while training a model to be useful as a starting point, or fine tuning point for new or changed data [18].

In conjunction with image processing, CBR has seen a wide array of uses [17], once again with a great deal of research in the medical domain. Despite the wide

usage and success of CBR in a variety of medical domains (e.g., [12] [10] [7]), most applications require hand-crafted features (e.g, [1] [13] [11]) generated by Subject Matter Experts (SMEs), a practice which does not scale to the Exascale-level computation and learning that Deep Learning making possible [8].

As similar application of CBR to our novel object detection system is a website classifier on sites by using image data from the websites instead of the textual data [11]. Although this work also used the feature vector from latter stages of a convolutional neural network, this was used to classify the images form the website into existing categories without leaving the possibility for novelty.

Also different from other previous works is that our system novel object detection system performs upsupervised learning in an online, incremental manner, not doing offline dataset analysis to search for out of distribution classes.

3 Hybrid CNN-CBR Architecture

Our approach, Novel Object Discovery Using Convolutional Neural Networks and Case-Based Reasoning (NOD-CC), is a hybrid of two learning and classification methods (Figure 1). The Convolutional Neural Network component (labeled as CNN) is intended to classify images of object types for which sufficient training instances are available. Additionally, it converts raw images into feature vectors for use by the Case-Based Reasoning component (labeled as CBR). The CBR component is intended to learn from and classify object types that are not classifiable by the CNN. A meta-algorithm, labeled as Controller, determines whether the classification from the CNN or CBR component is used to provide final image classification. In the following sub-sections, we will provide details about each of the three primary components: CNN, CBR, and Controller.

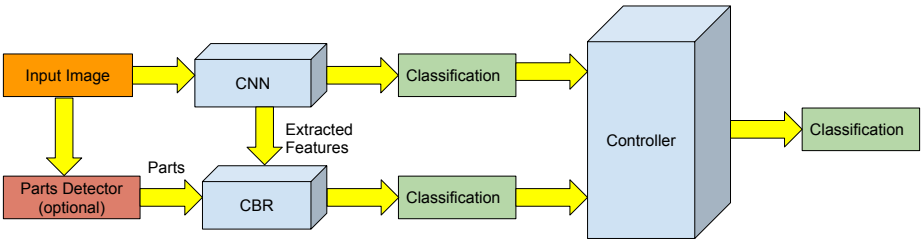

Fig. 1. Architecture of the NOD-CC image classification system. The classifications are shown in green and are produced by the three decision algorithms shown in blue. The inputs to the decision algorithms are shown in yellow, the input image in orange, and the optional parts detector in red.

3.1 Convolutional Neural Network Component

When used for object classification, a Convolutional Neural Network takes as input an image and outputs a classification for what type of object is visible in the input. CNNs contain two primary stages: *convolutional and pooling layers*, and *fully-connected layers*. The convolutional and pooling layers take the raw image input and extract a feature vector containing the relevant features that were learned during training. For example, these features may include the presence (or absence) of certain edges, shapes, or complex geometric objects (composed of many shapes). This feature vector is used as input to the fully-connected layers, which then use the feature vector to determine a classification for the image. In NOD-CC, the feature vector computed by the convolutional and pooling layers is also provided to the CBR component. This is done to leverage the ability of CNNs to automatically learn and perform feature extraction, and avoids any manual feature engineering for the CBR component.

NOD-CC is agnostic to the particular CNN architecture used; since all CNNs can produce an intermediate feature vector (which can be provided to the CBR component) and a classification, any architecture can be used. In our work, we use the Inception-v3 architecture [22]. We selected this architecture based on it having been shown to achieve similar classification performance compared to more computationally expensive architectures such as ResNet [5] and ResNeXt [25]. An additional benefit of using the Inception-v3 architecture is that it allows the possibility of future extensions of NOD-CC, as part of future work, to use a hierarchical image grammar. This would allow not only novel object discovery but also hierarchical class relationships between classes (e.g., that a novel object type is a subclass of an existing object type). Inception-v3 facilitates this by using a set of auxiliary classifiers, used to combat the vanishing gradient problem during training, that could be used to facilitate predictions at multiple levels of granularity.

3.2 Case-Based Reasoning Component

Although CNNs can achieve high accuracy when classifying objects in images, their performance is dependent on the set of class labels (i.e., object types) contained in the training data. If the training data contains images labeled with the set of labels $\mathcal{L} = \{l_1, \ldots, l_n\}$, a CNN (and most other learning algorithms) will only be able to classify those n object types. Any images of objects with a label l_m (where $l_m \notin \mathcal{L}$) will either be misclassified as one of the labels in $\mathcal{L}$ or unclassified (i.e., the CNN will output a low confidence for all labels such that an *unknown* output is produced). This issue is particularly problematic for CNNs since they require a large set of example images labeled as l_m before they be accurately trained to predict that object type. CBR, on the other hand, likely does not have the same peak classification performance on massive image datasets but is capable of one-shot learning. Once a single image with label l_m is encountered, it can be stored as a case and reused to classify other instances of that object type.

For the CBR component of NOD-CC, we use our previous case-based novel object discovery approach, NOD-CBR [23]. NOD-CBR stores each training image $I_i \in \mathcal{I}$ (where $\mathcal{I}$ is the set of all images) as a case C_i in the case base CB ($C_i \in CB$). Cases are encoded as triplets containing the feature vector representation of the image F_i, a set of detectable image parts P_i, and the ground truth object label l_i: $C_i = \langle F_i, P_i, l_i \rangle$. Using case-based reasoning nomenclature, the feature vector and parts set of the image are the *problem*, and the class label is the *solution*.

Recall from the previous subsection that the convolutional and pooling layers of the CNN component convert a raw input image into a feature vector $F_i = \langle f_1, \ldots, f_v \rangle \in \mathcal{F}$ (where v is an integer value defined by the CNN architecture and $\mathcal{F}$ is the set of all feature vectors). Thus, both the CBR component and the fully-connected layers of the CNN component use an identical feature vector representation as produced by the convolutional and pooling layers mapping from images to features: $features : \mathcal{I} \rightarrow \mathcal{F}$.

Each case also contains the set of parts $P_i \subset \mathcal{P}$ that are detectable in the input image, where $\mathcal{P}$ is the set of all parts that may be detected. There parts are generic lower-level structures of an image, like *hands*, *feet*, *wheels*, or *wings*. Since parts are generic, different objects types can share parts (e.g., both *dogs* and *cats* have *legs*, *heads*, *ears*, *tails*). However, even images of the same object type may have different detectable parts based on variations in pose, occlusion, or photographic style. For example, in Figure 2, the cats do not have an identical set of detectable parts due to different poses and image framing. Our work assumes the presence of a parts extractor that returns the set of detected parts in an image: $parts : \mathcal{I} \rightarrow \mathcal{P}$. However, as we will discuss shortly, while our approach can leverage parts information it is not necessary for classification (i.e., it can classify using only the feature vector).

The NOD-CBR object discovery and classification algorithm is shown in Algorithm 1. While full details of the algorithm are described in our previous work [23], we will provide a brief overview of its reasoning process. Given an input image, the algorithm will extract the feature vector representation (i.e., from the CNN component) and the set of detectable parts (i.e., from the parts extractor). If the either the case base is empty (Lines 4-5), no cases are sufficiently similar to the input image's feature vector representation (Lines 7-9, based on a threshold λ_f), or there are cases with similar feature vectors but their detectable parts are not similar (Lines 11-17, based on a threshold λ_p), then NOD-CBR generates a new label for the input image. In that situation, it believes the image to be of a newly discovered object type. Otherwise (Line 19), it uses the class label from the the most similar retrieved case. In all situations, a new case is retained and added to the case base (Line 20).

An advantage of this approach is that it can start from a variety of initial case base configurations: an empty initial case base if no prior knowledge exists, a case base containing cases for all images used to train the CNN, or a sampling of cases of each object type if the full training set is too large. It should also be noted that while the *generateLabel*() function in Algorithm 1 will generate a unique

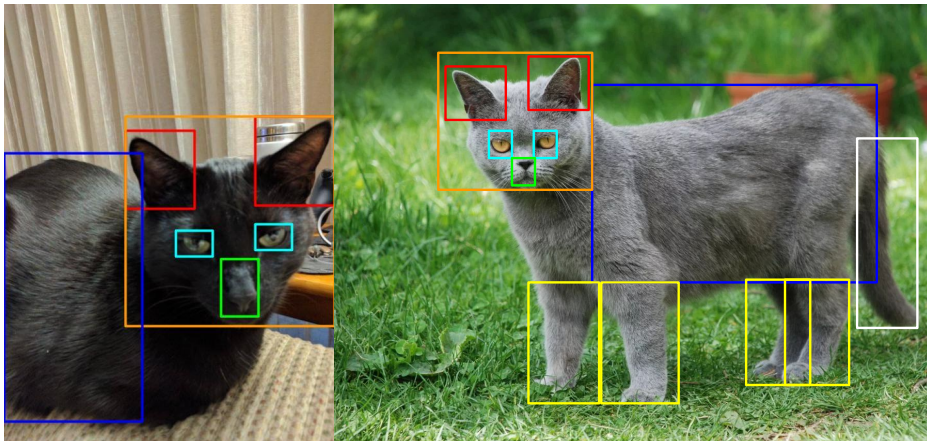

Fig. 2. The variation in pose of the two cats, as well as the framing of the picture can drastically effect the observable parts. The cat on the left in the so-called *catloaf* position is hiding his legs under his torso, and the way the picture is framed does not show its tail, while the cat on the right has all major parts visible.

label for a newly discovered object type, it will likely not be a meaningful class label (e.g., returning the label *object5849* rather than *lion*). However, images with newly generated labels (i.e., the newly discovered object types) could be presented to a human expert, either online or offline, to receive more meaningful object labels.

3.3 Controller Component

The CNN component and CBR component both output a classification for the input image. However, there is no guarantee that they will predict the same object type. The role of the controller is to receive as input the predictions from both components and output a final predicted class label.

In our work, we use three different Controller strategies:

- **Always CNN:** The classification output by the CNN component is used regardless of the the CBR component's classification. This is equivalent to the CNN component operating in isolation.
- **Always CBR:** The classification output by the CBR component is used regardless of the the CNN component's classification. This is equivalent to the CBR component operating in isolation.
- **Conditional CBR:** The classification of the CNN component is used unless the CNN has low confidence in its prediction. This occurs when none of the class labels are above an abstention threshold λ_a. In situations where the CNN does not output a class label, the prediction of the CBR component is used.

Algorithm 1: NOD-CBR algorithm for image classification and novel object discovery

Function: *classify*$(I_{in}, CB, k, \lambda_f, \lambda_p)$ ***returns*** l_{in}

```
1  F_in ← features(I_in);
2  P_in ← parts(I_in);
3  l_in = ∅;
4  if CB = ∅ then
5      l_in ← generateLabel();
6  else
7      topK ← retrieveTopK(F_in, CB, k, λ_f);
8      if topK = ∅ then
9          l_in ← generateLabel();
10     else
11         nn = ∅; nnSim = −1;
12         foreach C_i ∈ topK do
13             sim ← partSim(P_in, C_i.P_i);
14             if sim > nnSim and sim > λ_p then
15                 nn = C_i; nnSim = sim;
16         if nn = ∅ then
17             l_in ← generateLabel();
18         else
19             l_in ← nn.l_i;
20 CB ← CB ∪ ⟨F_in, P_in, l_in⟩ ;
21 return l_in;
```

4 Evaluation

Our empirical evaluation demonstrates the image discovery and classification performance of NOD-CC when the complete set of object types that will be encountered at run-time is not known in advance. More specifically, the following hypotheses are evaluated:

H1: The CNN component will be unable to correctly classify any object types not present in the training set.
H2: The CBR component, NOD-CBR, will outperform the CNN component when the training images do not contain instances of all object types that may be encountered at run-time.
H3: NOD-CC will achieve higher classification performance than the CNN component alone when the training images do not contain instances of all object types that may be encountered at run-time.
H4: NOD-CC will achieve higher classification performance than NOD-CBR alone when the training images do not contain instances of all object types that may be encountered at run-time.

4.1 Dataset

The image dataset used during our evaluation is the publicly available *PASCAL-Part* dataset [2], a subset of the images from the *Visual Object Classes Challenge 2010* dataset [3]. The dataset contains images with 20 coarse-grained ground truth object types: *aeroplane*, *bicycle*, *bird*, *boat*, *bottle*, *bus*, *car*, *cat*, *chair*, *cow*, *dining table*, *dog*, *horse*, *motor bike*, *person*, *potted plant*, *sheep*, *sofa*, *train*, and *tv monitor*. Additionally, each image has between 0 and 24 annotated detected parts. However, as discussed previously, images may differ on the number of annotated parts based on the quality and properties of each image.

Other properties of the *PASCAL-Part* dataset that make it appropriate for this evaluation include the variation between scale, orientation, pose, lighting, and ambient setting of the objects. The images include many complex, real-world environments so there is a high-degree of image clutter, object occlusion, and background scenes. For example, one image of a person is in a forested environment where less than 10% of the visible pixels are of the person, while in another a person takes up 90% of the visible pixels but is partially occluded by the water they are swimming in.

Our current work is focused on classifying a single object type in each image. To facilitate this, we filtered the *PASCAL-Part* dataset to only the images that contain a single class label, thereby reducing the dataset from $10,103$ images to $4,737$. While this may seem like a limitation of our approach, many computer vision applications first propose sub-regions of a cluttered image to classify (e.g., the *region proposal* stage of a Region-Based Convolutional Neural Network [4]), and then provide at most a single object label for each sub-region (i.e., a traditional CNN classification). Additionally, even though each image only contains a single labeled object, nearly all of the images contain a variety of unlabeled background objects. Since the size of the filtered *PASCAL-Part* dataset is quite small by Deep Learning standards, the CNN component used in our work, the Inception-v3 architecture, was pretrained on the much larger *Open Images v4* dataset [9] and then fine-tuned using the filtered *PASCAL-Part* dataset. It should be noted that there is no overlap between the images contained in the two datasets (i.e., pretraining on *Open Images v4* will not provide any images from the testing sets we use).

For our experiments, we used the filtered *PASCAL-Part* dataset to create 20 experimental datasets. The original dataset comes pre-partitioned into training and testing sets. For each of the 20 experimental datasets, 5 of the 20 object types were selected at random (such that no two experimental datasets used the same set of 5 object types). All images of the 5 selected object types were removed from the training set but left in the testing set. Thus, all testing sets contain images of all 20 object types, but the training sets only contained images of 15 object types. These experimental datasets were partitioned in advance, such that all experimental variations would work on an identical set of datasets.

4.2 Scoring Metrics

Our previous work [23] demonstrated the ability of NOD-CBR, when starting from an empty case base, to discover and classify classes. More specifically, we evaluated its ability to maximize class purity (i.e., provide the same generated label to images of the same object type) while minimizing the divergence in the number of discovered classes from the true number of classes (i.e., not over-partitioning the data). Given that we have previously demonstrated the efficacy of NOD-CBR on these tasks, our evaluation will measure the performance of our hybrid NOD-CC architecture's classification performance when class labels from the testing set are not present in the training set (i.e., novel object types are encountered at run-time).

For each testing image provided to NOD-CC, there are four possible ways in which the classification prediction of NOD-CC can align with the image's ground truth label, ordered from best to worst:

1. **Correct:** The class label predicted by NOD-CC matches the ground truth class label. This is the ideal situation and is considered to be a 100% match. C represents the percentage of training instances labeled correctly.
2. **Known Novel:** NOD-CC correctly predicts that the class label was not one of the class labels in its training set. Since a random guess would correctly predict a novel class 25% of the time (since 5 of 20 classes are not in the training set), we consider this to be a 25% match. KN represents the percentage of training instances labeled as known novel.
3. **Abstention:** NOD-CC does not have enough confidence in any of its potential predications, so it abstains from making a prediction. Since guessing a class label randomly would provide the correct prediction approximately 5% of the time (since there are 20 classes), we consider an abstention to be a 5% match. Essentially, this prevents NOD-CC from being forced to provide a random guess to boost its accuracy and allows it to abstain when it is unsure. A represents the percentage of training instances that were abstained from labeling.
4. **Incorrect:** NOD-CC predicts a known class label (i.e., a class label present in the training set) but it does not match the ground truth class label. This is incorrect and considered to be a 0% match. I represents the percentage of training instances labeled incorrectly.

During each evaluation, each image in the testing dataset is used as input to NOD-CC and a comparison between the predicted class and ground truth class label is used to calculate our scoring metrics: accuracy (ρ_A), precision (ρ_P), recall (ρ_R), and F1 score (F_1). Although the precision, recall, and F1 score calculations use well-established equations, we use a modified accuracy function based on the previous discussions of the four ways NOD-CC's classification can align with the ground truth classification.

$$\rho_A = \frac{(C + .25 \times KN + .05 \times A) \times TP}{TP + FN} \qquad F_1 = 2\frac{\rho_P \times \rho_R}{\rho_P + \rho_R}$$

$$\rho_P = \frac{TP}{TP + FP} \qquad \rho_R = \frac{TP}{TP + FN}$$

For every class label in the dataset (all training classes unseen at training time are considered to be of a single class labeled as*Novel Class*), we compute the accuracy (ρ_A), precision (ρ_P), recall (ρ_R), and f-score (F_1). For each experimental run (i.e., providing the testing instances from a single experimental dataset to Algorithm 1) the mean of each of the class-level metrics is computed. We further vary our experiments by randomizing the order in which testing instances are provided to Algorithm 1. This is important since it is a learning algorithm (i.e., new cases are stored) so the order of testing instances may impact performance. For each of the 20 experimental datasets, 20 random orderings were used. This resulted in 400 total experimental runs (20 datasets $\times$ 20 orderings) and the reported results are the averages of the metrics over all 400 runs.

4.3 Always CNN Variant

As a baseline, we evaluated the **Always CNN** variant of NOD-CC (i.e., when the CBR component is ignored). The abstention parameter λ_a was determined through cross-validation on the entire dataset, such that the F_1 was maximized. Recall that the **Always CNN** variant is unable to learn online; it is only able to abstain from providing a label. Assuming a perfectly balanced set of classes, since the CNN is only trained on 15 classes with the remainder only appearing in the testing set, its maximum accuracy is bounded as: $max(\rho_A) = (\frac{15}{20} \times 100\%) + (\frac{5}{20} \times 5\%) = 76.3\%$. In reality, due to the imbalance of the datasets the true maximum accuracy was lower - 63.9% in our experiments. We report an additional metric, *Relative Mean Accuracy (RMA)*, that measures the fraction of $max(\rho_A)$ that was achieved. We also report the minimum (Min. ρ_A), maximum (Max. ρ_A), median (Med. ρ_A) and standard deviation (σ ρ_A) of the accuracy (i.e., when examining each experimental run individually). The performance of **Always CNN** is shown in Table 1.

One item of note in these baseline results is that the precision is significantly higher than the recall. This is intuitive in a system that uses a threshold to determine confidence in classifications (i.e., λ_a); the system only provides classifications when it is confident in its predictions and thereby lowers the number of false positives. In these results, as expected, the **Always CNN** approach is never able to correctly label unknown classes, providing evidence to support **H1**.

4.4 Always CBR Variant

As an additional control, we use the *Always CBR* variant of NOD-CC (i.e., the CNN always abstains, so only CBR is used). This variant was evaluated both with observable parts information (i.e., a parts detector component was

Table 1. Performance of the various NOD-CC configurations

Variant	ρ_A	ρ_P	ρ_R	F_1	RMA	Min. ρ_A	Max. ρ_A	Med. ρ_A	σ ρ_A
Always CNN	42.99	61.31	37.98	44.32	67.27	25.98	61.27	41.80	9.15
Always CBR w/ Parts	58.45	54.30	59.66	56.18	81.70	43.49	68.93	61.33	6.57
Always CBR w/o Parts	49.82	49.77	49.41	48.21	69.67	37.49	63.44	59.78	7.27
Conditional CBR w/ Parts	59.90	56.52	60.98	58.17	83.77	54.00	64.12	60.35	2.66
Conditional CBR w/o Parts	53.73	51.39	53.75	52.44	75.15	49.84	61.91	55.23	2.67

available) and without. When parts are not available, Algorithm 1 only uses the image features during retrieval. For these experiments, the CBR component was initially given a case base containing all training instances.

Always CBR has a higher theoretical maximum accuracy than **Always CNN** because it has the ability to label an image as a novel class rather than abstaining: $max(\rho_A) = (\frac{15}{20} \times 100\%) + (\frac{5}{20} \times 25\%) = 81.3\%$. Based on the class imbalance of the datasets, the true maximum accuracy was determined to be 71.5%. Similar to with *Always CNN*, this was used to calculate the RMA. The results are shown in Table 1.

Although the availability of detectable object part information is beneficial, **Always CBR** is able to outperform **Always CNN** even without parts. The only metric **Always CNN** performs better on is precision. As we mentioned previously, this is a result of the CNN algorithm being able to abstain, thereby lowering its false positive rate. Overall, the results demonstrate the benefits CBR can provide when the full set of object classes is not known in advance. Even considering the performance of these approaches relative to their maximum accuracy (i.e., RMA), **Always CBR** still outperforms **Always CNN**. These results provide evidence to support **H2**.

4.5 Conditional CBR Variant

In this variant, we use both the CBR and CNN components (i.e., our full architecture). As described previously, the classification from the CNN is used unless the CNN abstains. If the CNN does abstain, the CBR component is used for classification. We use the same configurations (i.e., λ_a threshold and initial case base) for the CNN and CBR components as described in the previous experiments. Similar to the **Always CBR** variant, we evaluate the **Conditional CBR** both with and without parts information. The results are shown in Table 1. Across all metrics, except precision, both variants of **Conditional CBR** outperform **Always CNN**. This demonstrates that the ability of CBR to dynamically detect and learn from previously unseen class types provides significant benefit to the CNN component. In situations where the CNN abstains, the CBR component is able to provide assistance. This provides support for **H3**.

When comparing **Always CBR** to **Conditional CBR**, the **Conditional CBR** variants outperform across all five core metrics (accuracy, precision, recall, f-score, and RMA). This includes both the variants that use parts information as well as those that do not. The results show that the **Conditional CBR** performance has fewer extreme results (i.e., minimums and maximums closer to the mean) and significantly lower standard deviation. This is beneficial because it provides both improved performance as well as less uncertainty about the potential performance on an unknown dataset. Additionally, these results demonstrate the combination of both the CNN and CBR components are necessary for maximum performance; neither module is sufficient for novel object discovery on their own. These results provide support for **H4**.

5 Conclusions and Future Work

In this work we described NOD-CC, a hybrid architecture that uses Case-Based Reasoning and Convolutional Neural Networks to discover novel object types during the image classification process. NOD-CC leverages the automated feature extraction and image classification performance of CNNs while minimizing their requirement for large, pre-labeled training datasets by using CBR's instance-based learning capabilities. NOD-CC can be used with any CNN implementation so it is not tied to a specific CNN architecture, training methodology, or parameter selection. This is particularly important given the rapid advancement in the field of CNNs.

Additionally, NOD-CC can use detected object parts to further improve its performance, although it performs well even if such additional information is unavailable. We evaluated our approach on a publicly available image dataset and showed NOD-CC had improved performance over a CNN or CBR module alone. Our results demonstrated that NOD-CC was able to discover previously unknown classes of objects (i.e., not represented in training data), learn from a single instance of the novel object type, and classify future instances of those objects. Additionally, NOD-CC performed these tasks without compromising the discriminatory power of the CNN.

Future work will involve using the WordNet hierarchy in conjunction with the hierarchical multi-class capabilities afforded by Inception-style architectures in order to perform hierarchical clustering of classes. Thus, a novel class could be placed in a hierarchy relative to known classes, possibly revealing a parent-child relationship. For example, if an image dataset contained labeled images of *balloons* and *baskets*, it could be learned that they are related to a newly discovered object type, an image of a *hot air balloon*. Similarly, textual relations between the known class labels could be used to generate a more semantically meaningful label for the novel object type (e.g., *balloon basket*). We also wish to investigate additional methods for using CBR for classification. Even in our dynamic approach described in this paper, we set an abstaining threshold λ_a for detection to be used unilaterally across all classes. There is an intuitive reason to believe that a CBR system (i.e., a meta-algorithm) for determining when to

deploy a second CBR system (i.e., an image classifier) may be useful in this effort.

References

1. Begum, S., Ahmed, M.U., Funk, P., Xiong, N., Folke, M.: Case-based reasoning systems in the health sciences: a survey of recent trends and developments. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **41**(4), 421–434 (2011)
2. Chen, X., Mottaghi, R., Liu, X., Fidler, S., Urtasun, R., Yuille, A.: Detect what you can: Detecting and representing objects using holistic models and body parts. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
3. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. http://www.pascal-network.org/challenges/VOC/voc2010/workshop/index.html
4. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 580–587. IEEE Computer Society (2014)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
6. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
7. Kumar, A., Kim, J., Cai, W., Fulham, M., Feng, D.: Content-based medical image retrieval: a survey of applications to multidimensional and multimodality data. Journal of digital imaging **26**(6), 1025–1039 (2013)
8. Kurth, T., Treichler, S., Romero, J., Mudigonda, M., Luehr, N., Phillips, E., Mahesh, A., Matheson, M., Deslippe, J., Fatica, M., et al.: Exascale deep learning for climate analytics. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis. p. 51. IEEE Press (2018)
9. Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Duerig, T., et al.: The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. arXiv preprint arXiv:1811.00982 (2018)
10. Lenz, M., Bartsch-Spörl, B., Burkhard, H.D., Wess, S.: Case-based reasoning technology: from foundations to applications, vol. 1400. Springer (2003)
11. López-Sánchez, D., Corchado, J.M., Arrieta, A.G.: A cbr system for efficient face recognition under partial occlusion. In: International Conference on Case-Based Reasoning. pp. 170–184. Springer (2017)
12. Macura, R.T., Macura, K.J.: Macrad: Radiology image resource with a case-based retrieval system. In: International Conference on Case-Based Reasoning. pp. 43–54. Springer (1995)
13. Micarelli, A., Neri, A., Sansonetti, G.: A case-based approach to image recognition. In: European Workshop on Advances in Case-Based Reasoning. pp. 443–454. Springer (2000)
14. Minsky, M., Papert, S.: Perceptrons: An Introduction to Computational Geometry. MIT Press, Cambridge, MA, USA (1969)

15. Page, A., Turner, J., Mohsenin, T., Oates, T.: Comparing raw data and feature extraction for seizure detection with deep learning methods. In: The Twenty-Seventh International Flairs Conference (2014)
16. Patrini, G., Rozza, A., Krishna Menon, A., Nock, R., Qu, L.: Making deep neural networks robust to label noise: A loss correction approach. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1944–1952 (2017)
17. Perner, P., Holt, A., Richter, M.: Image processing in case-based reasoning. Knowledge Eng. Review **20**, 311–314 (09 2005). https://doi.org/10.1017/S0269888906000671
18. Ravi, D., Wong, C., Lo, B., Yang, G.Z.: A deep learning approach to on-node sensor data analytics for mobile or wearable devices. IEEE journal of biomedical and health informatics **21**(1), 56–64 (2017)
19. Sani, S., Massie, S., Wiratunga, N., Cooper, K.: Learning deep and shallow features for human activity recognition. In: International Conference on Knowledge Science, Engineering and Management. pp. 469–482. Springer (2017)
20. Sani, S., Wiratunga, N., Massie, S.: Learning deep features for knn-based human activity recognition. (2017)
21. Sathyanarayana, A., Joty, S., Fernandez-Luque, L., Ofli, F., Srivastava, J., Elmagarmid, A., Arora, T., Taheri, S.: Sleep quality prediction from wearable data using deep learning. JMIR mHealth and uHealth **4**(4), e125 (2016)
22. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. CoRR **abs/1512.00567** (2015), http://arxiv.org/abs/1512.00567
23. Turner, J.T., Floyd, M.W., Gupta, K.M., Aha, D.W.: Novel object discovery using case-based reasoning and convolutional neural networks. In: Cox, M.T., Funk, P., Begum, S. (eds.) Case-Based Reasoning Research and Development. pp. 399–414. Springer International Publishing, Cham (2018)
24. Turner, J., Page, A., Mohsenin, T., Oates, T.: Deep belief networks used on high resolution multichannel electroencephalography data for seizure detection. In: 2014 AAAI Spring Symposium Series (2014)
25. Xie, S., Girshick, R.B., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. CoRR **abs/1611.05431** (2016), http://arxiv.org/abs/1611.05431