Network Generalization Prediction for Safety Critical Tasks in Novel Operating Domains

Molly O'Brien[1] molly@jhu.edu
Mike Medoff [2] mmedoff@exida.com
Julia Bukowski[3] julia.bukowski@villanova.edu
Greg Hager[1] hager@cs.jhu.edu

[1] Department of Computer Science, Johns Hopkins University, Baltimore MD 21218
[2]exida LLC, Sellersville PA 18960
[3]Department of Electrical and Computer Engineering, Villanova University, Villanova, PA 19085

Abstract

It is well known that Neural Network (network) performance often degrades when a network is used in novel operating domains that differ from its training and testing domains. This is a major limitation, as networks are being integrated into safety critical, cyber-physical systems that must work in unconstrained environments, e.g., perception for autonomous vehicles. Training networks that generalize to novel operating domains and that extract robust features is an active area of research, but previous work fails to predict what the network performance will be in novel operating domains. We propose the task ***Network Generalization Prediction****: predicting the expected network performance in novel operating domains. We describe the network performance in terms of an interpretable Context Subspace, and we propose a methodology for selecting the features of the Context Subspace that provide the most information about the network performance. We identify the Context Subspace for a pretrained Faster RCNN network performing pedestrian detection on the Berkeley Deep Drive (BDD) Dataset, and demonstrate Network Generalization Prediction accuracy within* 5% *of observed performance. We also demonstrate that the Context Subspace from the BDD Dataset is informative for completely unseen datasets, JAAD and Cityscapes, where predictions have a bias of* 10% *or less.*

1. Introduction

Deep Neural Networks (networks) are being integrated into commercial, safety critical, autonomous systems that operate in unconstrained environments, e.g., perception for autonomous vehicles. When a network is deployed in an unconstrained environment, the operating domain, i.e., the distribution of context features that describe the network's environment, can change significantly from the testing domain, i.e., the distribution of context features that describe the test data. Safety critical systems are regulated by international functional safety standards, e.g., ISO 26262 for the automotive industry, IEC 61508 for electronics and software. Functional safety standards leverage various techniques to verify the safety of software, including requirement specification, i.e., linking required system behavior to specific code modules, white box testing, i.e., testing specific inputs that cover all branches or behavior in the code, and code inspection and review to identify human error. These techniques are challenging or impossible to apply directly to networks, e.g., labeled data is used to implicitly specify the correct behavior in supervised learning, networks are black box systems, and network weights cannot be manually inspected to identify failure cases.

New techniques are needed to bridge the gap between the high performance of deep networks and the verification required for safety critical systems. In particular, the ability to predict how a network's performance will change in a novel operating domain can enable verifying the required level of performance before a network is deployed, we denote this task Network Generalization Prediction. We propose a methodology for Network Generalization Prediction for networks trained via supervised learning. Our contributions are as follows:

1. We introduce the concept of a Context Subspace, a low-dimensional space, encoding the context features most informative about the network performance.
2. We propose a greedy feature selection algorithm for identifying the Context Subspace by 1) ranking the context features by the information they provide about the network loss, and 2) selecting the subspace dimensionality that leads to accurate Network Generalization Prediction.
3. We leverage a Context Subspace for accurate Network Generalization Prediction for pedestrian detec-

tion in diverse operating domains, with a prediction error from 0.5% to 2% for not safety critical pedestrians (pedestrians not in the road), and a prediction error from 2% to 5% for safety critical pedestrians (pedestrians in the road).

4. We demonstrate that the Context Subspace identified for the Berkeley Deep Drive Dataset (BDD) can be used to predict pedestrian recall in completely unseen datasets, the JAAD and Cityscapes Datasets, with a prediction bias of 10% or less.

2. Background

2.1. Network Dependability

Avizienis et al. defined software dependability as "the ability to deliver service that can justifiably be trusted," where dependability encompasses availability, reliability, safety, integrity, and maintainability[1]. To describe the dependability of a learned model, O'Brien et al. defined ML Dependability as "the probability that a model will succeed when operated under specified conditions"[14]. Cheng et al. proposed that Robustness, Interpretability, Completeness, and Correctness contribute to a network's Dependability [4]. Ponn et al. trained a random forest to predict whether a network would detect a pedestrian, based on pedestrian attributes; they denote this task Detection Performance Modeling[15]. Where as Detection Performance Modeling predicts whether one specific object will be detected, Network Generalization Prediction predicts the expected network performance for a given operating domain, described by a distribution of context features.

2.2. Network Generalization

It has been shown that underspecification causes network performance to degrade when deployed in operating conditions different from the training and testing conditions[6]. The WILDS benchmark was released to provide datasets with "in-the-wild" distribution shifts between the training and test data [11]. Subbaswamy et al. propose to evaluate a model's robustness to distribution shifts with one fixed evaluation set [18]. Common techniques to improve network generalization include extracting features robust to changing conditions[19], [10], zero or few-shot learning [24], [23], and identifying when an input is outside the network's training distribution [13], [7].

2.3. Feature Selection

Feature selection algorithms aim to select a subset of the available features, typically to use the features as input to train a model for a given task. Feature selection algorithms can be classified as filter methods, i.e., features are scored according to their association with the task label, wrapper methods, i.e., features are selected to minimize task error, and embedded methods, i.e., features are selected in the model training process [3]. The Mutual Information [12] is often used in filter methods to measure the information between a given feature and the desired label [22]. As exhaustive feature selection search is typically intractable, greedy feature selection algorithms are often used [9], [21], [8]. Note, greedy feature selection is related to matching pursuit in the sparse approximation literature [20] and has applications in compressed sensing [2].

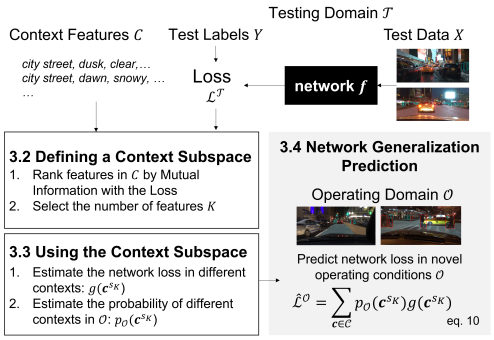

Figure 1. Overview of Network Generalization Prediction.

3. Methods

3.1. Problem Formulation

It is well known that in supervised learning, a network, f, is trained to produce a label, y_i, from data, x_i, and a loss function, $l(f(x_i), y_i)$, is used to drive training. In Network Generalization Prediction, we are **not** training f. Instead, we aim to predict the performance of a fixed network f, trained via supervised learning, when deployed in an operating domain, $\mathcal{O}$, that differs from the testing domain, $\mathcal{T}$, see Figure 1. The performance of f is measured using test data, $X = \{x_i\}_{i=1}^N$, and test labels, $Y = \{y_i\}_{i=1}^N$, via a loss function $L = \{l(f(x_i), y_i)\}_{i=1}^N$, where the elements of L are assumed to be discrete and bounded, e.g., an object detection flag, whether a safety criteria was satisfied, or a discretized classification error.

$\mathcal{T}$ is described via J context features, $\mathbf{C} = \{\mathbf{c}_i\}_{i=1}^N$, where $\mathbf{c}_i$ indicates a J dimensional context vector associated with x_i. Context features, e.g., image brightness, weather, or robot speed, can be categorical or numerical; numerical features are assumed to be discrete or discretized. It is possible for multiple test samples to map to the same context, i.e., $\mathbf{c}_i = \mathbf{c}_j, i \neq j$. $p_{\mathcal{T}}(\mathbf{c})$ denotes the probability of encountering $\mathbf{c}$ in $\mathcal{T}$. $\mathcal{O}$ is described by the probability of encountering $\mathbf{c}$ in $\mathcal{O}$, $p_{\mathcal{O}}(\mathbf{c})$. In many practical applications, the likelihood of encountering a context may be known without annotated data, e.g., there is a 25% chance

of snow in Boston, etc. Note, labeled test data from $\mathcal{O}$ is not required. We assume that while the distribution of contexts shifts between the testing and operating domains, i.e., $p_{\mathcal{T}}(\mathbf{c}) \neq p_{\mathcal{O}}(\mathbf{c})$, the expected network performance in context $\mathbf{c}$ is stable for both the testing and operating domains. Table 1 describes the Notation used in the Methods Section.

As is typical, we approximate the posterior expected loss in $\mathcal{T}$, $\mathcal{L}^{\mathcal{T}}$, using the empirical loss:

$$\mathcal{L}^{\mathcal{T}} = E[l(f(X), Y)] = \frac{1}{N}\sum_{i=1}^{N} l(f(x_i), y_i) \tag{1}$$

We define $g(\mathbf{c}) = E[l(f(X), Y|\mathbf{c})]$. Let $\mathbb{I}(\mathbf{a}, \mathbf{b})$ be an indicator function that is equal to 1 if $\mathbf{a} = \mathbf{b}$ and 0 otherwise. $g(\mathbf{c})$ can be computed as:

$$g(\mathbf{c}) = \frac{\sum_{i=1}^{N} \mathbb{I}(\mathbf{c}_i, \mathbf{c}) * l(f(x_i), y_i)}{\sum_{i=1}^{N} \mathbb{I}(\mathbf{c}_i, \mathbf{c})} \tag{2}$$

$\mathcal{L}^{\mathcal{T}}$ can equivalently be computed as:

$$\mathcal{L}^{\mathcal{T}} = \sum_{\mathbf{c}\in\mathbf{C}} p_{\mathcal{T}}(\mathbf{c}) g(\mathbf{c}) \tag{3}$$

Likewise, we can now express the Network Generalization Prediction, $\hat{\mathcal{L}}^{\mathcal{O}}$, as:

$$\hat{\mathcal{L}}^{\mathcal{O}} = \sum_{\mathbf{c}\in\mathbf{C}} p_{\mathcal{O}}(\mathbf{c}) g(\mathbf{c}) \tag{4}$$

This formulation holds theoretically for any number of context features J. However, as J grows linearly, computing Eqn. 4 requires exponentially more test samples to cover every possible $\mathbf{c} \in \mathbf{C}$. Thus, we introduce the Context Subspace, $\mathbf{C}^{S_K}$, a low-dimensional space, encoding the context features most informative about the network performance.

3.2. Defining a Context Subspace

We are interested in selecting the K context features that provide the most information about the network loss, to include these features in $\mathbf{C}^{S_K}$. Let $S_K = \{s_k\}_{k=1}^{K}$ be the indices of context features of interest and $\mathbf{C}^{S_K} = \{C^{s_k}\}_{k=1}^{K}$, where $C^{s_k} = \{c_i^{s_k}\}_{i=1}^{N}$ are the annotated attributes for each example in the test set for context feature s_k. To select the context features to include in $\mathbf{C}^{S_K}$, we 1) rank the context features by how much information they provide about the network loss, 2) select the $\mathbf{C}^{S_K}$ dimensionality K to enable accurate Network Generalization Prediction.

3.2.1 Ranking Context Features

Recall, the Mutual Information is often used to rank features in filter feature selection algorithms and is computed as $I(L, C^j)$ for loss L and context feature C^j:

	Notation
$X = \{x_i\}_{i=1}^{N}$	The Test Data
$Y = \{y_i\}_{i=1}^{N}$	The Test Labels
f	The trained network
$L = \{l(f(x_i), y_i)\}_{i=1}^{N}$	The Test Set Loss
$\mathbf{C}$	The context features
$\mathbf{c} \in \mathbf{C}$	A context vector
$g(\mathbf{c})$	The expected loss of f in $\mathbf{c}$
$\mathbf{C}^{S_K}$	The Context Subspace
$\mathcal{T}$	The Testing Domain
$\mathcal{O}$	The Operating Domain
$p_{\mathcal{T}}(\mathbf{c})$, $p_{\mathcal{O}}(\mathbf{c})$	The probability of $\mathbf{c}$ in $\mathcal{T}$, $\mathcal{O}$
$\mathcal{L}^{\mathcal{T}}$	The observed loss in $\mathcal{T}$
$\hat{\mathcal{L}}^{\mathcal{O}}$	The predicted loss in $\mathcal{O}$

Table 1. Notation.

$$I(L, C^j) = \sum_{\ell\in L}\sum_{c\in C^j} p(\ell, c) log(\frac{p(\ell, c)}{p(\ell)p(c)}) \tag{5}$$

where $p(\ell, c)$ indicates the joint probability of ℓ and c, and $p(\ell)$ and $p(c)$ indicate the marginal probabilities for ℓ and c, respectively. The Interaction Information is a generalization of the Mutual Information to K features. The Interaction Information between L and the context features $C^{s_1}, ..., C^{s_K}$ is defined as:

$$\begin{aligned} I(L, C^{s_1}, ..., C^{s_K}) = I(L, C^{s_1}, ..., C^{s_{K-1}}) \\ - I(L, C^{s_1}, ..., C^{s_{K-1}}|C^{s_K}) \end{aligned} \tag{6}$$

For two features, this becomes:

$$I(L, C^{s_1}, C^{s_2}) = I(L, C^{s_2}) - I(L, C^{s_2}|C^{s_1}) \tag{7}$$

Where $I(L, C^{s_2}|C^{s_1})$ can be computed as:

$$\begin{aligned} I(L, C^{s_2}|C^{s_1}) = \sum_{\ell\in L}\sum_{c_2\in C_2}\sum_{c_1\in C_1} p(\ell, c_2, c_1) \\ \times log\left(\frac{p(\ell, c_2, c_1)}{p(\ell, c_1)p(c_2, c_1)}\right) \end{aligned} \tag{8}$$

The computational complexity of $I(L, C^{s_1}, ..., C^{s_K})$ grows combinatorially with K. We are interested in ranking the context features by the Interaction Information, but computing the exact Interaction Information becomes intractable as K grows. To make computation tractable, we propose $\Delta I(L, C^{s_1}, ..., C^{s_K})$ to approximate how much more information including context feature C^{s_K} in the Context Subspace provides about L.

$$\Delta I(L, C^{s_1}, ..., C^{s_K}) = I(L, C^{s_K}) - \sum_{k=1}^{K-1} I(C^{s_k}, C^{s_K}) \tag{9}$$

Algorithm 1 Greedy ΔI Context Selection

$S_K = \{\}$
for $k = 1 : K$ **do**
  $s_k^* \leftarrow argmax_j[I(C^j, L) - \sum_{s_k \in S_K} I(C^j, C^{s_k})]$
  $\forall j \in J \setminus S_K$
  $S_K = S_K \cup s_k^*$
end for
$g(\mathbf{c}^{S_K}) = E[l(f(X), Y|\mathbf{c}^{S_K})]$

Intuitively, $\Delta I(L, C^{s_1}, ..., C^{s_K})$ subtracts the redundant information in C^{s_K}, $\sum_{k=1}^{K-1} I(C^{s_k}, C^{s_K})$, from the information it provides about the loss, $I(L, C^{s_K})$. Note that the computational complexity of computing $\Delta I(L, C^{s_1}, ..., C^{s_K})$ grows linearly with K. Like the Interaction Information, $\Delta I(L, C^{s_1}, ..., C^{s_K})$ can be positive or negative. In Appendix A, we show that for independent features in the Context Subspace, $\Delta I(L, C^{s_1}, C^{s_2})$ approaches $I(L, C^{s_1}, C^{s_2})$ as C^{s_2} approaches perfect information on L. We propose a greedy algorithm to iteratively select the K most informative features from the context, see Algorithm 1.

3.2.2 Selecting the Context Subspace Dimensionality

Selecting the number of features, K, to include in $\mathbf{C}^{S_K}$ is not trivial. Recall, the context features are discretized. If each feature has n unique values, the total number of unique contexts is n^K: as you increase the number of context features K, the number of unique contexts to describe with the test data increases exponentially. Including more features can lead to a more descriptive $\mathbf{C}^{S_K}$ but can also lead to many untested contexts in $\mathbf{C}^{S_K}$. To select K, we compute the expected prediction error for a given subspace dimensionality, ϵ_K. Using the K most informative context features, $g(\mathbf{c}^{S_K}) = E[l(f(X), Y|\mathbf{c}^{S_K})]$ can be computed according to Eqn. 2. where $\mathbf{c}^{S_K}$ is a K dimensional feature vector in $\mathbf{C}^{S_K}$. We iteratively compute the prediction error within the test set, ϵ_K, to estimate the expected prediction error $\tilde{\epsilon}_K$, see Algorithm 2. First, we randomly partition the test set into a fit set and a val set: X^{fit}, Y^{fit}, $\mathbf{C}^{fit}$ with N^{fit} samples and X^{val}, Y^{val}, $\mathbf{C}^{val}$ with N^{val} samples respectively. We estimate $g^{fit}(\mathbf{c}^{S_K})$ using the fit set. We compute the observed loss from the val set, $\mathcal{L}^{val}$. Let $p^{val}(\mathbf{c}^{S_K})$ indicate the probability of encountering context $\mathbf{c}^{S_K}$ in $\mathbf{C}^{val}$. The prediction error, ϵ_K, is the difference between the observed validation loss, $\mathcal{L}^{val}$, and the predicted validation loss using $g^{fit}(\mathbf{c}^{S_K})$. This procedure can be iterated multiple times, and the subsequent ϵ_K's averaged, to estimate the expected prediction error, $\tilde{\epsilon}_K$, for different random fit and val partitions of the test set. We select the K that minimizes $\tilde{\epsilon}_K$.

$\tilde{\epsilon}_K$ measures the expected prediction error within $\mathcal{T}$.

Algorithm 2 Context Subspace Dimensionality Selection

$\tilde{\epsilon}_K = \{\}$
for $K = 1 : J$ **do**
  $\epsilon_K s = []$
  for $iteration$ **do**
    split test set into fit and val set
    $g^{fit}(\mathbf{c}^{S_K}) = E[l(f(X^{fit}), Y^{fit}|\mathbf{c}^{S_K})]$
    $\mathcal{L}^{val} = \frac{1}{N^{val}} \sum_{i=1}^{N^{val}} l(f(x_i^{val}), y_i^{val})$
    $\epsilon_K = |\mathcal{L}^{val} - \sum_{\mathbf{c}^{S_K} \in \mathbf{C}^{S_K}} p^{val}(\mathbf{c}^{S_K}) g_k^{fit}(\mathbf{c}^{S_K})|$
    $\epsilon_K s.append(\epsilon_K)$
  end for
  $\tilde{\epsilon}_K = mean(\epsilon_K s)$
end for
$K \leftarrow argmin_K \quad \tilde{\epsilon}_K$

When the context is informative about the loss, we expect $\tilde{\epsilon}_K$ to decrease as K increases until an optimal K^* is reached, then $\tilde{\epsilon}_K$ will begin to rise as K increases and there are many untested contexts. If $\tilde{\epsilon}_K$ is flat or increasing as K increases, it indicates that the context features available are not informative about the loss.

After we have ranked the context features and selected the number of features to include in the subspace, we can form $\mathbf{C}^{S_K}$. The K most informative context features form the axes of the subspace. Recall, we assumed the context features are categorical or numerical and discrete, this yields a finite set of context partitions, $\mathbf{c}^{S_K} \in \mathbf{C}^{S_K}$.

3.3. Using the Context Subspace

We use $\mathbf{C}^{S_K}$ to describe the expected network loss in different contexts, $g(\mathbf{c}^{S_K})$, and to describe the probability of encountering a context in the operating domain, $p_{\mathcal{O}}(\mathbf{c}^{S_K})$. We can compute $g(\mathbf{c}^{S_K})$ using Eqn. 2, note we use the entire test set to compute $g(\mathbf{c}^{S_K})$ once we have selected the subspace dimensionality K. Recall, we do not assume to have labeled test data in $\mathcal{O}$, but we do assume to know $p_{\mathcal{O}}(\mathbf{c}^{S_K})$. Individual context feature probabilities can be multiplied to obtain a joint probability distribution if the context feature probabilities are assumed to be independent.

3.4. Network Generalization Prediction

We can now perform Network Generalization Prediction, where $\hat{\mathcal{L}}^{\mathcal{O}}$ is the predicted loss in $\mathcal{O}$:

$$\hat{\mathcal{L}}^{\mathcal{O}} = \sum_{\mathbf{c}^{S_K} \in \mathbf{C}^{S_K}} p_{\mathcal{O}}(\mathbf{c}^{S_K}) g(\mathbf{c}^{S_K}) \tag{10}$$

Recall, we selected a small number of informative context features so that it would be practical to describe the unique contexts $\mathbf{c}^{S_K} \in \mathbf{C}^{S_K}$, but there may be untested contexts in $\mathbf{C}^{S_K}$. For conservative predictions, we assume the maximum loss in untested contexts. The maximum loss may

correspond to a binary failure or a large expected error. Leveraging $\mathbf{C}^{S_K}$ renders Network Generalization Prediction practical for interestingly complex applications, like perception for autonomous vehicles.

4. Experimental Results

4.1. Pedestrian Detection Generalization

Perception for autonomous vehicles is an active area of research, and systems that use deep networks to detect and avoid obstacles, like pedestrians, while driving are commercially available. Some of these commercial systems can be used in any driving conditions, at the user's discretion, and the operating domains can vary significantly in terms of the lighting conditions, e.g., daytime compared to night, road conditions, e.g., dry roads in clear weather compared to slippery roads in rainy or snowy weather, and obstacle density, e.g., a residential street compared to a restricted access highway. It would be impractical for autonomous vehicle developers to test a perception system in every possible operating domain, but it is also imperative to know whether it is safe to use a perception system in a given operating domain. We perform experiments analogous to an autonomous vehicle developer: we test a fixed network in one testing domain, $\mathcal{T}$, and predict the network's performance in novel operating domains, where the distribution of context features vary significantly from $\mathcal{T}$. Our goal is to accurately predict the observed network performance when the network is used in a novel operating domain, $\mathcal{O}$.

We test a pretrained Faster RCNN [17] object detector for pedestrian detection, where the objects detected as $person$ are used as pedestrian detections. In our analysis, we consider pedestrians whose ground truth bounding box area is ≥ 300 pixels. We evaluate the network performance at the pedestrian level. Pedestrians correctly detected with an $IoU > 0.5$ and a confidence score > 0.5 are assigned a loss of 0; pedestrians that are not detected are assigned a loss of 1[1]. Pedestrians in images with multiple people are considered independently; images with no pedestrians present are not assigned any loss.

BDD Dataset [25] was recorded across the continental US and includes data from varying times of day (daytime, dawn/dusk, or night), weather conditions (clear, partly cloudy, overcast, rainy, foggy, or snowy), and scene types (city street, residential, or highway). BDD images are of size 720×1280. We use $10,000$ images from the BDD Dataset for testing, denoted the BDD Test Set. We use the remaining $70,000$ images in the BDD Dataset, denoted the BDD Operating Set, to define novel operating domains. The BDD Test Set and BDD Operating Set correspond to the BDD "Validation" and "Train" folds, respectively.

[1]We are predicting the network's recall. We do not assign a loss for false positive detections; this same methodology can be used to predict network precision if that is of interest. We focus on recall because failing to predict a pedestrian who is truly present in the scene is a higher safety risk than trying to avoid a pedestrian who is not present.

4.2. Defining the Context Subspace

We evaluate the network performance at the pedestrian level; therefore, context features are assigned to individual pedestrians. We do not know a priori which pedestrian attributes are informative about the network loss, so we include all available context features. The BDD dataset includes metadata on the image time of day, weather, and scene type. We include the metadata as context features. We also include the image brightness and the pedestrian bounding box brightness. We define the road(s) to be the safety critical (SC) region(s) in the images. Pedestrians in the road are labeled SC, pedestrians outside the road, e.g., on the sidewalk, are labeled not safety critical (NSC). The road is defined using the drivable area annotations. Whether a pedestrian is SC, denoted the safety critical flag, is included as a context feature. To capture information about the obstacle density in the scene, we include the total number of pedestrians, the number of SC pedestrians, and the number of NSC pedestrians in the image as context features.

4.2.1 Ranking Context Features

We use Algorithm 1 to rank the context features by how much information they provide about the network loss. When computing the mutual information for a numerical feature with more than 10 unique values, we uniformly partition the feature into 10 discrete bins. Categorical features are labeled discretely with their assigned labels. See Figure 2 left for the ΔI computed for the first three iterations of Algorithm 1. The six most informative features were found to be: 1) image brightness, 2) safety critical flag, 3) scene , 4) number SC pedestrians, 5) time of day, and 6) bounding box brightness.

4.2.2 Selecting the Context Subspace Dimensionality

To select the number of features to include in the Context Subspace, we compute $\tilde{\epsilon}_K$ for values of K from 1 to 6. For each dimensionality, K, we compute ϵ_K 50 times by randomly partitioning the test data into 50% for fitting $g(\mathbf{c}^{S_K})$ and 50% for validation. We select the K with the minimum expected prediction error $\tilde{\epsilon}_K$ over the 50 iterations. $K = 3$ was found to be optimal, with an average prediction error of 0.63%, see Figure 2 right. We subsequently define the Context Subspace with three dimensions: 1) image brightness, 2) safety critical flag, and 3) scene.

The image brightness is a continuous feature; we uniformly partition the image brightness into 10 bins. The safety critical flag and the scene type are discrete and categorical features with 2 and 3 possible values, respectively.

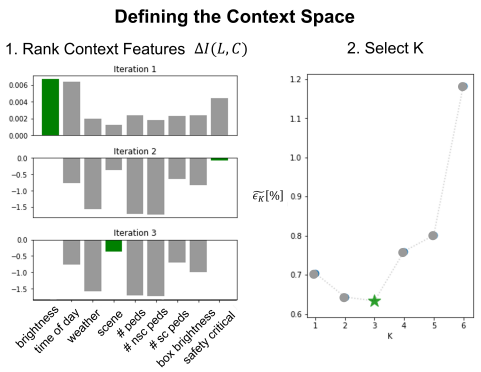

Figure 2. Defining the Context Subspace. 1) Rank Context Features: The $\Delta I(L, C)$ between different context features and the loss in the BDD Test Set for the first three rounds of Algorithm 1. Note that in iteration one, $\Delta I(L, C) = I(L, C)$ so the features' scores are non-negative. 2) Select K: We estimate the expected prediction error for different Context Subspace dimensionalities, K, and choose the dimensionality with the lowest expected prediction error: in this case, $K = 3$. We form the Context Subspace with the three most informative context features: brightness, safety critical flag, and the scene type.

This results in a Context Subspace, $\mathbf{C}^{S_K}$, with 60 discrete contexts, $\mathbf{c}^{S_K}$.

4.3. Using the Context Subspace

We use $\mathbf{C}^{S_K}$ to estimate the expected network loss in a context, $g(\mathbf{c}^{S_K})$, and to describe the probability of encountering a context in $\mathcal{O}$, $p_{\mathcal{O}}(\mathbf{c}^{S_K})$. For all tested contexts, $g(\mathbf{c}^{S_K})$ is computed according to Eqn. 2. All untested contexts are assigned an expected loss of 1, i.e., a 100% chance of failing to detect a pedestrian. The BDD Operating Set is used to define four novel operating domains: 1) day, small groups; 2) day, large groups; 3) night, small groups; and 4) night, large groups, see Figure 3. The time of day annotated in the images was used to assign "day" or "night". The SC and NSC pedestrians are considered independently. Pedestrians in images with fewer than 5 (N)SC pedestrians are categorized as small groups; pedestrians in images with 5 or more (N)SC pedestrians are categorized as large groups, i.e., in an image with 2 SC pedestrians and 15 NSC pedestrians, the SC pedestrians would be labeled 'small group' and the NSC pedestrians would be labeled 'large group'. We compute $p_{\mathcal{O}}(\mathbf{c}^{S_K})$ for each $\mathcal{O}$ by counting the number of pedestrians that fall into each $\mathbf{c}^{S_K} \in \mathbf{C}^{S_K}$ and dividing by the total number of pedestrians.

4.4. Pedestrian Detection Generalization Prediction

We predict the network loss in the novel operating domains defined in 4.3 using Eqn. 10. Our network loss is equivalent to the fraction of pedestrians that are not detected by the network; we convert the predictions into the predicted network recall by subtracting the fraction of pedestrians that are not detected from 1, see Figure 4. For reference, we include the average percent of pedestrians detected in the Test Set as a naïve baseline. We then pass the BDD Operating Set through the network; the observed recall is computed as the fraction of pedestrians that were correctly detected. Figure 4 illustrates that our predictions are accurate with Network Generalization Prediction accuracy between 0.5% and 2.5% for NSC pedestrian recall and 2% and 5% for SC pedestrian recall. All the SC predictions underpredict the observed recall; this demonstrates that our predictions are conservative. Note, the only prediction with significant error is for night, large group SC pedestrians. Only one image in the BDD Operating Set falls into this category, so the observed performance is based on minimal data.

4.5. Generalization Prediction for Unseen Datasets

As a preliminary study, we investigate whether the Context Subspace, $\mathbf{C}^{S_K}$, defined using the BDD Test Set and the network loss, $g(\mathbf{c}^{S_K})$, estimated from the BDD Test Set provide information about completely unseen datasets. Unseen datasets include shifts in the context feature distributions, as well as changes in camera parameters and physical setup that are not captured by the test set. As such, we expect predictions for unseen dataset to contain bias, i.e., the prediction error for an unseen dataset will have a consistent non-zero offset. We are interested in determining the magnitude of this prediction bias to evaluate the usefulness of Network Generalization Prediction across datasets. We perform Network Generalization Prediction for the JAAD Dataset [16], and the Cityscapes Dataset with the gtFine labels [5], see Figure 5 for sample images. For both datasets, the (N)SC pedestrian image brightness distribution is computed from the images.

The JAAD Dataset was recorded in North America and Europe; it includes primarily daytime images from residential and city streets in varying weather conditions. JAAD images are of size 1080×1920. For the JAAD Dataset, we sampled images every three seconds from the videos to limit temporal correspondence between frames; this resulted in 1,031 images. Pedestrians in the road were manually annotated as SC, all others were labeled NSC. Scene annotations are not available for the JAAD dataset. To estimate the probability distribution of scenes, the scene type was annotated for a subset of 100 images, we assume the distribution holds for the entire dataset. The marginal (N)SC image brightness distributions and scene type distribution are multiplied to obtain the joint probability distributions for the JAAD Dataset.

The Cityscapes Dataset contains $3,475$ images recorded in 50 cities across Germany in the daytime during fair

Figure 3. BDD Novel Operating Domains. We define operating domains based on the time of day and the number of pedestrians in an image. Images with fewer than 5 (N)SC pedestrians fall under small groups. Images with 5 or more (N)SC pedestrians fall under large groups. Sample images from the operating domains are shown. NSC pedestrians are outlined in blue. SC pedestrians are outlined in red. Drivable area is shown in random transparent colors.

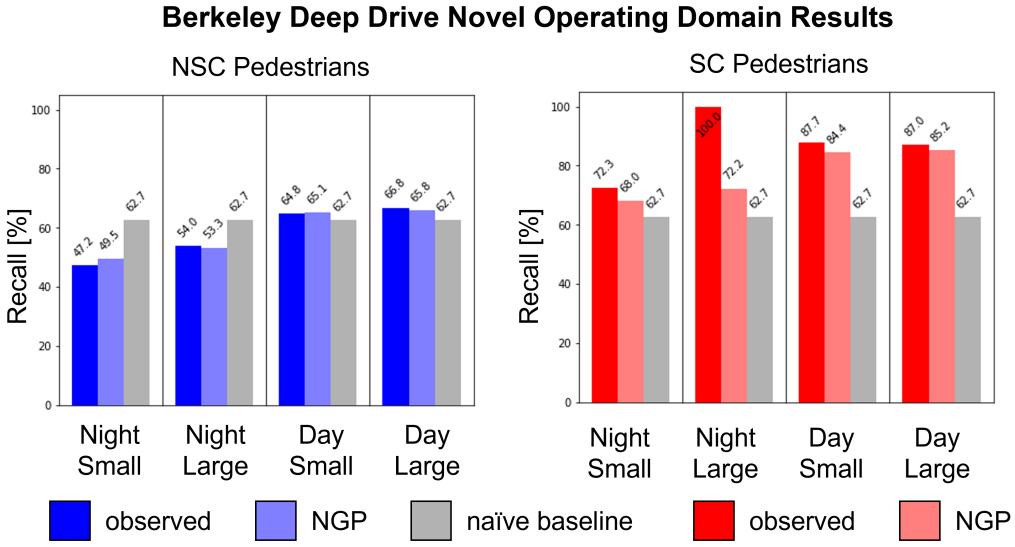

Figure 4. Network Generalization Prediction (NGP) Results. NSC pedestrian recall and SC pedestrian recall are shown separately. The observed pedestrian recall for the images of each operating domain are shown in bright blue or red. The NGP predicted recall is shown in light blue or red. The naïve baseline indicates the average recall over all pedestrians in the Test Set. Note that the naïve baseline is the same for every operating domain.

weather conditions. Cityscapes images are of size 1024×2048. We defined the pedestrian bounding boxes using the outermost edges of the labeled person instance segmentations, and we used the semantic segmentation of the road to define the SC region in the image. For Cityscapes, the scene type is known to be "city street".

We make Network Generalization Predictions for the JAAD and Cityscapes Datasets using $g(\mathbf{c}^{S_K})$, estimated using the BDD Test Set, see Figure 6. The prediction bias is consistently around 10%, with a minimum prediction error of 5% for SC pedestrian recall in the JAAD Dataset. We underpredict pedestrian recall for the JAAD Dataset and we overpredict pedestrian recall for the Cityscapes Dataset.

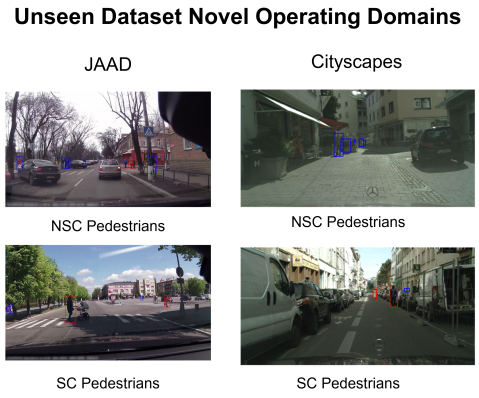

Figure 5. Unseen Dataset Novel Operating Domains. Sample images from the unseen datasets. NSC pedestrians outlined in blue. SC pedestrians outlined in red.

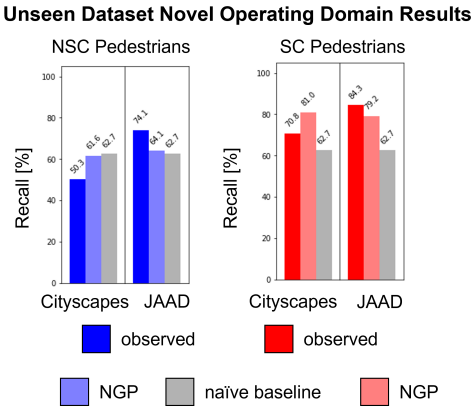

Figure 6. Network Generalization Prediction for Unseen Datasets. NSC pedestrian recall and SC pedestrian recall are shown separately. The observed pedestrian recall for the images of each novel dataset are shown in bright blue or red. The NGP predicted recall is shown in light blue or red. The naïve baseline indicates the average recall over all pedestrians in the Test Set. Note that the naïve baseline is the same for both unseen datasets.

5. Discussion

We make accurate Network Generalization Predictions for the BDD Operating Set, where the observed recall varies from 47% to 87%. This demonstrates that a fixed test set can be used to predict a network's performance in diverse, novel operating domains. The observed recall for SC pedestrians is about 20% higher than for NSC pedestrians. This makes intuitive sense, as SC pedestrians tend to be central in the image and closer to the vehicle. This is encouraging, because the performance of perception systems for autonomous vehicles will ultimately be determined by how well they detect SC pedestrians and obstacles. However, in the BDD Test Set there are many more examples of NSC pedestrians, $11,169$, than SC pedestrians, 484. This leads to more untested contexts for the SC pedestrians, which in turn leads to the slight underprediction of SC recall.

For unseen datasets, we find a Network Generalization Prediction bias of 10%; we believe these results are promising and that the results indicate the Context Subspace identified for one dataset, e.g., one camera setup and one physical setup, can be informative for unseen datasets. Investigating how network performance changes between datasets and identifying what physical changes lead to performance differences is a direction for future work.

Network Generalization Prediction can be used to link network behavior in novel operating domains to required levels of performance. The Context Subspace can be leveraged for quasi-white box testing by testing the network across variations in context features that are known to impact network behavior. The Context Subspace also makes the network behavior interpretable by elucidating where failure is more likely. In addition to making the Network Generalization Prediction tractable, we believe the Context Subspace can be used during network training to extract features that are robust to changes in the Context Subspace. The Context Subspace can also be used for online error monitoring, e.g., an autonomous vehicle could notify the driver if it detects the surrounding scene is a context with subpar expected performance. We believe the Context Subspace is a tool that can make network performance more interpretable during training, testing, and deployment.

6. Conclusions

We propose the task Network Generalization Prediction and leverage a Context Subspace to render Network Generalization Prediction tractable with scarce test samples. We identify the Context Subspace automatically and demonstrate accurate Network Generalization Prediction for Faster RCNN used for pedestrian detection in diverse operating domains. We show that the Context Subspace identified for the BDD Dataset is informative for completely unseen datasets. We believe that accurate Network Generalization Prediction, with an interpretable Context Subspace, is a step towards bridging the gap between the high performance of deep networks and the verification required for safety critical systems.

References

[1] Algirdas Avizienis, J-C Laprie, Brian Randell, and Carl Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE transactions on dependable and secure computing*, 1(1):11–33, 2004.

[2] Gábor Braun, Sebastian Pokutta, and Yao Xie. Info-greedy sequential adaptive compressed sensing. *IEEE Journal of selected topics in signal processing*, 9(4):601–611, 2015.

[3] Jie Cai, Jiawei Luo, Shulin Wang, and Sheng Yang. Feature selection in machine learning: A new perspective. *Neurocomputing*, 300:70–79, 2018.

[4] Chih-Hong Cheng, Chung-Hao Huang, Harald Ruess, Hirotoshi Yasuoka, et al. Towards dependability metrics for neural networks. In *2018 16th ACM/IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*, pages 1–4. IEEE, 2018.

[5] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[6] Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. Underspecification presents challenges for credibility in modern machine learning. *arXiv preprint arXiv:2011.03395*, 2020.

[7] Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira. Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[8] Jianhao Jiao, Yilong Zhu, Haoyang Ye, Huaiyang Huang, Peng Yun, Linxin Jiang, Lujia Wang, and Ming Liu. Greedy-based feature selection for efficient lidar slam. *arXiv preprint arXiv:2103.13090*, 2021.

[9] Rajiv Khanna, Ethan Elenberg, Alex Dimakis, Sahand Negahban, and Joydeep Ghosh. Scalable greedy feature selection via weak submodularity. In *Artificial Intelligence and Statistics*, pages 1560–1568. PMLR, 2017.

[10] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. Learning not to learn: Training deep neural networks with biased data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9012–9020, 2019.

[11] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Sara Beery, et al. Wilds: A benchmark of in-the-wild distribution shifts. *arXiv preprint arXiv:2012.07421*, 2020.

[12] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 69(6):066138, 2004.

[13] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. *arXiv preprint arXiv:1706.02690*, 2017.

[14] Molly O'Brien, William Goble, Greg Hager, and Julia Bukowski. Dependable neural networks for safety critical tasks. In *International Workshop on Engineering Dependable and Secure Machine Learning Systems*, pages 126–140. Springer, 2020.

[15] Thomas Ponn, Thomas Kröger, and Frank Diermeyer. Identification and explanation of challenging conditions for camera-based object detection of automated vehicles. *Sensors (Basel, Switzerland)*, 20(13), 2020.

[16] Amir Rasouli, Iuliia Kotseruba, and John K Tsotsos. Are they going to cross? a benchmark dataset and baseline for pedestrian crosswalk behavior. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 206–213, 2017.

[17] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

[18] Adarsh Subbaswamy, Roy Adams, and Suchi Saria. Evaluating model robustness to dataset shift. *arXiv preprint arXiv:2010.15100*, 2020.

[19] Saeid Asgari Taghanaki, Mohammad Havaei, Alex Lamb, Aditya Sanghi, Ara Danielyan, and Tonya Custis. Jigsaw-vae: Towards balancing features in variational autoencoders. *arXiv preprint arXiv:2005.05496*, 2020.

[20] Joel A Tropp and Anna C Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on information theory*, 53(12):4655–4666, 2007.

[21] Ioannis Tsamardinos, Giorgos Borboudakis, Pavlos Katsogridakis, Polyvios Pratikakis, and Vassilis Christophides. A greedy feature selection algorithm for big data of high dimensionality. *Machine learning*, 108(2):149–202, 2019.

[22] Jorge R Vergara and Pablo A Estévez. A review of feature selection methods based on mutual information. *Neural computing and applications*, 24(1):175–186, 2014.

[23] Wei Wang, Vincent W Zheng, Han Yu, and Chunyan Miao. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):13, 2019.

[24] Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2251–2265, 2018.

[25] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018.