Predicting Failures of Vision Systems

Peng Zhang[1] Jiuling Wang[2] Ali Farhadi[3] Martial Hebert[4] Devi Parikh[1]
[1]Virginia Tech [2]Univ. of Texas at Austin [3]Univ. of Washington [4]Carnegie Mellon University
[1]{zhangp, parikh}@vt.edu [2]jiuling@utexas.edu [3]ali@cs.uw.edu [4]hebert@ri.cmu.edu

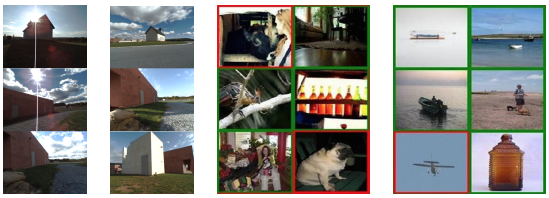
Abstract

Computer vision systems today fail frequently. They also fail abruptly without warning or explanation. Alleviating the former has been the primary focus of the community. In this work, we hope to draw the community's attention to the latter, which is arguably equally problematic for real applications. We promote two metrics to evaluate failure prediction. We show that a surprisingly straightforward and general approach, that we call ALERT*, can predict the likely accuracy (or failure) of a variety of computer vision systems – semantic segmentation, vanishing point and camera parameter estimation, and image memorability prediction – on individual input images. We also explore attribute prediction, where classifiers are typically meant to generalize to new unseen categories. We show that* ALERT *can be useful in predicting failures of this transfer. Finally, we leverage* ALERT *to improve the performance of a downstream application of attribute prediction: zero-shot learning. We show that* ALERT *can outperform several strong baselines for zero-shot learning on four datasets.*

1. Introduction

Computer vision systems today are not perfect. Unfortunately, given the ambiguous nature of many vision problems, they will *never* be perfect. Our community's primary focus has been on making these systems – let's call them BASESYS[1] – more and more accurate. To encourage systematic progress, we have established benchmarks like Caltech 101 [16], PASCAL [14], SUN [60], etc. and we strive to minimize the failures of BASESYS on these benchmarks.

Computer vision as part of a system: It is crucial to keep in mind that in the real world, many applications involve *pipelines*, where the output of one system is fed into another as input. For instance, models trained to classify local image patches may be fed into probabilistic models for semantic segmentation [51]. Semantic segmentation may be fed to a robot's path planning algorithm for navigating through its environment [41]. Estimates of vanishing points in an image may be fed into a 3D layout estimation algorithm [23, 46]. Attribute predictions may be fed to recognition systems to classify previously unseen object categories [15, 31]. The subsequent "system" may even be a user: an image memorability predictor [25] may be used by a graphics designer.



(a) (b) (c) (d)

Figure 1: Qualitative results of our proposed approach (ALERT) on two segmentation datasets: (a) and (c) show images predicted by ALERT as unreliable poor performing images; (b) and (d) show images predicted to be the best performing. For a ground robot (a,b), ALERT correctly flagged the images that are corrupted with strong shadows and over/under-exposure while retaining the images with good contrast. For PASCAL segmentation (c,d), the images for which BASESYS [7] generated poor segmentations are marked by a red border. ALERT clearly favored images with easy to segment isolated objects over complex images containing a large number of small regions. See author's webpage for example segmentations.

In all these cases, it is of course desirable for BASESYS to fail infrequently. But arguably, it is equally desirable for BASESYS to fail *gracefully*, instead of failing abruptly without warning. While minimizing failures has been the primary focus of the community, embracing and effectively dealing with the failures has been mostly ignored. For a variety of reasons such as verifiable safety standards for autonomous systems, there is a recent push towards BASESYS that can explain their outputs and are interpretable. In this paper we take a small step in this direction.

We push for a capability where BASESYS can generate a warning "I am unable to make a reliable decision for this input". Not all applications may have the luxury of taking advantage of such a warning. We argue that *many* do. For instance, if BASESYS issues a warning that it is unlikely to be able to accurately predict the presence of certain attributes in an image, the recognition system downstream can choose to ignore these unreliable attributes, and only use the reliable ones. This will make the overall recognition pipeline more robust as compared to one that accepts all attribute predictions from BASESYS at face value and propagates the errors. Consider a robot segmenting every frame in the video feed during a field experiment, where

[1]BASESYS may be a segmentation engine, an attribute predictor, a vanishing point estimator, an iPhone app that predicts aesthetic quality, etc.

it often encounters poor quality images. If BASESYS can identify such frames where the segmentation output is likely to be inaccurate (See Figure 1(a)), the robot can choose to skip those frames all together, thus saving computational time, and simply wait for a later, more reliable frame. Depending on the application, a delayed but accurate decision may be preferred over a timely inaccurate decision, which may be catastrophic. If the user of an app can be warned that the output cannot be trusted for a particular input, (s)he can make an informed decision. In image search if a user is looking for "black high-heeled shiny shoes", it might be better to not return images where the "high-heel" predictor can not be trusted, and only return images where *all three* attributes can be reliably detected. Generally, the resultant higher precision may be worth the possibly lower recall.

How to detect failures? There are two ways to equip a BASESYS with such a warning capability. First, we could analyze the output of the vision system to assess its confidence on any given input. In this case, the confidence evaluation has to be designed specifically for each system. Instead, we explore the alternative approach of evaluating the *input* itself in order to assess BASESYS's confidence. The main thesis of our approach is that while BASESYS may not be able to reliably predict the correct output for a certain input, predicting the *difficulty* or *ambiguity* of the input may still be feasible. A benefit of this approach is that it is applicable to *any* vision system because the reliability estimate is generated based on the input alone, without ever looking at the inner workings of BASESYS. This approach has the added benefit of not having to run BASESYS – typically computationally expensive – to assess its confidence. Instead it can quickly reject inputs that are not reliable to begin with. Finally, failure patterns may be different across domains. Designers of BASESYS can not foresee all domains in which it will be used. Hence, any confidence estimation or early-rejection mechanisms built into BASESYS may not generalize. Our approach, that we call ALERT, provides a simple tool for the *consumer* of BASESYS to train a failure predictor catered to his domain.

Why is it feasible? The idea of analyzing the input to classification and detection algorithms to assess its suitability is not new and has been around in a variety of application-driven communities such as Automatic Target Recognition (ATR) [32, 59] and biometrics [20, 53]. In these signal and image processing fields, the distinction between errors due to inherent ambiguities in the feature space and errors due to corrupted input becomes critical. While both may be difficult to recover from, the latter can at least be detected. As the vision community inches closer to having real-world applications and away from hand-curated datasets such as Caltech 101 [16] or PASCAL [14], we argue that recognizing this distinction becomes critical in our community too. Note that, while analyzing performances of algorithms as a function of biases in these datasets has been addressed [28, 54], we are advocating the distinct task of *predicting* the performance of a system on a given input. With this work we hope to bring the community's attention to building self-evaluating systems that can reliably predict their own failures.

A valid concern a reader may have is that if a system that can reliably predict failures of BASESYS can be trained, does that not mean BASESYS could have already been trained to be better? Assuming BASESYS has been trained well, should it not be impossible to train ALERT with accuracy better than chance?[2] We argue that this reasoning is not applicable to a wide range of vision systems. First, many BASESYS are not learning-based approaches in the first place (e.g., vanishing point estimation as in [33]). Among those that are, applications where the output space is very large (e.g., exponential for segmentation or any structured output task), ALERT identifying that the label predicted by BASESYS is incorrect provides little information about what the correct label in fact is. Further, the features used to train BASESYS (e.g., local image patches for segmentation) may be complementary to the features used to train ALERT (e.g., image-level gist). It is not trivial to incorporate these features (e.g., gist) into BASESYS itself (e.g., vanishing point estimation or segmentation). Hence, generally speaking, research efforts towards systems such as ALERT are orthogonal to efforts towards improving BASESYS. For applications such as attribute-predictors where BASESYS has a small label-space (binary) ALERT may boil down to identifying images where the response can not be trusted (e.g., gray-scale images for predicting the attribute red), and not images where the response can be confidently classified as being wrong. This is a subtle but important difference. While a "good" classifier should ideally have reliable confidence estimates, most existing training procedures do not explicitly enforce this. They optimize for assigning correct labels to training images, and not for the classifier's confidence being correlated with likelihood of failure. ALERT provides a way to deal with the fact that discriminative classifiers tend to be overconfident.

In order to demonstrate the broad applicability of our simple but surprisingly effective approach we evaluate it on four diverse state-of-the-art BASESYS: attribute predictors (4 datasets), semantic segmentors (2 datasets), vanishing point estimators (2 datasets) and image memorability predictor (1 dataset). In all these cases, we show that ALERT can predict the accuracy of these diverse systems with significant reliability: ALERT improves the accuracy of BASESYS by allowing it to make fewer, but more accurate decisions. We introduce two metrics for evaluating failure prediction approaches, and show that ALERT outperforms state-of-the-art systems in terms of these metrics. Finally, we use ALERT on attribute-predictors on 4 datasets and outperform strong baselines at conventional classifica-

[2] See [43] for a formal argument on "biometric-completeness".

tion accuracies on a downstream task: zero-shot learning.

2. Related Work

Our work addresses an issue that is critical for real applications and has received attention in various communities.

Meta-recognition: Inspired by meta-cognition "knowing about knowing" [18], Scheirer *et al*. [50] coined the term "meta-recognition" for performance prediction methods that analyze post-recognition scores [56]. These methods have mainly been explored in biometrics with the recent exception of [50]. The analysis proceeds by examining the output of a matching system on a test instance. This output often consists of the test instance's similarity score to all instances in the dataset. The analysis can be used to automatically determine thresholds to make match / non-match decisions [50], intelligent fusion schemes [47, 48], etc. ALERT differs from this line of work in two prominent ways (1) ALERT does not require the output of BASESYS on a test instance to predict its performance – it only uses the input test instance itself. This is particularly desirable when BASESYS is computationally expensive. (2) Methods such as [49, 50] are only applicable to BASESYS that rely on similarity scores for recognition. Many computer vision systems (such as segmentation, vanishing point estimation, etc.) do not fall in this category. ALERT is broadly applicable to all such applications. In computer vision, Welinder *et al*. [58] and Aghazadeh and Carslsson [1] predict the global performance of a classifier on a corpus of test instances. ALERT instead provides an instance-specific reliability measure.

Image quality assessment: The biometrics community has been using image quality measures as predictors for matching performance [20, 53]. In this context, the concept of input "quality" is closely tied to BASESYS – poor quality simply refers to the inability of BASESYS to provide accurate results for that input. Our work follows the same philosophy. Such methods, as do we, only analyze the input instance (e.g., fingerprint scan) to assess the likely quality of match. We argue that a similar level of self-evaluation should be part of all computer vision systems as they are involved in a variety of real-world applications.

Predicting failures: Optimization algorithms can often detect when they have found the global optimum [52] or can indicate for which variables they are failing to find optimal assignments [22, 44]. In computer vision, Jammalamadaka *et al*. [26] recently introduced evaluator algorithms for human pose estimators (HPE). They used features specific to this application; ALERT on the other hand uses generic image appearance features for a wide variety of applications. While application-specific features can only boost ALERT's performance, in this paper we hope to promote this line of work by demonstrating the ease with which one can build self-evaluating systems. Aodha *et al*. [37] trained a classifier that can select one of several optical flow algorithms for a given input video sequence, which resulted in improved performance over any one algorithm alone. Liao *et al*. [36] use application-specific features to train a classifier that can predict whether their region proposal algorithm would fail on an image or not. Hoiem *et al*. [24] focus on analyzing the different sources of error in object detectors, and do not predict failure. Methods that predict performance by analyzing statistics of the training *and test* data [4, 57] are not applicable to our goal of predicting the reliability of individual test instances. Detecting errors has received a lot of attention in speech recognition [8, 45]. Recovering from these errors in spoken dialogue systems is often interactive [17]. In a similar spirit, a system like ALERT can be used to actively elicit labels for instances likely to be misclassified as in [3] to improve performance. KWIK "Knows What It Knows" frameworks [35] allow for active exploration which is beneficial in reinforcement- and active-learning problems.

Estimating confidence of classifiers: The confidence of a classifier in its decision is often correlated to the likelihood of it being correct. Reliably estimating the confidence of classifiers has received a lot of attention in the pattern recognition community [13, 29, 39]. Applications such as spam-filtering [9], natural language processing [2, 12], speech [27] and even computer vision [61] have leveraged these ideas. However, different classifiers require different methods to estimate their confidences, making their general applicability limited. Moreover, many computer vision systems (e.g., vanishing point estimators) are not classifiers. Hence, system- and application-specific means of estimating the confidence of BASESYS would be required. For instance, Haeusler *et al*. [21] use an ensemble of classifiers to combine different stereo matching confidence estimates. ALERT is BASESYS independent, making it broadly applicable. Methods that use cross-validation to select a robust model from a pool of models [5] completely discard unreliable models from being used in the future. ALERT makes local instance-level decisions about whether BASESYS can be trusted for *that* instance or not.

Rejection ("none-of-the-above"): Refusing to make a decision on an instance is related to systems that identify an instance as belonging to none of the classes seen during training [11, 34]. The scenario we consider is not concerned with unfamiliar instances arising from discrepancies in the distributions of the training and test data. It is concerned with familiar but difficult instances. Gao and Koller [19] learn a relaxed hierarchy of binary classifiers, where the difficult categories may be ignored early on and are dealt with further down in the hierarchy. This allows them to trade-off speed for accuracy. The work of Deng *et al*. [10] makes decisions in a hierarchical fashion. Given a semantic hierarchy on the categories of interest (e.g., WordNet [38]), their approach trades off specificity for accuracy. They output the most specific label they can while being confident about

it. ALERT is complementary to these approaches and trades off the declaration rate of BASESYS for its accuracy.

3. Approach Overview

Given a vision system BASESYS, we wish to learn ALERT: a model that will detect with some reliability whether BASESYS is likely to fail on a given test instance.

As training data, we are given N instances $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_N\}$ and the corresponding accuracies (or measure of error) $\{y_1, \ldots, y_i, \ldots, y_N\}$ of BASESYS on these instances. These instances $\boldsymbol{x}_i$ may be images, video frames, entire videos or outputs of any sensing modality – whatever it is that BASESYS takes as input. The accuracy (or error) $\{y_i\}$ may be any metric that is appropriate to evaluate the output of BASESYS for a given input instance $\boldsymbol{x}_i$. In this work we only consider scenarios where y_i is a scalar, but extensions to vector y_i are conceivable. If BASESYS is a classifier that predicts a binary label for a given input instance, y_i may be the 0-1 loss. If BASESYS produces a semantic segmentation for an image $\boldsymbol{x}_i$, y_i may be the proportion of pixels in $\boldsymbol{x}_i$ correctly assigned to their ground truth label. We represent each instance $\boldsymbol{x}_i$ with $d-$dimensional appearance features: $\boldsymbol{x}_i \in \mathcal{R}^d$ (overloading notation). These appearance features may be any features that are appropriate for the modality of $\boldsymbol{x}_i$. We experiment with ALERT simply being a discriminatively trained classifier (SVM) when y_i is binary (for instance when BASESYS produces binary labels for $\boldsymbol{x}_i$), or a regressor (Support Vector Regressor) when y_i is a continuous scalar. Note that any sophisticated image descriptors, machine learning techniques and application- or BASESYS-specific features can be incorporated in ALERT. If one anticipates specific failure modes of BASESYS given knowledge about its inner workings, one could design features that specifically represent these failure modes.

BASESCORE: BASESYS often produce indicators of their confidence in their output e.g., the energy of a conditional random field model for semantic segmentation or the number of lines detected in an image that are poorly explained by the estimated vanishing points. We call such measures BASESCORE. Not only is BASESCORE BASESYS-specific (unlike ALERT), it also requires us to run BASESYS at test time – which may be computationally expensive – and obtain its output before we can estimate the confidence. Our approach needs to run BASESYS only to compute y_i to train ALERT. At test time, ALERT does not need to run BASESYS to estimate reliability of the input instance. BASESCORE may be quite reliable since it is aware of the inner workings of BASESYS. ALERT on the other hand has the benefit of capturing potentially orthogonal image features. Which one performs better may depend on the application. But note that BASESCORE is less general than ALERT and not preemptive – it does not allow for early rejection of unreliable input instances. If early rejection is not critical for an application, combining ALERT with BASESCORE (which involves running BASESYS) can yield a better failure predictor than both.

In Section 4 we describe how we use ALERT to avoid failures in a variety of applications. In Section 5 we use ALERT to predict the failure of attribute predictors at transferring knowledge to previously unseen categories. This allows us to improve the performance of a downstream application (zero-shot learning [31]).

4. Avoiding Failures

We applied our approach to three diverse applications. For all three, ALERT was trained by combining multiple kernels for 14 generic image features such as dense SIFT, color, texton, gist, HOG, line histograms, local binary patterns, self-similarity and so on, using the implementation of [60]. In other words, the feature representation for $\boldsymbol{x}_i$ is the same for all three applications. Note that these can capture image quality [20, 53] in addition to other visual properties. We use half the images to train ALERT and the other half to test. We now provide the specific details of BASESYS, y_i and BASESCORE. Quantitative results on these applications are in Section 4.4.

4.1. Semantic Segmentation

We explored semantic segmentation in two domains.

Robot: This first dataset contains images collected by a robot. The original 1278x958 images were scaled and rectified to 320x240. Each pixel was assigned to one of 8 semantic labels (sky, tree, asphalt, grass, building, object, concrete and gravel) using the semantic segmentation algorithm of Munoz *et al*. [40] as BASESYS, which constructs a hierarchical segmentation of the image and predicts a distribution of labels for each segment by combining, at each level of the hierarchy, image features with the distribution of labels predicted at the previous level. We experimented with two different accuracy measures y_i. The first is the proportion of pixels assigned to their correct labels and the second is the proportion of pixels from each class assigned to their correct labels averaged across the classes. While the first measure favors classes that occur more frequently in images, this second measure normalizes for the class distribution. BASESYS provides a distribution over all the labels for each pixel in the input image. For BASESCORE, we compute the entropy of this distribution, and averaged it across all pixels in an image.

PASCAL: The second scenario is the PASCAL VOC 2012 segmentation challenge where each pixel in a diverse set of indoor and outdoor images is to be assigned to one of 20 semantic categories (person, chair, etc.) or background. As BASESYS we use the state-of-the-art approach of Carreira and Sminchisescu [6, 7]. It involves generating a large number of plausible object segment hypotheses using constrained parametric min-cuts and bottom up cues, followed

Figure 2: Qualitative examples for the camera rotation matrix estimation task. Left: images predicted as poor performing images by ALERT. We see that the three dominant directions are often not clearly visible (e.g., ceiling cropped out of the picture). Right: images predicted as best performing images by ALERT. The 3D orientation of the room is clearly evident.

by re-ranking them using mid-level cues. This re-ranking is done by training class-specific regressors which provide a high score if a segment is a good match for the category. In addition to the two accuracy metrics described above we also experiment with the standard PASCAL $\frac{\text{intersection}}{\text{union}}$ metric. As BASESCORE, we average the score assigned to each segment in an image by the class-specific regressor of BASESYS corresponding to the label that the segment took in the segmentation. Figure 1 shows qualitative results.

4.2. Vanishing Point Estimation

We explored vanishing point estimation (VP) on 301 indoor scene images in Hedau *et al*. [23]'s dataset, and camera parameter (focal length and rotation) estimation on Satkin *et al*. [46]'s dataset containing 353 bedroom and 166 living room images from the SUN [60] dataset. BASESYS involves detecting lines in image $\boldsymbol{x}_i$ and clustering them in 3 directions (and outliers) [33] to estimate the VPs. Under the Manhattan world assumption, the VP estimates and the orthogonality constraint are used to compute the intrinsic camera parameters and the camera rotation matrix. The error e in each VP estimated in image $\boldsymbol{x}_i$ is measured as its distance from the lines associated with the corresponding ground truth VP. We used several measures for y_i: the minimum of the three e's, their mean, the sum of the two smallest errors, the difference between the estimated and ground truth focal length relative to the true focal length, and the error between the ground truth camera rotation matrix $R(\boldsymbol{x}_i)$ and that estimated by BASESYS is $\hat{R}(\boldsymbol{x}_i)$ measured as the geodesic distance on the 3D manifold of rotation matrices:

$$y_i = \frac{1}{\sqrt{(2)}} ||\log(R(\boldsymbol{x}_i)^T \hat{R}(\boldsymbol{x}_i))||_F. \quad (1)$$

As BASESCORE for VP estimation, we use the proportion of lines detected in the image that were assigned to the "outliers" cluster when estimating VPs. There is no natural BASESCORE for camera rotation and focal length estimation. See Figure 2 for qualitative results.

4.3. Image Memorability Prediction

We explored image memorability prediction, where the task is to predict how memorable an input image is [25] in the $[0, 1]$ range. We use the dataset of Isola *et al*. [25] containing 2222 images. As BASESYS we use their approach that trains a Support Vector Regressor using state-of-the-art image features such as HOG, gist, SIFT, self-similarity and pixel histograms. We computed the error y_i in the prediction as the relative difference between the score predicted by BASESYS and the ground truth memorability score.[3] Since BASESYS is a regressor, there is no natural definition of BASESCORE for this application.

4.4. Evaluation

We evaluate ALERT using two different metrics.

ADR: The first is an Accuracy of BASESYS vs. Declaration Rate (ADR) curve. Declaration Rate is the proportion of test images on which BASESYS does not output a decision, in fear of providing an incorrect decision. This curve is computed by sorting the test images in descending order of their reliability as estimated by ALERT. We then retain only a DR proportion of the test images (DR $\in [0, 1]$), and discard the rest. We compute the accuracy of BASESYS on these retained images and plot accuracy vs. DR. For some applications like vanishing point estimation, it is more natural to use error instead of accuracy. If ALERT were perfect, it would discard the worst performing images. Accuracy would be very high at low DR and then fall gracefully as DR approached 1. If ALERT performed at chance level, on average, the accuracy would remain constant with varying DR. We compare the performance of ALERT to these upper and lower bounds (Figure 3). We see that even an approach as straightforward as ALERT can perform significantly better than chance. As expected, BASESYS-specific BASESCORE often performs better. See author's webpage for examples where ALERT outperforms BASESCORE. A simple summation of ALERT and BASESCORE improves performance of both. Clearly, ALERT captures information orthogonal to BASESYS's beliefs. Some BASESYS may have more systematic failure modes than others, hence ALERT helps in some cases more than others. BASESYS for image memorability prediction included many of the same features that ALERT was trained on. ALERT still predicts its failures reliably, but not as dramatically as in other applications.

RAM: Our second metric evaluates ALERT's ability to trade-off the risk of making an incorrect decision with not making a decision at all. This Risk-Averse Metric (RAM) gives BASESYS +1.0 points for a correct answer, -0.5 points for an incorrect answer and 0 points if it makes no decision.[4] We believe such a metric is crucial when dealing with real applications. By refusing to make a decision when ALERT raises a warning, we expect BASESYS to gain more points by trading off incorrect decisions for no decisions. In applications where y_i is a continuous scalar accuracy measure,

[3] ALERT regresses to this error. This can be thought of as analogous to one round of L2 boosting, which involves regressing to the error of a regressor. Also relevant is the notion of twicing in statistics [55].

[4] In fact, the ImageClef Robot Vision Challenge http://www.imageclef.org/2013/robot uses such a Risk-Averse Metric for recognition. This is similar to, but distinct from applications where false positives are less expensive than false negatives e.g., pedestrian detection.

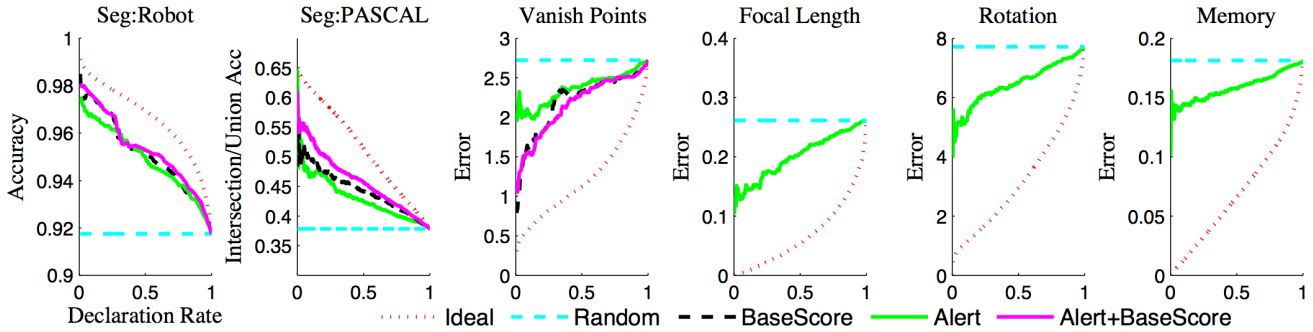


Figure 3: Accuracy or Error of BASESYS vs. Declaration Rate Curves. We used accuracy and intersection/union for evaluating segmentation (robot and PASCAL respectively), and the mean of three errors for vanishing point estimation. Results for other choices of y_i described in Section 3 can be found on author's webpage. Results are averaged over 10 random train/test splits. BASESCORE is not shown for focal length, rotation and memorability prediction because the first two are direct products of vanishing point estimates, and the inner workings of BASESYS for the latter do not lead to an obvious definition for BASESCORE. Best viewed in color.

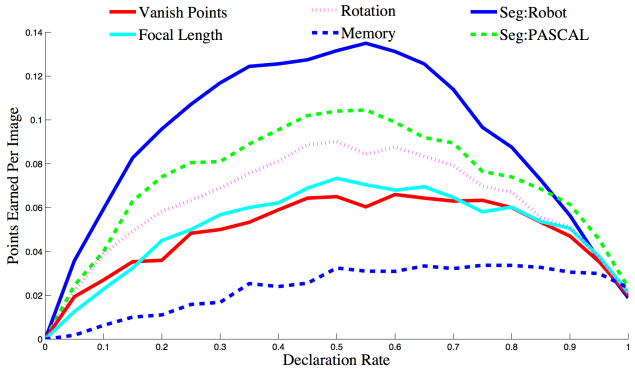


Figure 4: Risk-averse Metric: Points achieved by BASESYS when equipped with ALERT at varying declaration rates (DR). ALERT can help earn more points than the default strategy of always making a decision (DR = 1) or the other extreme of always refusing to make decisions (DR = 0).

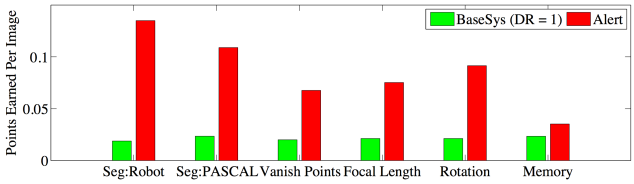


Figure 5: Risk-averse Metric: Summary of Figure 4. Points achieved by ALERT by picking the appropriate DR as compared to BASESYS.

how high does y_i have to be for a decision to be considered to be "correct"? E.g., how good does a segmentation have to be for it to be "correct"? Different applications may place different thresholds T on y_i to define "good enough" (correct) vs. incorrect. If T is high (i.e. most images are incorrect), we would expect ALERT to help a lot. If T is low (i.e. most images are correct), ALERT will not be beneficial and BASESYS may as well make decisions on all images because it is likely to get them right anyway. Across the different applications, we pick a value of T such that 65% of the images are considered to be correctly processed. This is an (perhaps optimistic) assessment of how well current vision systems work. In Figure 4, we plot the average points (per test image) gained by BASESYS when equipped with ALERT. We see in all cases, ALERT can help BASESYS gain more points than it would if it did not use ALERT (i.e. DR = 1.0). Operating at the optimum DR (determined via cross validation) can allow BASESYS to gain as many as 7× the points it would without ALERT (Figure 5).

To understand the intervals in which ALERT is more helpful, let us consider the cost of an incorrect decision ($-C$) relative to that of a correct decision (1.0). In our experiments so far, we used $C = 0.5$ as described earlier. If C is very low or very high, ALERT would have to be very accurate in predicting failures for BASESYS to benefit from it. Otherwise, DR = 1 and DR = 0 would be optimum for low C and high C respectively. For instance, if C is very high, even one mistake undetected by ALERT can reduce the total points earned by BASESYS to a large negative value, and BASESYS would be better off not making any decisions at all (DR = 0). The quality of ALERT can be assessed by the range in values of C for which it is still beneficial. In our experiments, we find that ALERT can help BASESYS for C in $[0.14, 9]$, $[0.14, 2.34]$, $[0.41, 0.67]$, $[0.12, 0.82]$, $[0.16, 1.86]$ and $[0.16, 1.5]$ for the 6 applications in Figure 5. Since the optimum DR is to be determined via cross validation, while unable to boost performance outside these ranges, ALERT would default to DR = 0 or 1 and not hurt performance.

5. Benefiting A Downstream Application

Visual attributes are mid-level semantic image properties that are considered to be shareable across categories. Hence, attribute predictors are often used to transfer information from one set of categories to a previously unseen set. This involves transfer of knowledge. ALERT can be used to predict the potential failure of BASESYS in transferring knowledge. This can in turn be used to benefit downstream applications, such as zero-shot learning [31].

We experimented with attribute predictors in four domains: animals, faces, scenes and objects. We use the Animals With Attributes (AWA) dataset of Lampert *et al.* [31] containing 8609 images, the Public Figures Face Database (PubFig) of Kumar *et al.* [30] containing 42461 images,

Figure 6: Qualitative results for the "wearing sunglasses" attribute on PubFig. Left: Images predicted by ALERT as poor performing. Many of these have poor lighting and shadows especially in the eye region. Right: images predicted by ALERT as high performing. These are all taken under good lighting conditions. The images for which the BASESYS attribute predictor generated the incorrect output are marked by a red border.

7160 images from the SUN Attribute Database (SUN) of Patterson and Hays [42] and the aPascal+aYahoo (UIUC) dataset of Farhadi *et al*. [15] containing 8999 images. They contain 85, 73, 102 and 64 attributes respectively. We collected ground truth attribute annotations for PubFig (not available with dataset) using Amazon Mechanical Turk.

As BASESYS we use the attribute predictors provided by the authors for AWA, PubFig and UIUC. For SUN, we use the authors' code to train the predictors. Each predictor is a binary classifier converted to a probabilistic estimate $\pi(\boldsymbol{x}_i)$. We use the 0-1 loss as y_i. If the attribute predictor is sure of the attribute presence/absence, but is wrong, its decision is considered to be a mistake. If it is sure and right, or unsure of its decision, its decision is not a mistake:

$$y_i = \begin{cases} 0, & \text{if } \hat{l}\left(\boldsymbol{x}_i\right) = l\left(\boldsymbol{x}_i\right) \\ 0, & \text{if } \pi(\boldsymbol{x}_i) \in [\eta_1, \eta_2] \\ 1, & \text{otherwise} \end{cases} \tag{2}$$

where $l\left(\boldsymbol{x}_i\right)$ is the true binary label of $\boldsymbol{x}_i$ and $\hat{l}\left(\boldsymbol{x}_i\right)$ is the label predicted by BASESYS. We set η_1 and η_2 to 0.25 and 0.75 respectively in all our experiments. We train one ALERT for every attribute. The threshold on the score beyond which ALERT raises a warning was chosen via cross validation. Since we already have trained attribute predictors, we use their outputs as image features to train ALERT. We train ALERT on a held out set of unseen categories, and test it on a disjoint set of categories.[5] Qualitative results are in Figure 6. ADR curves similar to Figure 3 are shown on author's webpage. ALERT significantly outperforms BASESCORE in most datasets, because ALERT is explicitly trained to detect attribute classifiers' failures to generalize to unseen categories – which BASESCORE is typically not aware of. We now demonstrate how a downstream application that uses outputs of attribute predictors as input can benefit from such an ALERT.

Zero-shot Learning: Zero-shot learning [31] involves learning novel categories from their binary attributes-based descriptions (e.g., zebras are stripped, black and white, have four legs, etc.). Each category c is described with a list of M attributes $[a_1, \ldots, a_M]$, $a_m \in \{0, 1\}$. The probability of an instance $\boldsymbol{x}$ belonging to class c is

$$p(c|\boldsymbol{x}) \propto \prod_{m=1}^{M} p(a_m|\boldsymbol{x}), \tag{3}$$

[5]See author's webpage for details on train/val/test splits.

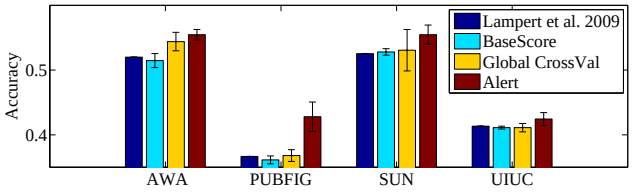


Figure 7: Zero-shot learning results.

where $p(a_m = 1|\boldsymbol{x}) = \pi_m(\boldsymbol{x})$. These attribute predictors are trained on a set of previously seen categories, which are disjoint from the set of unseen test categories. The attribute predictors are expected to generalize across this domain transfer. The test instance $\boldsymbol{x}$ is assigned the unseen class with the highest probability. The standard approach [31] uses all M attributes in Equation 3 ignoring the variability in the reliability of each attribute across input images. Instead, when using ALERT, for each test image we ignore the attributes for which ALERT raises a warning.[6] The same set of attributes is ignored for all classes, so the probabilities $p(c|\boldsymbol{x})$ are comparable. Note that different attributes are ignored for different test images.

We report zero-shot learning results in Figure 7. In addition to comparing to the standard approach of using all attributes [31], we compare to two other baselines. They both ignore attributes as needed, but by using other strategies than ALERT. We set the parameters of both baselines so that they ignore on average the same number of attributes per image as ALERT[7] making comparisons fair. The first baseline (BASESCORE) ignores attributes that are not confident (i.e. $\pi_m(\boldsymbol{x}_i)$ is close to 0.5). The second baseline (cross-validation) computes the accuracies of all attributes on a validation set, and ignores the least accurate ones (similar in spirit to how poselets are selected in [5]). Note that this baseline ignores the same attributes for all images, while ALERT adapts its decisions to individual input instances.

We see that ALERT outperforms all three baselines. Since the zero-shot model is probabilistic, its decision is not significantly influenced by under-confident attribute predictions anyway, making BASESCORE less effective. The cross-validation baseline also fails to improve performance consistently (except for AWA) due to its global nature. However, ALERT makes image-specific decisions regarding which attributes to ignore and consistently improves performance across all four datasets. PubFig has the largest number of unseen test categories making knowledge transfer and ZSL harder. Here ALERT shows the most improvement. Note that ALERT here uses attribute predictions themselves as features. So the improvement in performance is not because of access to additional visual information – it is be-

[6]Notice that in this zero-shot learning application, ALERT is used on images from categories not seen by BASESYS during its training, making the "if you can train ALERT, you could have trained a better BASESYS in the first place" reasoning discussed in the introduction further inapplicable.

[7]ALERT ignores 12 out of 85 attributes per test image for AWA, 12 out of 73 for PubFig, 32 out of 102 for SUN and 7 out of 64 for UIUC.

cause of explicitly reasoning about failures.

6. Conclusion

We take a step in the direction of building self-evaluating vision systems that fail gracefully, making them more usable in real world applications even with their existing imperfections. We introduce ALERT, a straightforward warning system that analyzes the input instance and predicts if a vision system is likely to produce an unreliable response. We demonstrate its generality by applying it to four diverse applications ranging from segmentation and 3D layout estimation to image memorability prediction and attributes-based scene and object recognition. We show that an approach as simple as ALERT can predict failure surprisingly reliably, and can help improve the performance of a downstream application. We advocate the use of two metrics (accuracy vs. declaration rate curves and risk-averse metrics) to facilitate further progress in failure predicting systems.

Acknowledgement This research is supported in part by ARO YIP W911NF-14-1-0152 and ONR MURI N000141010934. We thank S. Satkin, D. Fouhey, and A. Suppe for providing data.

References

[1] O. Aghazadeh and S. Carlsson. Properties of datasets predict the performance of classiers. In *BMVC*, 2013. 3
[2] N. Bach, F. Huang, and Y. Al-Onaizan. Goodness: A method for measuring machine translation confidence. In *ACL*, 2011. 3
[3] M. Bilgic and L. Getoor. Reflect and correct: A misclassification prediction approach to active inference. *ACM KDD*, 2009. 3
[4] M. Boshra and B. Bhanu. Predicting performance of object recognition. *PAMI*, 2000. 3
[5] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, 2009. 3, 7
[6] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation, release 1. http://sminchisescu.ins.uni-bonn.de/code/cpmc/. 4
[7] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *CVPR*, 2010. 1, 4
[8] S. Choularton. Early stage detection of speech recognition errors, 2009. 3
[9] S. J. Delany, P. Cunningham, and D. Doyle. Generating estimates of classification confidence for a case-based spam filter. In *International Conference on Case-based Reasoning*, 2005. 3
[10] J. Deng, J. Krause, A. Berg, and L. Fei-Fei. Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition. In *CVPR*, 2012. 3
[11] U. R. Devarakota, B. Mirbach, and B. Ottersten. Confidence estimation in classification decision: A method for detecting unseen patterns. In *ICAPR*, 2007. 3
[12] M. Dredze and K. Crammer. Confidence-weighted linear classification. In *ICML*, 2008. 3
[13] R. P. W. Duin and D. M. J. Tax. Classifier Conditional Posterior Probabilities. In *Joint IAPR Int. Workshops on Advances in Pattern Recognition*, 1998. 3
[14] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. 1, 2
[15] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009. 1, 7
[16] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *Workshop on Generative-Model Based Vision, CVPR*, 2004. 1, 2
[17] E. Filisko and S. Seneff. Error detection and recovery in spoken dialogue systems. In *NAACL Workshop: Spoken Language Understanding for Conversational Systems*, 2004. 3
[18] J. Flavell and H. Wellman. Metamemory. *Perspectives on the Development of Memory and Cognition*, 1988. 3
[19] T. Gao and D. Koller. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In *ICCV*, 2011. 3
[20] P. Grother and E. Tabassi. Performance of biometric quality measures. *PAMI*, 2007. 2, 3, 4
[21] R. Haeusler and D. K. Rahul Nair. Ensemble learning for confidence measures in stereo vision. In *CVPR*, 2013. 3
[22] P. Hammer, P. Hansen, and B. Simeone. Roof duality, complementation and persistency in quadratic 0-1 optimization. *Math. Programming*, 1984. 3
[23] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *ICCV*, 2009. 1, 5
[24] D. Hoiem, Y. Chodpathumwan, and Q. Dai. Diagnosing error in object detectors. In *ECCV*, 2012. 3
[25] P. Isola, J. Xiao, A. Torralba, and A. Oliva. What makes an image memorable? In *CVPR*, 2011. 1, 5
[26] N. Jammalamadaka, A. Zisserman, M. Eichner, V. Ferrari, and C. V. Jawahar. Has my algorithm succeeded? an evaluator for human pose estimators. In *ECCV*, 2012. 3
[27] H. Jiang. Confidence measures for speech recognition: A survey. *Speech Communication*, 2005. 3
[28] A. Khosla, T. Zhou, T. Malisiewicz, A. Efros, and A. Torralba. Undoing the damage of dataset bias. In *ECCV*, 2012. 2
[29] M. Kukar. Estimating confidence values of individual predictions by their typicalness and reliability. In *ECAI*, 2004. 3
[30] N. Kumar, A. Berg, P. Belhumeur, and S. Nayar. Attribute and simile classifiers for face verification. In *ICCV*, 2009. 6
[31] C. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009. 1, 4, 6, 7
[32] N. J. Leap. *A confidence paradigm for classification systems*. Air Force Institute of technology, Biblioscholar publisher, 2012. 2
[33] D. C. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. In *CVPR*, 2009. 2, 5
[34] Y. J. Lee and K. Grauman. Object-graphs for context-aware category discovery. In *CVPR*, 2010. 3
[35] L. Li, M. L. Littman, and T. J. Walsh. Knows what it knows: a framework for self-aware learning. In *ICML*, 2008. 3
[36] Z. Liao, A. Farhadi, Y. Wang, I. Endres, and D. A. Forsyth. Building a dictionary of image fragments. In *CVPR*, 2012. 3
[37] O. Mac Aodha, G. J. Brostow, and M. Pollefeys. Segmenting video into classes of algorithm-suitability. In *CVPR*, 2010. 3
[38] G. A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 1995. 3
[39] M. Muhlbaier, A. Topalis, and R. Polikar. Ensemble confidence estimates posterior probability. In *Int. Conf. on Multiple Classifier Systems*, 2005. 3
[40] D. Munoz, J. A. D. Bagnell, and M. Hebert. Stacked hierarchical labeling. In *ECCV*, 2010. 4
[41] L. Navarro-Serment, A. Suppe, D. Munoz, D. Bagnell, and M. Hebert. An architecture for online semantic labeling on ugvs. In *Proc. SPIE Unmanned Systems Technology XV*, 2013. 1
[42] G. Patterson and J. Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*, 2012. 7
[43] P. J. Phillips and J. R. Beveridge. An introduction to biometric-completeness: the equivalence of matching and quality. In *IEEE international conference on Biometrics: Theory, applications and systems*, 2009. 2
[44] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary mrfs via extended roof duality. Technical report, CVPR, 2007. 3
[45] A. Sarma and D. D. Palmer. Context-based speech recognition error detection and correction. In *NAACL (Short papers)*, 2004. 3
[46] S. Satkin, J. Lin, and M. Hebert. Data-driven scene understanding from 3D models. In *BMVC*, 2012. 1, 5
[47] W. Scheirer, N. Kumar, P. Belhumeur, and T. Boult. Multi-attribute spaces: Calibration for attribute fusion and similarity search. In *CVPR*, 2012. 3
[48] W. Scheirer, A. Rocha, R. Michaels, and T. Boult. Robust fusion: Extreme value theory for recognition score normalization. In *ECCV*, 2012. 3
[49] W. J. Scheirer, A. Rocha, and T. E. Boult. Learning for meta-recognition. *IEEE T.IFS*, 2012. 3
[50] W. J. Scheirer, A. Rocha, R. J. Micheals, and T. E. Boult. Meta-recognition: The theory and practice of recognition score analysis. *PAMI*, 2011. 3
[51] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling appearance, shape and context. *IJCV*, 2007. 1
[52] D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. In *Optimization for Machine Learning*. MIT Press, 2011. 3
[53] E. Tabassi, C. Wilson, and C. Watson. Fingerprint image quality. *NIST Internal Report 7151*, 2004. 2, 3, 4
[54] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *CVPR'*, 2011. 2
[55] J. W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977. 5
[56] P. Wang, Q. Ji, and J. L. Wayman. Modeling and predicting face recognition system performance based on analysis of similarity scores. *PAMI*, 2007. 3
[57] R. Wang and B. Bhanu. Learning models for predicting recognition performance. In *ICCV*, 2005. 3
[58] P. Welinder, M. Welling, and P. Perona. A lazy man's approach to benchmarking: Semisupervised classifier evaluation and recalibration. In *CVPR*, 2013. 3
[59] J. J. Westerkamp, D. C. Gross, and A. Palomino. Automatic target recognition of time critical moving targets using 1d high range resolution (hrr) radar. *IEEE Aerospace and Electronic Systems Magazine*, 2000. 2
[60] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 1, 4, 5
[61] W. Zhang, S. X. Yu, and S.-H. Teng. Power svm: Generalization with exemplar classification uncertainty. In *CVPR*, 2012. 3