ClassSim: Similarity between Classes Defined by Misclassification Ratios of Trained Classifiers

Kazuma Arino1*, Yohei Kikuta2*
1 Self-employed†
2 Cookpad Inc.
kazuma.arino@gmail.com, yohei-kikuta@cookpad.com

Abstract

Deep neural networks (DNNs) have achieved exceptional performances in many tasks, particularly, in supervised classification tasks. However, achievements with supervised classification tasks are based on large datasets with well-separated classes. Typically, real-world applications involve wild datasets that include similar classes; thus, evaluating similarities between classes and understanding relations among classes are important. To address this issue, a similarity metric, ClassSim, based on the misclassification ratios of trained DNNs is proposed herein. We conducted image recognition experiments to demonstrate that the proposed method provides better similarities compared with existing methods and is useful for classification problems.

1 Introduction

Deep neural networks (DNNs) have demonstrated improved performance for various tasks. In particular, supervised classification tasks in computer vision are said to be solved. This statement is correct if the datasets are ideal, i.e., they include a large number of images, well-annotated accurate labels, well-separated, semantically different target classes and identical distributions of training and test data. As an ideal case, ImageNet [Deng et al., 2009] classes, which are used to evaluate classification tasks, are well-organized [Deselaers and Ferrari, 2011]: usually visually distinct, and distinguishable from a taxonomy perspective.

However, real-world applications typically involve non-ideal datasets. For example, consumer generated medias generate huge but wild data [Izadinia et al., 2015]. This type of data forms supervised datasets wherein labels are manually assigned by users. As a result, in such datasets, labels for given similar images can vary and classes can be disorganized. In addition, classes that are objective variables of models are not always well-separated semantically, which means that a dataset may contain similar classes, e.g., spaghetti, carbonara, and Alfredo classes. These classes are similar and difficult to distinguish visually.

Herein, we focus on the difficulties associated with handling fluctuated labels for given similar images and estimating the similarities between classes. Once good similarities are obtained, visual relations among classes are evident and the performance of various machine learning tasks, such as classification, can be improved. Note that defining similarities is important, but difficult. Previous studies have imposed rather strong assumptions, e.g., data probabilistic distributions are Gaussian, simple and low dimensional features can represent various images.

A similarity metric based on the misclassification ratios of a trained DNN is proposed herein. The proposed similarity only depends on an assumption that DNN classifiers can capture the characteristics of data distribution. We believe this assumption is correct because DNNs, particularly convolutional neural networks, have demonstrated high performance for image classification [Russakovsky et al., 2015].

We find that the proposed similarity is useful for various vision tasks, such as understanding semantic gaps, creating robust models using misclassified examples [Li and Snoek, 2013], and reorganizing target classes. To the best of our knowledge, no previous studies have investigated inter-class similarity computations based on DNNs predictions.

2 Related work

There exists two types of similarities in recognition problems; similarity between elements (a pair of images such as single city image and another single buildings image) and similarity between classes (a pair of classes such as city and buildings), see Figure 1. Similarity between elements is employed to search for visually similar products and in visual authentication systems, and similarity between classes is applied to understand semantic gaps and visual taxonomies.

1*Equal contribution.
†This work was done while at Cookpad Inc.
‡Source code including all experimental results is available at https://github.com/karino2/ClassSim/
2.1 Similarities between images

Many methods to compute similarity between images have been proposed. Recently, DNNs have been used to extract image features to compute similarities [Wang et al., 2014; Han et al., 2015]. For example, DNNs based similarities have been applied to image retrieval [Wu et al., 2013], person re-identification [Yi et al., 2014], facial recognition [Schroff et al., 2015], and visual similarity for product design [Bell and Bala, 2015].

Note that image-to-image similarity is well studied in various aspects; it is out of the scope of this study.

2.2 Similarities between classes

Few methods exist to compute the similarity between classes. Compared to image-to-image similarity, estimating class-to-class similarity is much more difficult because a class can include various images and the number of images is not fixed.

However, a method to estimate the similarities between classes has been proposed [Wang et al., 2008; Guan et al., 2009]. In that method, images are divided into patches, and features are extracted from each patch using traditional methods, such as RGB color moment. In addition, to compute the distance between classes, we must assume that the images are generated from Gaussian mixture models (GMMs). Note that the number of GMM components must be determined manually relative to the number of target classes. In addition, the distance between classes expresses an inverse relation with similarities; they are not normalized, and their absolute values are meaningless. Here, two distances are involved, i.e., parametric distance (PD), which is the quadratic distance of the means and variances of a GMM, and an approximation of KL divergence. These two methods return similar results. Here, strong assumptions and simplifications were used to treat inter-class similarities realistically.

In this study, we find ways to improve inter-class similarity and compare our results to those obtained using PD.

2.3 Open set classification

Open set classification problems [Bendale and Boult, 2015] are inherent and difficult in real-world applications. Thus, few studies have addressed such problems.

However, a solution that employs features extracted using a DNN and meta-recognition has been proposed [Bendale and Boult, 2016]. This solution is useful to eliminate dissimilar unknown unknowns and is, in particular, effective for fooling images.

In addition, support vector machine-based methods have been studied for broader applications. Some studies [Schölkopf et al., 2001; Scheirer et al., 2014] attempted to discriminate a target class from other classes including unknown unknowns. Such studies can be interpreted as attempts to find methods to improve one vs. rest (OVR) classifiers to handle unknown unknowns. In other words, they attempt to generalize classifiers by isolating a target class from the other classes from various perspectives. Note that these studies did not employ DNNs.

In this study, as a first step, we created OVR classifiers using a DNN and attempted to improve classification performance using a supervised dataset.

3 Problem formulation

The target problem is defining similarities between classes that include an arbitrary number of images. Here, let $c \in C$ be a class, $X_c$ be a set comprising images whose labels are all $c$, and $x \in \bigcup X_c$ be an image. The goal is to formulate a quantitative similarity between $c_i$ and $c_j$. In the following, we consider a case in which one image has one and only one label.

We consider three types of labels. The first is latent labels. We assume that images are generated by unknowable generative models whose latent variables correspond to the labels. An image $x$ is generated by following a probabilistic distribution $p(x|c)$. Here, any functional form of the distribution is not assumed. Generally, latent labels are difficult to estimate by its nature. The second is annotated labels. Here, labels are assigned manually and used as supervised datasets to train a model, corresponding to the labels of $X_c$. After being generated, an image from $p(x|c)$ is not always annotated as $c$ owing to stochasticity\footnote{Our original intent was to improve OVR classifiers to handle open set problems in our service.}. We assume that assigning probabilities of annotated labels are controlled by probabilities $p(c|x)$. The third label type is predicted labels. Here, labels are set by the distribution $p(c|x)$ in a deterministic manner. A predicted label is determined as follows:

$$\arg \max_{c \in C} p(c|x) \quad \text{where} \quad \sum_{c \in C} p(c|x) = 1. \quad (1)$$

As shown in Figure 2, an image is generated from $p(x|c = c_i)$, labels are assigned by following the probabilities $p(c_i|x)$ and $p(c_j|x)$, and the predicted label is determined by $\arg \max_{c \in C} p(c|x)$.

\footnote{This is natural because annotated labels can differ for different people. For example, an image generated from buildings can be annotated as buildings by one person and city by another.}
We define the similarity between classes $c_i$ and $c_j$ as the intersection area of two probability distributions:

$$A_{\text{intersection}} = \int_{p(x|c_i) < p(x|c_j)} p(x|c_j) \, dx + (i \leftrightarrow j).$$

The intersection area represents the occurrence frequency of a condition wherein it is impossible to uniquely identify the latent labels of the generated images. The size of this area reflects the indistinguishability between two classes. The larger the area, the more similarity the two classes show. The proposed similarity has the following properties:

- normalization: possible value range is $[0, 1]$,
- symmetry: $\{c_i, c_j\}$ and $\{c_j, c_i\}$ provide the same value.

Since $p(x|c)$ is intractable, an exact computation of similarity is difficult. Therefore, the problem is to estimate similarity as approximately as possible using $p(c|x)$, which can be learned approximately from the given datasets $\bigcup_c X_c$.

### 4 Proposed approach

In this section, we propose ClassSim to approximately represent Equation 2. ClassSim is defined using classifiers $f(x)$ trained to learn $p(c|x)$ for the given datasets. This is the main contribution of this paper.

In addition, we propose two level model that enhances the performance of OVR classifiers as an application of the proposed ClassSim.

#### 4.1 ClassSim

We first describe an ideal case. Here, the prior distributions are identical for the class pair $(c_i, c_j)$, i.e., $p(c_i) = p(c_j)$, and we have an ideal binary classifier that returns the score according to the true distribution $p(c|x)$:

$$f_{\text{ideal}}^{c_i, c_j}(x) = \begin{cases} 1 & \text{for } p(c_j|x) > p(c_i|x), \\ 0 & \text{for } p(c_j|x) \leq p(c_i|x). \end{cases}$$

For images $x_{ci} \in X_{c_i}$ where $c_i$ is the annotated label, misclassification occurs when the classifier returns 1. Although $f_{\text{ideal}}^{c_j, c_i}$ knows the true distribution $p(c|x)$, this misclassification is unavoidable because $x_{ci}$ can be generated from the region $p(c_j|x) > p(c_i|x)$ (Figure 2).

Let $N_{c_i} = |X_{c_i}|$ and $N_{c_j|c_i}$ be the total number of misclassifications defined as

$$N_{c_j|c_i} = \sum_{x_{ci} \in X_{c_i}} I[f_{\text{ideal}}^{c_j, c_i}(x_{ci}) = 1],$$

where $I$ is the indicator function. Then, we can show the following under ideal conditions:

$$A_{\text{intersection}} \simeq \frac{N_{c_j|c_i}}{N_{c_i}} + \frac{N_{c_i|c_j}}{N_{c_j}}. \quad (5)$$

To understand Equation 5, consider that the image space is discretized into a finite number of volumes and the distributions remain constant in each volume. Then, consider image $x_0 \in X_{c_i}$ satisfying $p(c_j|x_0) > p(c_j|x_0)$ and a small volume $\Delta x$ around the point where the distributions remain constant. The effective number within the volume, denoted $\Delta N_{c_j|c_i}$, is expressed as follows:

$$\Delta N_{c_j|c_i} = N_{c_i}p(x_0|c_i)\Delta x.$$  \quad (6)

Taking summation, the left side of Equation 6 becomes

$$\sum \{x|x \in X_{c_i} \cap p(c_j|x) > p(c_i|x)\} \Delta N_{c_j|c_i} = N_{c_j|c_i}. \quad (7)$$

The right side of Equation 6 can be expressed as follows:

$$N_{c_i} \sum_{\{x|x \in X_{c_i} \cap p(c_j|x) > p(c_i|x)\}} p(x|c_i)\Delta x,$$

$$= N_{c_i} \sum_{\{x|x \in X_{c_i} \cap p(c_j|x) > p(c_i|x)\}} p(x|c_i)\Delta x,$$

$$\simeq N_{c_i} \times (\text{the right half side of } A_{\text{intersection}}). \quad (9)$$

where $p(c_j|x) = p(x|c_j)/p(x|c_i)$, which is ensured by Bayes’ theorem and the assumed identical priors. By the same argument, by interchanging $i$ and $j$, we can derive Equation 5.

#### General definition of ClassSim

Here, we generalize the above ideal binary case. Generally, the prior can be different for each class, and the exact form of $p(c|x)$ cannot be obtained. Therefore, we define ClassSim constructed by the misclassification ratios of the trained classifiers, which approximate the distribution:

$$\text{ClassSim}(X_{c_i}, X_{c_j}) = \frac{1}{2} \left( \frac{N_{c_j|c_i}}{N_{c_i}} + \frac{N_{c_i|c_j}}{N_{c_j}} \right),$$

where $N_{c_j|c_i}$ is the ratio of the number of elements $x_{ci} \in X_{c_i}$ predicted as $c_j$ by the classifier. Generally, different classifiers can be used to compute $N_{c_j|c_i}$ and $N_{c_i|c_j}$; thus, we require $|C|(|C| - 1)$ classifiers to compute the similarities of all pairs of classes in this case. The factor 1/2 ensures that the value is in the range $[0, 1]$ because the possible maximum value of $N_{c_j|c_i}$ can be $N_{c_i}$. This definition obviously possesses symmetry under $i \leftrightarrow j$.

From a classifier perspective, the proposed similarity can be interpreted as the difficulty of classification between two classes. In addition, scores across different pairs of classes can be compared because their absolute values have meaning, that is, the ratio of misclassification.
The important points of the proposed similarity are that (1) it only uses trained classifiers and (2) no assumption is made about the functional forms of the distributions or geometric structures of the feature space, which are significant differences observed from previous methods. Owing to recent advances in DNN classifiers, it is easier to create good classifiers that can capture the distribution $p(c|x)$ than directly estimating the generative distribution $p(x|c)$.

One vs. Rest classifier case

As a concrete classifier case, we introduce an OVR classifiers case for $\text{ClassSim}$ computation. Note that this is one case used for the experiments discussed in the next section. In this case, there are $|C|$ classes and $|C|$ classifiers $f_{c, \text{other}} \in [0, 1]$. We can compute $\text{ClassSim}$ using $f_{c, \text{other}}$ and $f_{c, \text{other}}$. Here, the number of misclassified samples for $x_{ci} \in X_{ci}$ is given by

$$N_{cj}|ci = \sum_{x_{ci} \in X_{ci}} I[f_{cj,\text{other}}(x_{ci}) > 0.5].$$

From an implementation perspective, we only require $|C|$ classifiers to compute the similarities of all pairs rather than $|C|(|C| - 1)/2$ with the binary classifier case.

Compared to the ideal binary classifier case, we can interpret the OVR classifier $f_{cj,\text{other}}$ as the approximation of $f_{ij,\text{ideal}}$ by averaging $c_i$ for all $c_i \in C \setminus \{c_j\}$. If $c_i$ is similar to $c_j$ and rather different from the other classes, the similarity tends to be large because the classifications are easy to "misclassify". The misclassification ratio $N_{cj}|ci$ can be understood as how $c_j$ is similar to $c_i$ compared to the other classes. From this observation, $\text{ClassSim}$ is still a good metric for similarity between two classes.

Multi-class classifier case

Here, we consider a multi-class classifier case. We require only one classifier $f \in \mathbb{R}^{|C|}$ s.t. $\sum_{c \in C} f_c(x) = 1$ in this case.

The number of misclassified samples for $x_{ci} \in X_{ci}$ is given by

$$N_{cj}|ci = \sum_{x_{ci} \in X_{ci}} I[\text{arg max}_c f_c(x_{ci}) = c_j].$$

For a pair of two similar classes, the similarity of the multi-class case shows the same tendency as the OVR classifier case; however, its value is relatively smaller. We demonstrate that this phenomenon is true and compare both cases in detail in the next section.

4.2 Two level model

The proposed $\text{ClassSim}$ is useful for understanding the similarities between classes and various applications, such as improving classifiers. As an application of $\text{ClassSim}$, we introduce two level model that enhances OVR classifications.

As stated previously, improvements to OVR classifications lead to better solutions for open set problems. Among the many different potential improvement directions, we focus on the classification of datasets that include similar classes because this is a difficult problem in real-world applications for which the proposed similarity has high affinity.

Baseline model

The simple OVR classifiers introduced in the previous subsection is used as a baseline model. For each target class $c_i$, an OVR classifier $f_{c,\text{other}}$ is trained using datasets $X_{ci}$ and $\bigcup_{c \in C \setminus \{c_i\}} X_c$. In total, we have $|C|$ OVR classifiers.

In the prediction phase, these trained OVR classifiers are applied in some order. Here, each OVR classifier is trained individually; thus, the scores across different classifiers cannot be compared. Therefore, when the first OVR classifier returns a score above a threshold (we use 0.5 in this paper) is found, we select its target label as a predicted label. Although we can use some heuristics based on domain knowledge in practical applications, simple alphabetical order of class names is used herein. If no classifier has a score greater than the threshold, the predicted label is defined as $\text{none}$. 

Two level model

We propose an enhancement to OVR classifiers by constructing one more set of OVR classifiers $f_{c,\text{other}}^{(2)}$ that is applied after the first set of classifiers $f_{c,\text{other}}$. For each target class $c_i$, $f_{c,\text{other}}^{(2)}$ is constructed as follows. First, a set of classes including similar classes to $c_i$ is defined (we use 0.1 as the similarity threshold in this paper):

$$C_{c_i, \text{sim}} = \{c \in C \setminus \{c_i\} | \text{ClassSim}(X_{ci}, X_c) > 0.1\}.$$ (13)

Second, OVR classifiers are trained using $X_{ci}$ and $X_c$. From the construction procedure considered herein, $f_{c,\text{other}}^{(2)}$ can distinguish small differences among similar classes. Note that the same threshold can be used for all target classes $c_i$ because $\text{ClassSim}$ can compare across different pairs of classes, which is why we can collect similar classes without human intervention.

Note that a situation in which there is no similar class for some target class may occur. In this case, we have no $f_{c,\text{other}}^{(2)}$ for the target class.

Two level model are defined by applying $f_{c,\text{other}}^{(2)}$ after performing $f_{c,\text{other}}$. Here, we require one more threshold for $f_{c,\text{other}}^{(2)}$, setting 0.5 as with that of $f_{c,\text{other}}$. The pseudocode of two level model is as follows.

Algorithm 1 Definition of two level model

Require: image $x$, classes $c \in C$, OVR classifiers $f, f_{c,\text{other}}^{(2)}$ for $c \in C$ do
if $f_{c,\text{other}}^{(2)}(x) > 0.5$ then
    if $f_{c,\text{other}}^{(2)}$ exists then
        if $f_{c,\text{other}}^{(2)}(x) > 0.5$ then
            return $c$
        else
            return $c$
    else return $\text{none}$
end
5 Experiments

Two experiments were conducted to demonstrate the effectiveness of the proposed methods. The first experiment involved estimating the similarities between classes, and the results were compared to those of a previous study [Wang et al., 2008]. The second experiment focused on enhancing OVR classifiers using ClassSim.

To compare our results with the previous study, we attempted to collect the same datasets (16 classes of images gathered using the Yahoo image search API). Unfortunately, this API is no longer available; therefore, we use Bing image search to collect nearly equivalent datasets. We attempted to collect 1,000 images for each class employed in the previous study, but some of the classes contained less than 1,000 images.

In total, we obtained (16 classes, 11,803 images). We divided these images into (training) : (validation) : (test) = 0.8×0.8 : 0.8×0.2 : 0.2 datasets.

5.1 Similarities between classes

We trained 16 OVR classifiers using the training set, and these classifiers were trained using transfer learning from a pre-trained Inception v3 [Szegedy et al., 2016]. We then computed ClassSim on the validation set using the trained classifiers for each pair of classes.

For comparison, we reproduced the results of the previous study. In the previous study, each image was divided into 5×5 patches and traditional image features, such as RGB color moment, were used to compute PDs between classes. We used these distance values as similarities (note that smaller values indicate greater similarity).

In addition, we also conducted the same experiment using a single trained multi-class classifier. We show computed similarities and differences between the results of the OVR case and those of the multi-class case.

In this subsection we show the three most similar classes for each target class. The full results of computed similarities are shown in Appendix A.

Similar pair

The results of ClassSim computed by the OVR classifiers and PD are shown in Table 1. There are some overlaps between the two results. For example, the pair (bay, beach) was the most similar common pair in both cases, which is a natural result for a human sense. In addition, both methods provided \{f-16, city, clouds, bay\} as the most similar class for \{boeing and helicopter, buildings, sky, ocean\}, respectively.

We observed significant differences relative to other combinations. For example, the most similar class to city was buildings for CS and ocean for PD. This indicates that the proposed method obviously yielded a better result, see Table 3. Furthermore, CS provided \{sunset, sunrise\} as the most similar pair, whereas PD provided f-16 as the class most similar to sunset or sunrise. This demonstrates that the proposed method can bridge semantic gaps better than the previous method.

Comparison within a row

Here, we compare the relative scores among classes for a single target class, which leads to another advantage of the proposed similarity.

For example, the three classes most similar to buildings and its CS scores were \{city:0.656, ships:0.092, bay:0.069\}. Here, the score difference of the top two classes (a difference of approximately seven times) seems sensible because city is similar to buildings but ships is not. In contrast, PD provided \{city:9624, bay:9813, ocean:10152\}. Since the score difference between city and bay was less than that of bay and ocean, PD cannot distinguish as well as the proposed CS.

As a result, we conclude that the proposed method is much more robust than the previous method. In fact, our reproduced results for PD were a little different from those of the original paper.

Comparison across rows

Here, we investigate the differences across rows, and focus on birds and sunrise for CS. The highest score for birds and sunrise was 0.045 and 0.902, respectively. Note that the latter is more than 20 times greater than the former. This result is interpretable because birds is not similar to any other class and sunrise is very similar to sunset.

However, the same argument cannot be applied for the PD case. The shortest distance of birds was less than that of sunrise, which indicates that inter-row comparison is clearly meaningless for PD.

In contrast, the proposed method has a clear meaning for its absolute value. By definition, the value directly represents the misclassification ratio. We can think of the value as a quantitative measure of the challenges in distinguishing two classes.

Carrying this observation further, we can use the similarity to redesign classes, such as merging similar classes. For example, in this case, we may merge bay and beach for better classifications.7

Comparison between OVR and multi-class classifiers

The results of ClassSim computed by the multi-class classifier are shown in Table 2 where the results are compared with those of the OVR case.

Overall, the two results show good agreement. We can see that both case yielded the same most similar classes for each target class except for \{birds, city, ships\}. Although there exists other differences in the results, the multi-class case also leads better performances than PD. We can conclude that the proposed similarity is useful for different types of classifiers.

Note that the similarities of the multi-class case were lower than those of the OVR case. This is a natural consequence because in Equation 12 images whose annotated labels are \(c_i\) and predicted labels are \(c_j\) are counted as the misclassifications; therefore, images predicted as \(c \in C/\{c_i, c_j\}\) do not increase the value of the similarity. In contrast, the misclassifications of the OVR case include all images that are predicted as \(c_j\) by the binary classifier \(f_{c_j, other}\).

7We did this kind of redesign target classes in our service and found it effective.
The differences of scores were more obvious for the OVR case than the multi-class case. For example, the three classes most similar to \textit{f-16} and those scores were \{\textit{boeing}:0.0258, \textit{helicopter}:0.188, \textit{ships}:0.126\} for the OVR case, and \{\textit{boeing}:0.040, \textit{helicopter}:0.038, \textit{mountain}:0.013\} for the multi-class case. The OVR case gave clearer differences between \{\textit{boeing, helicopter}\}.

Let us explain some differences in the results. The most similar class to \textit{city} was \textit{buildings} for the OVR case and \textit{bay} for the multi-class case. This result is reasonable because we found some \textit{bay} images contain building. The most similar class to \textit{ships} was \textit{f-16} for the OVR case and \textit{bay} for the multi-class case. In this case it’s not easy to judge which result is better.

We conclude that, in this experiment, \textit{ClassSim} based on the OVR classifiers is slightly better than that of the multi-class classifier.

### 5.2 Enhancement of OVR classifiers

In this experiment, we evaluated the test dataset accuracy of the proposed \textit{two level model} by following Algorithm 1. Here, we used the same 16 OVR classifiers as in the previous subsection for the first set of classifiers. From the results in Table 1, we trained 14 \(f^{(2)}_{c,\text{other}}\) with \{training, validation\} datasets because \textit{birds} and \textit{face} have no similar classes above the threshold.

The classification results are shown in Table 5. The proposed \textit{two level model} demonstrated 11% better accuracy than the baseline model.

|                      | baseline model | \textit{two level model} |
|----------------------|----------------|-------------------------|
| accuracy             | 0.552          | \textbf{0.611}          |

Table 5: Classification results of baseline model and \textit{two level model}.

To observe the ways in which \textit{two level model} improved classifications, we show some images in Table 4. Since \(f^{(2)}_{c,\text{other}}\) was trained using datasets that only include similar classes, it can distinguish finer differences.

### 6 Summary

Herein, we formalized the similarities of a pair of classes and proposed \textit{ClassSim} based on the misclassification ratio of the trained classifiers that can well express the similarities.

Our experimental results demonstrate that the proposed similarity yields better performance than previous methods. The scores were easier to compare across multiple classes, and the differences were much clearer than those of prior studies. Thus, the proposed method can bridge semantic gaps better than previous methods. We then presented the effectiveness of \textit{two level model} based on \(f^{(2)}_{c,\text{other}}\) classifiers trained using only similar classes. Using the proposed similarity, we could collect similar classes without human intervention. The experimental results showed that \textit{two level model} improved the accuracy of the baseline model that is a simple OVR classi by approximately 11%.

Note that we have used the model in practical applications with over 150 classes and approximately 500,000 images. It has been shown that performance relative to unknown unknowns has been improved. In future, we plan to compare the proposed model to previous studies with an open set problem setting comprising publicly available dataset.
### Table 1: Top three similar classes and their scores by ClassSim (CS) and Parametric Distance (PD). Each row shows the three most similar classes to the class in the first column. CS is the similarity score ranging from 0 to 1 (higher values indicate greater similarity). PD is a positive real number (lower values indicate greater similarity).

|        | CS (OVR) | CS (multi-class) | PD |
|--------|----------|------------------|-----|
| bay    | beach:0.626 | ocean:0.320 | city:0.301 | beach:0.658 | mountain:0.951 | birds:0.7192 |
| beach  | bay:0.626 | ocean:0.245 | mountain:0.114 | bay:0.658 | mountain:0.959 | birds:0.7014 |
| birds  | ocean:0.045 | face:0.037 | sunset:0.028 | helicopter:0.5656 | f-16:0.6490 | boeing:0.6490 |
| boeing | f-16:0.258 | helicopter:0.153 | ocean:0.067 | f-16:0.3438 | clouds:0.3525 | helicopter:0.4918 |
| buildings | city:0.656 | ships:0.092 | bay:0.069 | city:0.9624 | bay:0.9813 | ocean:1.0152 |
| city   | buildings:0.656 | bay:0.301 | ships:0.097 | ocean:0.8576 | bay:0.8679 | mountain:0.9585 |
| clouds | sky:0.0787 | ocean:0.026 | sunset:0.128 | f-16:0.3421 | boeing:0.3525 | helicopter:0.5067 |
| face   | ocean:0.051 | sunrise:0.040 | birds:0.037 | f-16:0.7768 | helicopter:0.7849 | clouds:0.8118 |
| f-16   | boeing:0.258 | helicopter:0.188 | ship:0.126 | clouds:0.3421 | boeing:0.3438 | helicopter:0.4682 |
| helicopter | f-16:0.188 | boeing:0.153 | ship:0.098 | f-16:0.4682 | boeing:0.4918 | clouds:0.5067 |
| mountain | bay:0.188 | beach:0.114 | ocean:0.093 | beach:0.6909 | bay:0.6951 | birds:0.7117 |
| sky    | clouds:0.787 | sunset:0.317 | sunrise:0.302 | clouds:0.6609 | f-16:0.7161 | boeing:0.7467 |
| ships  | f-16:0.126 | ocean:0.108 | helicopter:0.098 | helicopter:0.7506 | birds:0.7520 | bay:0.7983 |
| sunset | sunrise:0.902 | sky:0.317 | ocean:0.163 | f-16:0.5253 | boeing:0.5365 | clouds:0.5447 |
| sunrise | sunset:0.902 | sky:0.302 | ocean:0.157 | f-16:0.5885 | boeing:0.6028 | clouds:0.6287 |
| ocean  | bay:0.320 | sky:0.271 | clouds:0.260 | bay:0.7270 | beach:0.8070 | mountain:0.8424 |

### Table 2: Top three similar classes and their scores by ClassSim computed using the one vs. all (OVR) classifiers and ClassSim computed using the multi-class classifier. Each row shows the three most similar classes to the class in the first column. The similarity score ranging from 0 to 1 (higher values indicate greater similarity).

|        | ClassSim (OVR) | ClassSim (multi-class) |
|--------|----------------|------------------------|
| bay    | beach:0.626 | ocean:0.320 | city:0.301 | beach:0.246 | city:0.123 | mountain:0.093 |
| beach  | bay:0.626 | ocean:0.245 | mountain:0.114 | bay:0.246 | city:0.040 | buildings:0.015 |
| birds  | ocean:0.045 | face:0.037 | sunset:0.028 | face:0.011 | ocean:0.009 | mountain:0.008 |
| boeing | f-16:0.258 | helicopter:0.153 | ocean:0.067 | f-16:0.040 | sky:0.005 | helicopter:0.005 |
| buildings | city:0.656 | ships:0.092 | bay:0.009 | city:0.122 | bay:0.044 | ships:0.017 |
| city   | buildings:0.656 | bay:0.301 | ships:0.097 | bay:0.123 | buildings:0.122 | ships:0.013 |
| clouds | sky:0.787 | ocean:0.260 | sunset:0.128 | sky:0.248 | ocean:0.041 | mountain:0.021 |
| face   | ocean:0.051 | sunrise:0.040 | birds:0.037 | ocean:0.012 | birds:0.011 | sunset:0.008 |
| f-16   | boeing:0.258 | helicopter:0.188 | ship:0.126 | boeing:0.040 | helicopter:0.038 | mountain:0.013 |
| helicopter | f-16:0.188 | boeing:0.153 | ships:0.098 | f-16:0.038 | ships:0.025 | bay:0.011 |
| mountain | bay:0.188 | beach:0.114 | ocean:0.093 | bay:0.093 | clouds:0.021 | ocean:0.016 |
| sky    | clouds:0.787 | sunset:0.317 | sunrise:0.302 | clouds:0.248 | sunset:0.106 | sunrise:0.057 |
| ships  | f-16:0.126 | ocean:0.108 | helicopter:0.098 | bay:0.061 | helicopter:0.025 | ocean:0.022 |
| sunset | sunrise:0.902 | sky:0.317 | ocean:0.163 | sunrise:0.353 | sky:0.106 | ocean:0.026 |
| sunrise | sunset:0.902 | sky:0.302 | ocean:0.157 | sunset:0.353 | sky:0.057 | bay:0.020 |
| ocean  | bay:0.320 | sky:0.271 | clouds:0.260 | bay:0.087 | clouds:0.041 | beach:0.040 |

### Table 3: Random samples of images whose classes are city, buildings, and ocean.

![Random samples of images](image)
Table 4: Images improved by *two level model*. The left class is misclassified by the baseline model. The right class is the true label predicted by *two level model*.
## Full results of experiments

In this appendix, we provide the full tables of the similarity computations for both CS(OVR)-PD experiment and CS(OVR)-CS(multi-class) experiment. We split the full table into three tables in the both cases.

### CS(OVR) and PD experiment

|     | bay   | beach  | birds  | boeing | buildings | city |
|-----|-------|--------|--------|--------|-----------|------|
| **CS** | beach:0.626 | bay:0.626 | ocean:0.045 | f-16:0.258 | city:0.656 | buildings:0.656 |
|     | PD    | beach:6588 | bay:6588 | helicopter:5656 | f-16:3438 | ocean:8576 |
| **CS** | ocean:0.320 | mountain:6951 | face:0.037 | f-16:6490 | f-16:0.153 | clouds:3525 |
|     | PD    | mountain:6909 | face:0.028 | boeing:6490 | ocean:0.067 | helicopter:4918 |
| **CS** | city:0.301 | mountain:0.114 | sunset:0.028 | ocean:0.067 | bay:0.069 | ocean:10152 |
|     | PD    | birds:7192 | birds:7014 | helicopter:4918 | bay:0.069 | ocean:10152 |
| **CS** | mountain:0.188 | ocean:7270 | f-16:0.277 | sunset:0.059 | sunset:0.029 | beach:0.073 |
|     | PD    | ocean:7270 | helicopter:7738 | sunset:0.666 | sunset:0.029 | buildings:9624 |
| **CS** | ships:0.077 | sunset:0.095 | sunset:0.095 | city:0.035 | beach:0.026 | mountain:0.060 |
|     | PD    | sunset:8008 | helicopter:0.019 | clouds:6681 | sunrise:6028 | beach:9718 |
| **CS** | buildings:0.069 | city:0.073 | sunrise:0.013 | beach:7014 | bay:0.025 | sunrise:0.026 |
|     | PD    | buildings:0.069 | sunrise:0.013 | beach:7014 | bay:0.025 | sunrise:0.026 |
| **CS** | sunset:0.056 | sunset:0.056 | sunset:0.056 | city:0.035 | beach:0.026 | mountain:0.060 |
|     | PD    | city:8679 | city:0.030 | ocean:8070 | sunset:0.029 | ocean:8576 |
| **CS** | city:8679 | building:0.026 | building:0.026 | city:0.035 | beach:0.026 | mountain:0.060 |
|     | PD    | building:0.026 | building:0.026 | city:0.035 | beach:0.026 | mountain:0.060 |
| **CS** | f-16:0.028 | f-16:0.028 | city:0.044 | city:0.044 | city:0.044 | city:0.044 |
|     | PD    | f-16:9124 | f-16:9124 | sky:0.004 | sky:0.004 | f-16:9124 |
| **CS** | boeing:0.025 | clouds:9547 | sky:0.004 | sky:0.004 | sky:0.004 | sky:0.004 |
|     | PD    | clouds:9672 | sky:0.004 | sky:0.004 | sky:0.004 | sky:0.004 |
| **CS** | face:0.007 | clouds:0.141 | city:9718 | city:9718 | city:9718 | city:9718 |
|     | PD    | buildings:9813 | city:9718 | city:9718 | city:9718 | city:9718 |
| **CS** | birds:0.004 | sunrise:10123 | building:0.011 | building:0.011 | building:0.011 | building:0.011 |
|     | PD    | sunrise:10123 | building:0.011 | building:0.011 | building:0.011 | building:0.011 |

Table 6: [1/3] Comparison of the similarities of ClassSim (CS) computed by the one vs. rest (OVR) classifiers and parametric distance (PD). Column name represents the target class. The pairs of {class : similarity} are shown in descending order of the similarities for each column. CS is the similarity score ranging from 0 to 1 (higher values indicate greater similarity). PD is a positive real number (lower values indicate greater similarity).
Table 7: [2/3] Comparison of the similarities of ClassSim (CS) computed by the one vs. rest (OVR) classifiers and parametric distance (PD). Column name represents the target class. The pairs of \{class : similarity\} are shown in descending order of the similarities for each column. CS is the similarity score ranging from 0 to 1 (higher values indicate greater similarity). PD is a positive real number (lower values indicate greater similarity).

|        | clouds | face   | f-16   | helicopter | mountain | sky    |
|--------|--------|--------|--------|------------|----------|--------|
| **CS** | sky:0.787 | ocean:0.051 | f-16:0.188 | f-16:0.188 | bay:0.188 | clouds:0.787 |
|        | f-16:3421 | f-16:7768 | f-16:4682 | beach:0.188 | beach:6909 | clouds:6609 |
| **PD** | ocean:0.260 | sunse:0.040 | helicopter:0.188 | beach:0.114 | f-16:7161 | sunset:0.317 |
|        | helicop:3525 | boeing:0.153 | bay:4918 | f-16:6609 | sunset:0.302 | boeing:7467 |
|        | helicop:5067 | ships:0.098 | clouds:0.506 | sky:4682 | sky:4682 | clouds:0.506 |
| **CS** | sunse:0.116 | birds:0.037 | ship:0.126 | ocean:0.093 | birds:0.171 | sunse:0.271 |
|        | helicop:5067 | ship:0.126 | clouds:0.506 | sky:4682 | sky:4682 | clouds:0.506 |
| **PD** | sunse:0.085 | sunset:0.030 | mountain:0.028 | clouds:0.085 | mountain:0.055 | sunset:0.9274 |
|        | sunset:6287 | sunse:0.024 | building:0.033 | building:0.034 | bay:0.034 | sunset:0.9310 |
| **CS** | sunse:0.14 | f-16:0.023 | bay:0.028 | city:0.017 | city:0.060 | bay:0.034 |
|        | birds:9006 | birds:6490 | city:0.017 | city:0.060 | bay:0.034 | sunset:9310 |
| **PD** | ships:0.000 | mountain:0.019 | birds:0.027 | building:0.008 | building:0.013 | mountain:12548 |
|        | birds:6681 | sunrise:0.011 | ships:0.023 | building:0.008 | building:0.013 | mountain:12548 |
| **CS** | face:0.000 | boeing:0.016 | building:0.008 | building:0.008 | building:0.008 | building:0.008 |
|        | face:8118 | mountain:10737 | building:7604 | building:7604 | building:7604 | building:7604 |
| **PD** | face:0.000 | face:7768 | building:7604 | building:7604 | building:7604 | building:7604 |
| **CS** | face:0.000 | face:7768 | building:7604 | building:7604 | building:7604 | building:7604 |
|        | face:11457 | face:7768 | building:7604 | building:7604 | building:7604 | building:7604 |
| **PD** | face:0.000 | face:7768 | building:7604 | building:7604 | building:7604 | building:7604 |
| **CS** | sky:0.000 | sky:0.000 | sky:0.000 | sky:0.000 | sky:0.000 | sky:0.000 |
|        | sky:11249 | sky:11249 | sky:11249 | sky:11249 | sky:11249 | sky:11249 |
| **PD** | sky:0.000 | sky:0.000 | sky:0.000 | sky:0.000 | sky:0.000 | sky:0.000 |
| **CS** | city:0.000 | city:0.000 | city:0.000 | city:0.000 | city:0.000 | city:0.000 |
|        | city:14118 | city:14118 | city:14118 | city:14118 | city:14118 | city:14118 |
| **PD** | city:0.000 | city:0.000 | city:0.000 | city:0.000 | city:0.000 | city:0.000 |
| **CS** | buildings:0.000 | buildings:0.000 | buildings:0.000 | buildings:0.000 | buildings:0.000 | buildings:0.000 |
|        | buildings:14118 | buildings:14118 | buildings:14118 | buildings:14118 | buildings:14118 | buildings:14118 |
| **PD** | buildings:0.000 | buildings:0.000 | buildings:0.000 | buildings:0.000 | buildings:0.000 | buildings:0.000 |
Table 8: [3/3] Comparison of the similarities of *ClassSim* (CS) computed by the one vs. rest (OVR) classifiers and parametric distance (PD). Column name represents the target class. The pairs of `{class : similarity}` are shown in descending order of the similarities for each column. CS is the similarity score ranging from 0 to 1 (higher values indicate greater similarity). PD is a positive real number (lower values indicate greater similarity).
## CS(OVR) and CS(multi-class) experiment

|       | bay     | beach    | birds   | boeing  | buildings | city     |
|-------|---------|----------|---------|---------|-----------|----------|
| OVR   | beach:0.626 | bay:0.246 | ocean:0.045 | f-16:0.258 | city:0.656 | buildings:0.656 |
|       | bay:0.626 | ocean:0.040 | face:0.011 | city:0.122 | bay:0.123 |
| multi | OVR     | city:0.301 | ocean:0.040 | face:0.037 | chopper:0.153 | bay:0.301 |
|       |      | mountain:0.093 | ocean:0.009 | ocean:0.009 | chopper:0.005 | bay:0.044 |
| OVR   | city:0.301 | mountain:0.114 | ocean:0.028 | ocean:0.067 | bay:0.069 |
|       |      | buildings:0.015 | ocean:0.008 | chopper:0.005 | bay:0.044 |
| multi | OVR     | mountain:0.188 | ocean:0.087 | f-16:0.027 | bay:0.151 |
|       |      | sunset:0.014 | sunset:0.106 | f-16:0.005 | bay:0.073 |
| OVR   | city:0.123 | buildings:0.005 | bay:0.059 | sunset:0.029 | bay:0.073 |
|       |      | city:0.301 | city:0.122 | buildings:0.005 | bay:0.073 |
| multi | OVR     | mountain:0.0093 | ocean:0.011 | sky:0.005 | ocean:0.011 |
|       |      | sunset:0.011 | sky:0.047 | city:0.035 | ocean:0.011 |
| OVR   | buildings:0.069 | city:0.073 | sunrise:0.009 | sky:0.004 | ocean:0.056 |
|       |      | sunrise:0.013 | sky:0.004 | city:0.035 | ocean:0.056 |
| multi | OVR     | sky:0.047 | sky:0.030 | birds:0.019 | ocean:0.056 |
|       |      | city:0.008 | sunset:0.004 | sky:0.004 | ocean:0.056 |
| OVR   | sunrise:0.056 | sky:0.030 | sunset:0.004 | birds:0.019 | ocean:0.056 |
|       |      | city:0.008 | sunset:0.004 | sky:0.004 | ocean:0.056 |
| multi | OVR     | sunset:0.055 | sky:0.030 | sky:0.004 | ocean:0.056 |
|       |      | buildings:0.007 | sky:0.004 | city:0.035 | ocean:0.056 |
| OVR   | sunset:0.055 | sky:0.030 | sky:0.004 | city:0.035 | ocean:0.056 |
|       |      | sky:0.004 | city:0.035 | sky:0.004 | ocean:0.056 |
| multi | OVR     | f-16:0.028 | f-16:0.023 | sky:0.004 | ocean:0.056 |
|       |      | birds:0.004 | sky:0.004 | city:0.035 | ocean:0.056 |
| OVR   | f-16:0.028 | birds:0.004 | sky:0.004 | city:0.035 | ocean:0.056 |
|       |      | mountains:0.011 | sky:0.004 | city:0.035 | ocean:0.056 |
| multi | OVR     | bushes:0.0099 | sky:0.004 | sky:0.004 | ocean:0.056 |
|       |      | mountains:0.011 | sky:0.004 | sky:0.004 | ocean:0.056 |
| OVR   | bushes:0.0099 | sky:0.004 | sky:0.004 | sky:0.004 | ocean:0.056 |
|       |      | mountains:0.011 | sky:0.004 | sky:0.004 | ocean:0.056 |
| multi | OVR     | face:0.007 | face:0.007 | face:0.007 | face:0.007 |
|       |      | face:0.007 | face:0.007 | face:0.007 | face:0.007 |
| OVR   | face:0.007 | face:0.007 | face:0.007 | face:0.007 | face:0.007 |
|       |      | face:0.007 | face:0.007 | face:0.007 | face:0.007 |
| multi | OVR     | face:0.007 | face:0.007 | face:0.007 | face:0.007 |
|       |      | face:0.007 | face:0.007 | face:0.007 | face:0.007 |
| OVR   | face:0.007 | face:0.007 | face:0.007 | face:0.007 | face:0.007 |
|       |      | face:0.007 | face:0.007 | face:0.007 | face:0.007 |

Table 9: [1/3] Comparison of the similarities of ClassSim (CS) computed by the one vs. rest (OVR) classifiers and those of CS computed by the multi-class (multi) classifier. Column name represents the target class. The pairs of {class : similarity} are shown in descending order of the similarities for each column. The similarity score ranges from 0 to 1 (higher values indicate greater similarity).
Table 10: [2/3] Comparison of the similarities of ClassSim (CS) computed by the one vs. rest (OVR) classifiers and those of CS computed by the multi-class (multi) classifier. Column name represents the target class. The pairs of \{\text{class} : \text{similarity}\} are shown in descending order of the similarities for each column. The similarity score ranges from 0 to 1 (higher values indicate greater similarity).
|               | ships      | sunset     | sunrise    | ocean      |
|---------------|------------|------------|------------|------------|
| **OVR**       | f-16:0.126 | sunrise:0.902 | sunset:0.902 | bay:0.320  |
|               | bay:0.061  | sunrise:0.353 | sunset:0.353 | bay:0.087  |
| **OVR**       | ocean:0.108 | sky:0.317  | sky:0.302  | sky:0.271  |
|               | helicopter:0.025 | sky:0.106 | sky:0.057  | clouds:0.041 |
| **OVR**       | helicopter:0.098 | ocean:0.163 | ocean:0.157 | clouds:0.260 |
|               | ocean:0.022 | bay:0.020  | bay:0.020  | beach:0.040 |
| **OVR**       | city:0.097  | clouds:0.128 | clouds:0.116 | beach:0.245 |
|               | buildings:0.017 | ocean:0.014 | ocean:0.014 | sunset:0.026 |
| **OVR**       | buildings:0.092 | beach:0.095 | beach:0.106 | sunset:0.163 |
|               | f-16:0.013  | beach:0.014 | beach:0.009 | sky:0.022  |
| **OVR**       | bay:0.077   | bay:0.056  | bay:0.106  | sunset:0.163 |
|               | city:0.013  | bay:0.013  | bay:0.005  | sky:0.022  |
| **OVR**       | boeing:0.059 | mountain:0.046 | mountain:0.093 | sunrise:0.157 |
|               | beach:0.007 | bay:0.013  | bay:0.055  | ships:0.108 |
| **OVR**       | beach:0.023 | city:0.038 | city:0.008 | mountain:0.016 |
|               | birds:0.004 | city:0.008 | city:0.008 |               |
| **OVR**       | sunrise:0.022 | face:0.008  | face:0.005  |               |
|               | sunset:0.000 | face:0.040  | face:0.004  |               |
| **OVR**       | mountain:0.020 | buildings:0.029 | buildings:0.026 | city:0.056 |
|               | sunrise:0.000 | helicopter:0.004 | buildings:0.000 | face:0.012 |
| **OVR**       | face:0.016  | birds:0.028 | ships:0.022 | face:0.051  |
|               | sky:0.000   | birds:0.044 | helicopter:0.000 | f-16:0.009 |
| **OVR**       | birds:0.008  | face:0.024  | ships:0.000 | f-16:0.016  |
|               | mountain:0.000 | face:0.000  | f-16:0.000  | f-16:0.050  |
| **OVR**       | sunset:0.004 | ships:0.004 | birds:0.013 | birds:0.045 |
|               | face:0.000   | f-16:0.000  | city:0.000  | helicopter:0.000 |
| **OVR**       | sky:0.000   | helicopter:0.004 | helicopter:0.011 | buildings:0.025 |
|               | clouds:0.000 | buildings:0.000 | boeing:0.000 | city:0.000  |
| **OVR**       | clouds:0.000 | boeing:0.004 | boeing:0.000 | helicopter:0.004 |
|               | boeing:0.000 | boeing:0.000 | boeing:0.000 | boeing:0.000 |

Table 11: Comparison of the similarities of ClassSim (CS) computed by the one vs. rest (OVR) classifiers and those of CS computed by the multi-class (multi) classifier. Column name represents the target class. The pairs of \( \text{class : similarity} \) are shown in descending order of the similarities for each column. The similarity score ranges from 0 to 1 (higher values indicate greater similarity).
References

[Bell and Bala, 2015] Sean Bell and Kavita Bala. Learning visual similarity for product design with convolutional neural networks. ACM Transactions on Graphics (TOG), 34(4):98, 2015.

[Bendale and Boult, 2015] Abhijit Bendale and Terrance Boult. Towards open world recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1893–1902, 2015.

[Bendale and Boult, 2016] Abhijit Bendale and Terrance Boult. Towards open set deep networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1563–1572, 2016.

[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 Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pages 248–255. IEEE, 2009.

[Deselaers and Ferrari, 2011] Thomas Deselaers and Vittorio Ferrari. Visual and semantic similarity in imagenet. In Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, pages 1777–1784. IEEE, 2011.

[Goodfellow et al., 2014] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572, 2014.

[Guan et al., 2009] Genliang Guan, Zhiyong Wang, Qi Tian, and Dagan Feng. Improved concept similarity measuring in the visual domain. In Multimedia Signal Processing, 2009. MMSP’09. IEEE International Workshop on, pages 1–6. IEEE, 2009.

[Han et al., 2015] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3279–3286, 2015.

[Izadinia et al., 2015] Hamid Izadinia, Bryan C Russell, Ali Farhadi, Matthew D Hoffman, and Aaron Hertzmann. Deep classifiers from image tags in the wild. In Proceedings of the 2015 Workshop on Community-Organized Multimodal Mining: Opportunities for Novel Solutions, pages 13–18. ACM, 2015.

[Li and Snoek, 2013] Xirong Li and Cees GM Snoek. Classifying tag relevance with relevant positive and negative examples. In Proceedings of the 21st ACM international conference on Multimedia, pages 485–488. ACM, 2013.

[Nguyen et al., 2015] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 427–436, 2015.

[Russakovsky et al., 2015] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3):211–252, 2015.

[Scheirer et al., 2014] Walter J Scheirer, Lalit P Jain, and Terrance E Boult. Probability models for open set recognition. IEEE transactions on pattern analysis and machine intelligence, 36(11):2317–2324, 2014.

[Schölkopf et al., 2001] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. Neural computation, 13(7):1443–1471, 2001.

[Schooff et al., 2015] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 815–823, 2015.

[Szegedy et al., 2016] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2818–2826, 2016.

[Wang et al., 2008] Zhiyong Wang, Genliang Guan, Jiajun Wang, and Dagan Feng. Measuring semantic similarity between concepts in visual domain. In Multimedia Signal Processing, 2008 IEEE 10th Workshop on, pages 628–633. IEEE, 2008.

[Wang et al., 2014] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1386–1393, 2014.

[Wu et al., 2013] Pengcheng Wu, Steven CH Hoi, Hao Xia, Peilin Zhao, Dayong Wang, and Chunyan Miao. Online multimodal deep similarity learning with application to image retrieval. In Proceedings of the 21st ACM international conference on Multimedia, pages 153–162. ACM, 2013.

[Yi et al., 2014] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Deep metric learning for person re-identification. In Pattern Recognition (ICPR), 2014 22nd International Conference on, pages 34–39. IEEE, 2014.