AI Tool with Active Learning for Detection of Rural Roadside Safety Features

Hong Yi*
Renaissance Computing Institute
University of North Carolina
Chapel Hill, NC
hongyi@renci.org

Chris Bizon
Renaissance Computing Institute
University of North Carolina
Chapel Hill, NC
bizon@renci.org

David Borland
Renaissance Computing Institute
University of North Carolina
Chapel Hill, NC
borland@renci.org

Matthew Watson
Renaissance Computing Institute
University of North Carolina
Chapel Hill, NC
mwatson@renci.org

Matthew Satusky
Renaissance Computing Institute
University of North Carolina
Chapel Hill, NC
satusky@renci.org

Robert Rittmuller
Volpe National Transportation Systems Center
U.S. Department of Transportation
Cambridge, MA
robert.rittmuller@dot.gov

Randa Radwan
Highway Safety Research Center
University of North Carolina
Chapel Hill, NC
radwan@hsrc.unc.edu

Raghavan Srinivasan
Highway Safety Research Center
University of North Carolina
Chapel Hill, NC
srini@hsrc.unc.edu

Ashok Krishnamurthy
Renaissance Computing Institute
University of North Carolina
Chapel Hill, NC
ashok@renci.org

2021 IEEE International Conference on Big Data (Big Data) | 978-1-6654-3902-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/BigData52589.2021.9671360

***Abstract*—Roadway safety, especially in rural areas, is one of the most critical components in transportation planning. In collaboration with North Carolina Department of Transportation (NCDOT), UNC Highway Safety Research Center (HSRC), and DOT Volpe National Transportation Systems Center, UNC Renaissance Computing Institute (RENCI) developed a roadside feature detection solution leveraging multiple convolutional neural networks. The solution used an iterative active learning (AL) computer vision model training pipeline integrated into an AI tool to detect safety features such as guardrails and utility poles in geographically distributed NC rural roads. We utilized transfer learning by adopting the Xception neural network architecture [1] as the feature extraction backbone which was then used in an iterative AL process supported by a web-based annotation tool. The annotation tool not only allowed for the collection of annotations through an iterative AL process for multiple safety features, it also enabled visual analysis and assessment of model prediction performance in the geospatial context. AL techniques were used to direct human annotators to label images that would most effectively improve the model aimed at minimizing the number of required training labels while maximizing the model's performance. The iterative AL process combined with a common feature extraction backbone allowed fast model inference on millions of images in the AL sampling space. This enabled a rapid transition between AL rounds while also reducing the computing requirements for each round. Model feature extraction weights were then fine-tuned in the last round of AL to obtain the best accuracy. Since only about 2.7% of 2.6 million unlabeled images in the AL sampling space contain guardrails, there is a significant class imbalance problem that must be addressed in our AL sampling strategies for the guardrail classification model. In this paper, we present our AI tool processing pipeline and methodology and discuss our AL results and future work. Our AI tool can be used to detect roadside safety features and be extended to also locate them for assessing roadside hazards.**

***Index Terms*—deep learning, convolutional neural networks, transfer learning, active learning, class imbalance**

I. Introduction

Roadside crashes account for a significant portion of traffic fatalities in the United States. In North Carolina, roadway departure crashes occur in predominately rural roads that are geographically dispersed across many miles with narrow lanes and unpaved shoulders despite the low traffic volume. The outcome of a roadway departure crash is a function of many factors including the characteristics of the roadside. Roadside features that affect the severity of a roadway departure crash include longitudinal features such as guardrails, narrow objects or point features such as utility poles, signs, and trees, side slope, and offset distance (from the edge of the roadway) to the roadside objects. Information on the roadside features can be used to derive a roadside hazard rating that ranges from 1 through 7 with 1 being the least hazardous roadside and 7 being the most hazardous roadside [2]. More recently, procedures have been developed to quantify the safety effect of roadside design elements on single-vehicle run off road departure crashes [3].

To identify and implement safety countermeasures to reduce the severity of crashes involving the roadside, state and local agencies need information on roadside features in their vast rural roadway network. Manual safety inspection and widespread data collections on these rural roads is labor intensive and potentially dangerous when conducted in the field. Many State

agencies have video logs of at least a portion of their roadway system. An automated and intelligent AI approach to identify roadside features from video logs can help to facilitate safety initiatives more efficiently and effectively on the state's rural roadway system.

In 2018 and 2019, NCDOT collected video log data for all secondary roads (over 54 thousand miles) in NC, 76% of which are classified as rural roads. Images were acquired by three front facing cameras every 26 feet along the roadways; an example set of three images can be seen in Fig. 1.

Fig. 1. Example annotation of a guardrail from the annotation tool.

Artificial and machine learning techniques, particularly deep learning, have been shown to be highly effective at extracting features from image and video data, including data from traffic video [4]. We have explored the use of these methods by implementing a real-world system for extracting safety related roadside features from this collection of video log data. The implemented system allows transportation safety experts to annotate images with the presence or absence of particular roadside features. This creates a curated data set from which we built deep image classification models based on the Xception neural network architecture [1], which could then be used to infer the presence or absence of features in the remaining unlabeled images.

The rate limiting resource is labeled image data from which to build accurate models. To acquire labeled images most efficiently, we first developed a high-throughput annotation system to rapidly and accurately collect image labels from domain experts. We then implemented an active learning (AL) pipeline [5]–[7] to ensure that the images being labeled were those that were most likely to improve the currently existing model. AL has successfully been applied in image classification [8]–[10], but its iterative nature implies a computational burden. In order to make the iterative AL approach computationally efficient, we implemented a common-backbone approach to our image classification models, in which a pretrained backbone remains fixed, and only feature-specific classification heads are retrained in each AL round.

Most AL approaches employ an uncertainty sampling strategy [11] to select data for labeling along the decision boundary, i.e. where the model is most uncertain. However, these uncertainty sampling based AL approaches do not consider the data distribution characteristics of the unlabeled data and do not work well for data under significant class imbalance. Unfortunately, many safety-related features are rare. For instance guardrails appear in only 2.7% of the video log images. We therefore developed a similarity-based sampling strategy by adapting ideas proposed in the similarity-based AL framework [8] to address our guardrail class imbalance problem.

We applied this tool to detect two features, namely guardrail and utility pole. With only about 1.5% of 2.6 million unlabeled images annotated by domain experts, we achieved 99% accuracy for the guardrail classifier evaluated on a randomly selected holdout test set with over 12K images and 90% accuracy evaluated on a class-balanced holdout test subset with 642 total images. There were no class imbalance issues for the utility pole classification since about 64% of images in the AL sampling space contain utility poles. With only about 0.68% of 2.6 million unlabeled images annotated by domain experts, we obtained 88% accuracy for the utility pole classifier evaluated on a randomly selected holdout test set with 960 total images.

The rest of the paper is organized as follows. Section II includes a brief review of the related research. We then present our AI tool processing pipeline and methodology in Section III with results and discussion presented in Section IV. We discuss future research and conclude the paper in Section V.

II. Related Work

With increasing data availability and computational power especially with the continuous advancement of high performance Graphical Processing Units (GPU), deep learning (DL) convolutional neural networks (CNNs) have shown exemplary performance in image segmentation and classification related tasks in recent years and are one of the best learning algorithms in computer vision [12]. CNN is a feedforward multilayered hierarchical network and learns through backpropagation algorithm during training, where each layer performs multiple transformations using convolutional kernels [13]. The convolutional nature has enabled deep CNN architectures composed of multiple feature extraction stages to learn internal representations from the data automatically. Each feature extraction stage in deep CNN architectures is composed of a combination of convolutional layers, non-linear processing units, and subsampling layers [14] which provides the ability to extract features and learn complex representations at different levels of abstraction. The extracted features in a deep CNN trained for one task can be adaptively transferred to a different task using Transfer Learning (TL) [15]–[18].

TL aims to extract the knowledge from one or more source tasks and applies the knowledge to a target task [16]. There have been a number of popular deep CNN architectures emerging since 2012 with a trend to make the network increasingly deeper, mostly driven by the yearly ImageNet Large Scale Visual Recognition Challenge (ILSVRC) competition [19]. These deep CNN models were pretrained on more than one million curated images from the ImageNet database with high performance and are good candidates to be used for different image classification tasks via TL particularly when the number of training samples is low. For example, for roadside safety applications, Kolar et al. [17] used the core part of the deep CNN VGG-16 model [20] to transfer the image features stored in the VGG-16 model to their guardrail detection model. Abodo et al. [21] selected MobileNetV2 [22] via TL for

work zone detection. Kačan et al. [23] experimented both ResNet-18 [24] and DenseNet-121 [25] via TL for a multi-class classification of 33 road safety attributes and reported that using DenseNet-121 resulted in better accuracy on their validation dataset than using ResNet-18.

Although TL enables training a more accurate and effective CNN model for a target task with much less labeled data and training time, collecting labeled data representative of the target data distribution is a time-consuming and costly process. Active Learning (AL) [5]–[7] attempts to collect a small amount of data for annotation and build models in an iterative and efficient way. In each round of AL, the collected annotations are used to build a preliminary model, which is used to make predictions in the AL data sampling pool and to select the most informative samples with the presumption that those samples have the best chance of broadening the existing labeled instance distribution. These samples are then presented to users for labeling with the goal of achieving high accuracy using as few labeled instances as possible.

Strategies for effectively selecting samples for labeling is at the core of AL. Most AL algorithms employ the uncertainty sampling principle [26]. Some examples of AL sampling methods include strategies with a combination of high and low prediction confidence uncertainty sampling [27], Bayesian uncertainty estimates [28], and choosing exemplar subsets [29]. Sun et al. [30] incorporated an adversarial network into AL which performed well on datasets with out of distribution instances. In addition, some methods (e.g., [31]) do not rely on a fixed network architecture, but rather search for effective architectures in the AL loop.

Unbalanced learning is required in many applications such as fraud detection, medical image analysis for disease detection, and guardrail or work zone detection, where the training data for a binary classification problem contains many more instances of one class than the other. This class imbalance creates problems in labeling data as finding a sufficient number of rare class instances becomes difficult; it often results in models that are biased towards the common class which adversely affects the usefulness of assessment metrics. Branco et al. [32] presented a survey of unbalanced learning that discussed the main challenges raised by imbalanced data distributions and approaches to address these challenges. In addition, Johnson and Khoshgoftaar [33] presented a survey on deep learning with class imbalance. Although several classic machine learning AL approaches have been proposed to address imbalanced data (e.g., [34]), most deep learning AL approaches don't take severe class imbalance into consideration. Recently, approaches for applying AL to image classification problems with class imbalance have been developed [8]–[10]; these approaches select new samples for labeling in the extracted feature embedding space focusing on selecting samples most like the rare class and least like the common class. For example, a similarity-based active deep learning (SAL) framework for image classification under class imbalance was proposed [8]. SAL actively learns a similarity model to select unlabeled rare class samples for manual labeling while recommending with high confidence unlabeled common class samples for automatic pseudo-labeling based on similarity ranking. In our AL approach, we developed a similarity-based sampling strategy by adapting the rare class and dissimilar sample selection ideas proposed in the SAL framework to address our guardrail class imbalance problem.

III. Methodology

Fig. 2 shows our iterative process to create a training data set and a predictive image classification model to extract safety-related roadside features from video log data. First, we trained a baseline model using Xception architecture [1] via TL by leveraging the primary road guardrail assessment data NCDOT collected; second, we used this baseline model to extract features in a common feature extraction backbone and make predictions for millions of unlabeled secondary road images in the AL sampling space; third, we computed most informative samples for AL based on model predictions and extracted features; fourth, these samples were fed into the annotation tool to collect expert annotations for them; fifth, annotations were exported from the annotation tool and used to train a new model as part of AL; sixth, the new model was used to make fast predictions with a common feature extraction backbone and the new model predictions were fed into AL sampling in step 3. The AL loop continues until the new model has reached our target performance metrics. Last, the weights in the final AL round model including the feature extraction backbone were fine-tuned to obtain the best model accuracy. Through this AL process, we evaluated the accuracy of each AL round model on a balanced holdout test set. We reached the 90% accuracy for the final guardrail model starting from the initial 75% accuracy and the 88% accuracy for the final utility pole model starting from the initial 66% accuracy. We describe our AI tool processing pipeline and methodology in details next.

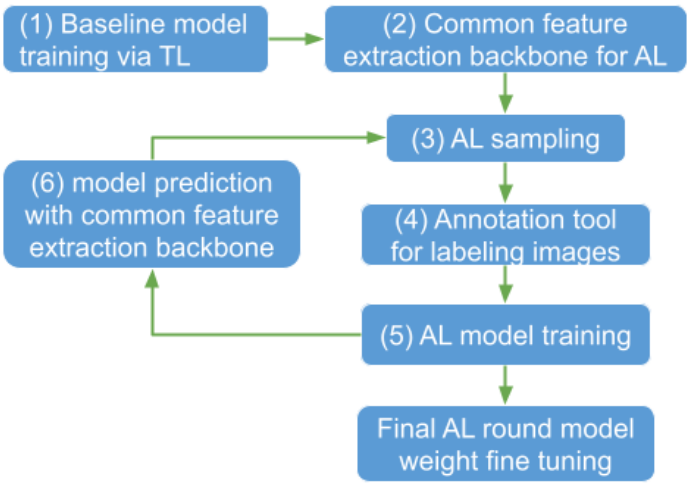

Fig. 2. Our AI tool iterative processing pipeline.

A. Baseline model training via transfer learning

In 2017, NCDOT collected guardrail assessment data manually extracted from the primary road video log data. This as-

sessment data provided valuable information for us to prepare labeled primary road images for training a guardrail model which was then used as a baseline model for subsequent AL on rural 2-lane secondary road data. Since the assessment data associated guardrail segments by start and end geographic locations, we mapped video log images to guardrail segments by locations and prepared a total of over 1.3 million labeled primary road images for training a guardrail model with half of them positive containing guardrails and the other half negative. Since NCDOT has been mainly concerned with the two-lane only rural secondary road safety features, we also prepared a labeled subset of the primary road images that were two-lane only to make training data align more with the target data distribution. The labeled two-lane only subset included $282,452$ total images evenly balanced between the positive and negative classes accounting for about 21% of the total labeled primary road image set. Since more training data usually leads to improved model performance by better addressing overfitting for high variance models, we trained a guardrail model on both the full data set and the two-lane only subset so that we could compare the performance of two models and select a better performing model as the baseline model. Specifically, for the full data set, 98% were randomly selected as training data evenly split between the positive and negative classes with 1% used as validation and the other 1% used as test data. For the two-lane only subset, 90% were randomly selected as training data evenly split between two classes with 5% used as validation and the other 5% used as test data. Although train-validation-test split ratio rules such as 80%-10%-10% and 60%-20%-20% are commonly used in machine learning, a much smaller percentage such as 1% for validation and test sets is sufficient and commonly used when there are a million or more labelled samples.

We used the Xception [1] network pre-trained on ImageNet to train guardrail models via TL. Xception was a deep CNN architecture from Google inspired by its predecessor Inception, where Inception modules have been replaced with depthwise separable convolutions. Xception architecture has the same number of parameters as Inception V3, yet has better performance than Inception V3 due to a more efficient use of model parameters. Chollet [1] showed Xception slightly outperformed Inception V3 on the ImageNet dataset which Incpetion V3 was designed for, and significantly outperformed Inception V3 on a larger image classification dataset comprising 350 million images and $17,000$ classes.

We used TensorFlow (TF) [35] 2 as the DL framework which includes Xception architecture with pre-trained weights that can be directly loaded in our application. We replaced the top layer of the Xception with three stacked layers: a Global Average Pooling layer to reduce the number of parameters by 100 fold, followed by a dropout layer with a rate of 0.25 to help prevent overfitting, and a fully connected layer on the top with a sigmoid activation for binary classification with the binary cross-entropy as the loss function. We used the Adam [36] optimizer with a learning rate of 0.001 initially to train the top classification layer only for 10 epochs which improved the model accuracy from the initial 54% to about 91%. Then we kept the bottom two block layers frozen and opened up the top 12 block layers of Xception for fine tuning the pre-trained weights with a very low learning rate of $1e-5$. In addition, we used the model checkpoint callback supported by TF to monitor validation loss during training to only save the best performing model with minimal validation loss at the end of each epoch. Fig. 3 shows the accuracy and loss plots of the guardrail models trained on full data and 2-lane only subset data during fine tuning, from which we can see that the full data model started overfitting after epoch 7 (top) and the 2-lane only model started overfitting after epoch 10 (bottom). As such, we selected epoch 7 full data model and epoch 10 2-lane only model as the baseline model candidates.

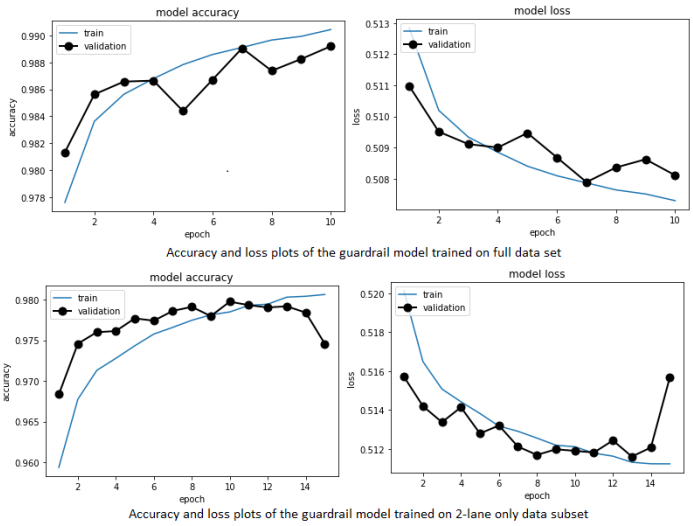

Fig. 3. Accuracy and loss plots of the guardrail models trained on full data (top) and 2-lane only data (bottom) during fine tuning.

We evaluated two model candidates on the 2-lane only balanced primary road test set which aligned more with the secondary road target set. Two model candidates had similar performance although the 2-lane only model had a little better accuracy 97.9% than 97.7% of the full data model. In addition, we compared predictions of two models on a NC eastern region of $741,476$ images in the secondary road target set and found two models had same predictions for 98% of these images. We manually inspected a subset of 300 randomly selected images, with 100 out of the whole set, 100 out of the 98% common prediction set, and 100 out of the 2% different prediction set, and compared the receiver operating characteristic (ROC) area under curve (AUC) scores of both model predictions on the subset. Both manual inspection and AUC scores indicated the 2-lane data model performed a little better (0.903 AUC score for the 2-lane data model vs. 0.853 for the full data model). As a result, we selected the 2-lane data model as our baseline guardrail model for subsequent AL.

B. Common feature extraction backbone for AL

Our objective is to create safety feature models via AL to make predictions in our unlabeled 2-lane secondary road

target set (over 40 thousand miles) composed of 14 divisions across NC. NCDOT transportation safety experts selected 4 geographically representative divisions in the target set that we used to create the AL sampling image space. There were about 2.6 million joined image samples in the AL sampling space from which we selected most informative samples for manual labeling based on model predictions from the previous AL round aimed at achieving high model accuracy using as few labeled instances as possible.

To ensure a smooth transition between AL rounds, predictions of the model from one AL round over millions of images in the AL sampling space must be finished in a reasonable amount of time. We conducted AL on a server with a 2 20-core 2.10GHz Intel Zeon CPUs, 700GB of RAM, and two NVIDIA Tesla V100 graphics cards with 32GB of RAM each. The server ran CentOS 7.9 operating system with NVIDIA driver 440.33, CUDA 10.2, cuDNN 7.6.5, and tensorflow 2.3. With image prefetching and caching implemented to significantly improve the model training and inference speed, it still took about one week to finish model inference over millions of images in the AL sampling space, which was too long to meet our one-week-per-AL-round requirement.

To address this AL performance issue, we employed a common backbone approach through the iterative AL process. In particular, feature extraction from input images was performed by a common backbone network in a single pass and shared across AL rounds. Our common backbone was comprised of a set of convolution and polling layers in the Xception architecture to produce a feature map containing higher-level summary information of images. Throughout the AL rounds, only the top fully connected classification layer was trained with 2048 input parameters from the shared common backbone and one output parameter for binary classification. This shared common backbone approach dramatically improved AL model prediction performance, reducing prediction time from about one week to under one hour. In addition, having a shared common feature extraction backbone across the AL rounds allowed for methods of effective AL sampling under class imbalance via analysis of similarities of extracted image feature vectors in the feature embedding space. We present the details of such an AL sampling approach in section III-C3.

C. Active Learning sampling strategies

Our goal for AL sampling strategies was to sort images for human annotation in the order of likelihood that the manually labeled images would contribute most to improve the model performance. Specifically, an offline process computed an uncertainty score for each image sample that corresponded to its sampling order determined by an AL sampling strategy. These uncertainty scores were then ingested into the annotation tool and used to return images for annotation in the order intended by the sampling process. We used the baseline model as described in section III-A to predict guardrail probabilities of unlabeled images in our AL sampling space before we started the AL process.

The objective of AL is to minimize the number of instances that require human annotations while still producing a model with satisfactory performance, which can be achieved via effective sampling or query strategies. Most AL approaches employ the uncertainty sampling strategy [11] to select data along decision boundary where the model is most uncertain about for labeling. However, these instances along decision boundary may not be representative of the unlabeled data pool and may sometimes lead to the selection of unrepresentative outliers or too many common class instances in data with significant class imbalance. To address these issues, some other heterogeneity-based sampling strategies may be employed to sample from instances that are dissimilar to what has already been sampled [5]. For example, the query-by-committee sampling approach [37] uses a committee of different classifiers to predict the class label of each unlabeled instance and selects those instances for which the classifiers disagree most. Different strategies have different trade-offs depending upon the underlying application and data distribution. We employed uncertainty sampling and the query-by-committee strategies to fit better with our application and data distribution in conjunction with a similarity-based sampling strategy we developed by adapting ideas proposed in the similarity-based AL framework [8] to address the significant class imbalance problem for guardrail classification. We describe our AL sampling strategies for guardrail and utility pole classifications below.

1) Uncertainty sampling: Before we started the AL process for the guardrail feature, we conducted preliminary analysis with manual inspection of the baseline model predictions on the secondary road holdout test set which revealed there were many false positives (FPs). The scatter plot in Fig. 4 shows distribution of prediction probabilities and where the wrong predictions lie in the decision space, in which blue and gray empty dots represent ground truth positive and negative instances, respectively, while red filled dots represent wrong model prediction instances. As a result, for the first round of guardrail feature AL, we sampled in the positive guardrail prediction set with uniform distribution across the geographic regions. Specifically, we randomly sampled from the positive prediction set five images at a time for each geographic region in turn and assigned an uncertainty score for each sample to correspond to its sampling order. This data specific uncertainty sampling was aimed at collecting more potential FP images to improve the model performance while making the image samples as geographically representative as possible.

Another generic uncertainty sampling strategy we used for both guardrail and pole features is to apply weighting towards the most uncertain predictions along the model decision boundary while ensuring good distribution across different geographical regions. Specifically, given a decision boundary b and a model prediction p_i for each image i, we calculated the distance to the decision boundary as $D_i = |p_i - b|$. The certainty score was then computed as $Z_i = D_i + N_i(0, s)$ where $N_i(0, s)$ denoted values sampled from a normal distribution of mean 0 and standard deviation s and s deter-

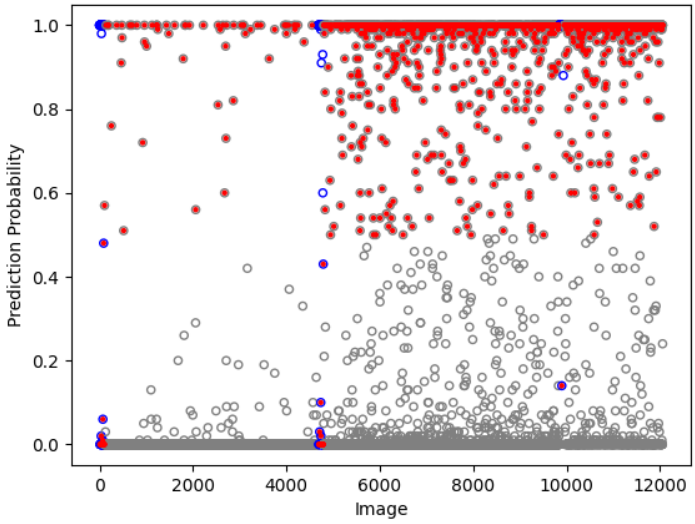

Fig. 4. Scatter plot for guardrail baseline model predictions on holdout test.

mined how far away from the decision boundary we were sampling. Sorting certainty score Z_i for all images in an ascending order determined the image order for AL sampling which naturally shuffled images clustered around the decision boundary b across and within different geographical regions. To determine b, we examined the precision-recall-threshold plot together with the ROC curve of the previous AL round model prediction on the balanced holdout test set and chose the cutoff that resulted in the most appropriate trade-off between precision and recall. To consider model prediction differences between regions, our AL sampling algorithm allowed to set different b and s per region to compute the certainty score Z_i.

2) Query-by-committee sampling: We performed image segmentation for utility poles using the ADE20k segmentation model [38], [39] on $30,000$ randomly selected joined images ($90,000$ left, front, and right single view images) across eastern, central, and western regions. The model was deemed to have predicted the presence of a utility pole if the segmentation map contained pixels of the ADE20k pole class. Using 100 randomly-selected front view images, we determined the HRNetV2 architecture to be the most accurate of the seven ADE20k segmentation architectures at around 72%; this accuracy was reproduced using the larger holdout test set. For each joined image, the three component images were processed individually and presence of a pole in any component image indicates its presence in the joined image. We also performed inference for utility poles on the same set of randomly selected images using the previous AL round pole model trained with the common feature extraction backbone model architecture. The query-by-committee approach was used, in which the committee was composed of these two model classifiers, to sample those images where these two models had different prediction classes.

3) Similarity-based sampling: we developed a similarity-based sampling strategy by adapting the rare class and dissimilar sample selection ideas proposed in the SAL framework [8] to take into account the extreme class imbalance for guardrail classification. Similar to the SAL framework, our sampling strategy attempted to sample more rare class samples effectively as well as images dissimilar from those already in the training dataset by sorting similarity scores computed in the feature embedding space. Specifically, in the 2048 dimension feature embedding space, we first computed the feature vector centroids $\bar{v_p}$ and $\bar{v_n}$ of positive (rare) and negative (common) image sets in the existing training data. Then we computed the cosine similarity scores SP_i and SN_i of each unlabeled image i to each of the two centroids $\bar{v_p}$ and $\bar{v_n}$. Based on SP_i and SN_i, we sampled instances with a uniform mixture of images from the positive (rare) class set Set_p and images dissimilar from those already in the training dataset Set_{ds}. To compose Set_p, we sorted images by SP_i in the descending order and selected the top $20,000$ images to ensure we sampled enough rare class images. To compose Set_{ds}, we initially computed a new similarity score by taking the larger value of SP_i and SN_i for each image i as $MaxS_i = max(SP_i, SN_i)$, sorted images by $MaxS_i$ in the ascending order, and selected the top $20,000$ images. However, this algorithm resulted in many noisy outlier images being selected which did not represent the real distribution of the new data. We applied some basic image processing techniques to filter out those noisy outlier images with extreme over- or under-exposure, however, filtering out all noisy outlier images without losing informative ones was tricky. To alleviate this problem, we adapted the algorithm by first constructing a selection set with those images whose SP_i and SN_i were both less than a decision boundary threshold 0.5, then computing a new similarity score by taking the smaller value of SP_i and SN_i for each image i in the selection set as $MinS_i = min(SP_i, SN_i)$, sorting images by $MinS_i$ in the ascending order, and selecting the top $20,000$ images. This adapted algorithm captured those unlabeled images of high entropy (most uncertainty) and diversity with less chance of selecting noisy outlier images. The uniform mixture of Set_p and Set_{ds} were then selected for manual annotation.

Fig. 5 shows the t-SNE visualization [40] of those $39,984$ images sampled using the similarity-based strategy for the fourth round of AL for the guardrail feature in the 2048 high-dimensional feature embedding space, where the orange and blue clusters represent samples from Set_p and Set_{ds}, with the green and red dots representing their centroids $\bar{v_p}$ and $\bar{v_n}$, respectively. Fig. 6 shows the 4860 human annotated images with 2500 from Set_p and 2360 from Set_{ds}, overlaid on the two clusters with non-annotated images filtered out, where blue and orange dots represent positive and negative annotations, respectively. Our analysis showed that all 2500 samples from Set_p were indeed in the positive (rare) class consistent with AL model predictions; the two negative annotations corresponding to the two orange dots in the blue cluster shown in Fig. 6 were in fact wrong annotations. Although sampling these images from Set_p generated sufficient rare class instances, they do not provide new information to the model. Further cluster analysis of images with false positive (FP) and false negative (FN) predictions revealed that these images tend

to cluster around regions where similarity scores disagree with prediction probabilities, which appeared to indicate decision boundaries in the feature embedding space. As a result, we also employed a sampling strategy to leverage both SP_i and SN_i as well as the prediction probabilities attempting to sample images around the decision boundaries where similarity scores disagreed with prediction probabilities.

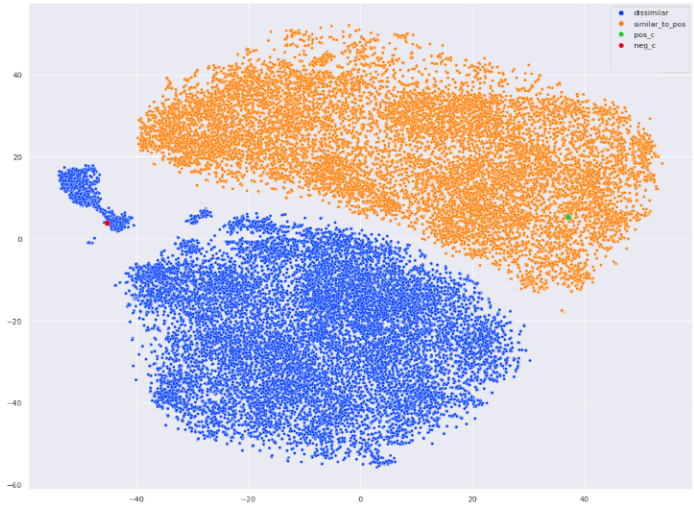

Fig. 5. Visualization of AL sampled images in the feature embedding space.

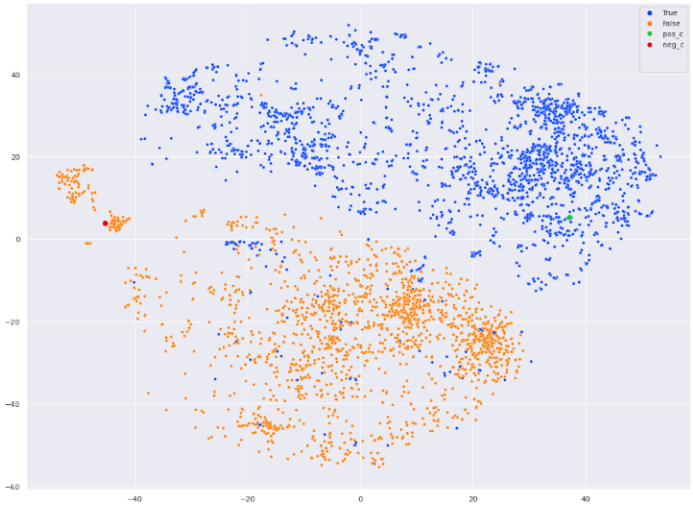

Fig. 6. Visualization of clusters with annotation overlay in the feature embedding space.

D. Annotation tool for labeling images in AL

We developed a web-based tool, Roadway Hazard Finder (RHF), to collect annotations from transportation safety experts to support AL. As a central component of our AI system, RHF was tightly integrated with our offline DL/AL pipeline, enabling easy ingestion of image samples selected from offline AL into the tool for annotation, and uploading of collected annotations from the tool for offline AL model training. The primary annotation interface received images for annotation prioritized by the AL system, and provided a simple point and click interface for annotating the images based on the current feature of interest (fig. 1).

RHF also provided diagnostic interfaces, such as a route browser (fig. 7) that enabled the user to virtually drive a particular route while visualizing a plot of all predictions and annotations along the route, and a prediction error table (fig. 8) that enabled the user to organize and review any discrepancies between the model predictions and user annotations.

Fig. 7. Route Browser with scatterplot depicting predictions and annotations.

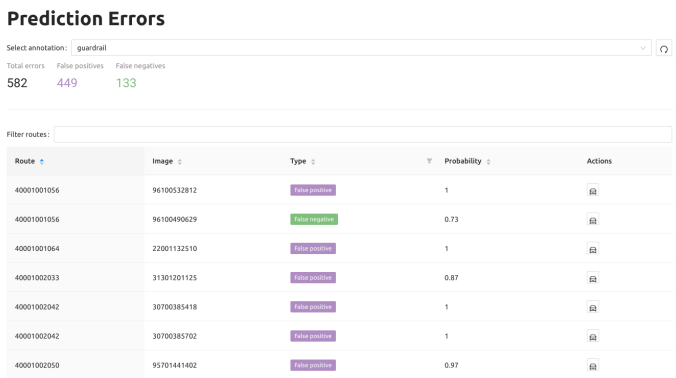

Fig. 8. Prediction errors table.

RHF was designed to be versatile and easily re-purposed for other similar tasks. For example, we used RHF to annotate randomly selected images across representative NC regions in our AL sampling space to create holdout test sets for assessing guardrail and utility pole models through the AL process.

E. AL model training

We used the common feature extraction backbone extracted from the baseline guardrail model through the iterative AL model training. Specifically, only the top fully-connected classification layer with 2049 parameters were trained through the AL rounds with the weights in the shared common feature extraction backbone frozen. We randomized initial weights from a uniform distribution and reset biases to zeros for the top classification layer before training for each AL round which we found was critical for good model convergence.

The baseline guardrail model was trained on joined images of left, front, and right views since the guardrail assessment data we used to prepare training data could only be mapped to joined images by geographic locations. On the other hand, our captured labeled data included annotations for each single view image, enabling us to train AL models on single view images. Since we resized our input images with varying resolutions to the original input image size $299X299$ for the Xception architecture, training on joined images would have threefold increase in likelihood that some thin, short, faraway features could disappear due to image resizing, which would increase FNs in model predictions and challenge models with the conflicting information between the annotations and resized images fed to models. As a result, we used collected annotations to prepare single view images for training AL models.

At each AL round, we trained the model using all annotations collected so far randomly split into 80% for training and 20% for validation, which created unbalanced training and validation data. Some commonly used approaches such as over-sampling rare class or under-sampling common class instances change the class distributions and don't work well for our application. Instead, we adjusted class weights that each training instance carried when computing the loss by giving rare class instances more weight than the common class instances. Specifically, we used the *compute_class_weight* function in *scikit-learn* [41] to compute weights for the two classes and pass the computed weights to the *fit* function in TF for model training. We also used early stopping callback supported by TF to stop training when the validation loss did not improve for 10 epochs.

F. Final model weight fine tuning

Although the common feature extraction backbone was used through AL to allow fast model training and inference through the AL process enabling quick transition between AL rounds, it did set an upper limit for final model accuracy due to less than ideal weights for the common backbone. Specifically, since the common backbone was extracted from the baseline guardrail model which was trained on primary road joined images, the weights in the common backbone were not satisfactory for the secondary road single view image target set. By fine tuning weights for the common feature extraction backbone for the final round model of AL, we improved the guardrail model accuracy from 84% to 90% evaluated on the balanced holdout test set and the pole model accuracy from 72% to 88%. Although fine tuning weights for the final model was justified by the big model accuracy improvement, a better designed common feature extraction backbone might be beneficial and remove the need for weight fine tuning.

IV. Results and Discussion

Figures 9 and 10 show ROC curves of the guardrail and utility pole models through AL evaluated on the respective balanced holdout test sets. As shown in the ROC curves, the final round of models with fine-tuned feature extraction weights had significant performance improvement over the AL models trained with the shared common feature extraction backbone. In addition, there was a big performance improvement in the first AL round guardrail model over the baseline model followed by minor performance improvement in subsequent AL rounds. Similarly, there was minor model performance improvement over AL rounds for the pole model. One reason that hindered the larger performance improvement over AL rounds might be the shared common feature extraction backbone we used for AL which set an upper performance limit evidenced by the significant performance improvement when unfreezing weights from the common backbone for fine tuning.

To gain more insight into the AL performance limit potentially resulting from using the shared common backbone, we experimented unfreezing weights for fine tuning from the common backbone of the round 1 and round 3 of AL guardrail models to see how performance would improve over AL rounds without using the shared backbone weights. As shown in Fig. 11, there was a much bigger performance improvement from fine tuned AL round 1 model (AUC: 0.897) to the fine tuned AL round 3 model (AUC: 0.947) than their counterparts with the shared backbone weights. This experiment demonstrated that the performance of overall AL models without using the shared common backbone did improve across AL rounds. Although using the common feature extraction backbone through the iterative AL process enabled a rapid transition between AL rounds, this efficiency came at the cost of model accuracy. To overcome this trade-off, we used the shared backbone approach only during the iterative AL process while unfreezing weights from the common backbone to fine tune all weights for the final model. An open research question we are interested in exploring is whether the selected samples for human annotations in AL would be similar regardless whether the shared backbone were used or not.

Table I shows the number of annotations we collected for guardrail and utility pole features over AL rounds. With only about 1.5% of 2.6 million unlabeled images annotated by domain experts, we achieved 99% accuracy for the guardrail classifier evaluated on a randomly selected whole holdout test set with $12,057$ images and 90% accuracy evaluated on the balanced holdout test subset with 642 total images. On the other hand, we don't have class imbalance issues for the utility pole classification since about 64% of images in the AL sampling space contain utility poles. With only about 0.68% of 2.6 million unlabeled images annotated by domain experts, we obtained 88% accuracy for the utility pole classifier evaluated on a randomly selected holdout test set with 960 total images. We also further evaluated our final models using all annotations we collected and verified model performance were in line with what we obtained from the holdout test sets as shown in Tables II and III.

To better understand model prediction performance, we used our annotation tool to inspect those FP and FN images in both holdout test and collected annotation data. Excluding a small proportion of wrongly classified FPs and FNs due to human annotation errors, we observed that for the guardrail model,

FPs were mainly composed of guardrail-like objects (e.g., fences, construction barriers, ditches, curbs, and snow) and irrelevant guardrails on a different road; for the pole model, FPs were mainly composed of pole-like objects such as trees and irrelevant poles that were too far away from the road to be treated as a safety hazard and thus annotated as negative. FNs were mainly composed of those thin, small, or faraway features that were missed by the models most likely due to the image resizing. Some image preprocessing techniques such as intelligent cropping or partitioning images into small tiles to feed into CNN without resizing could be explored to reduce FNs by addressing the feature disappearance issue.

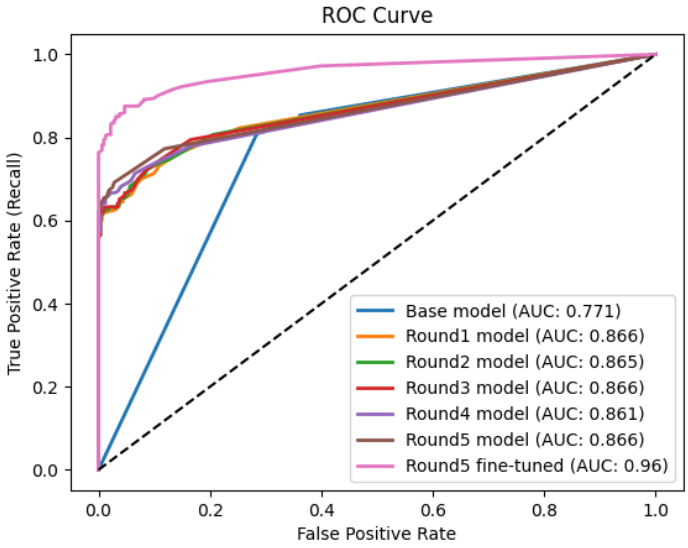

Fig. 9. ROC curve of guardrail models through AL.

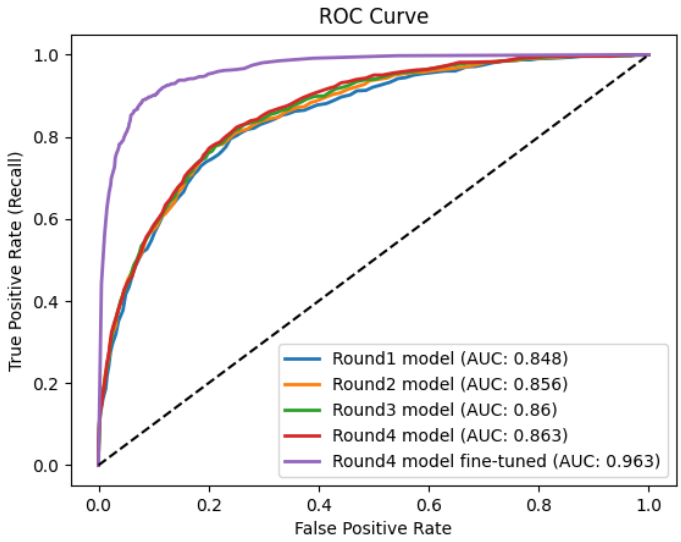

Fig. 10. ROC curve of pole models through AL.

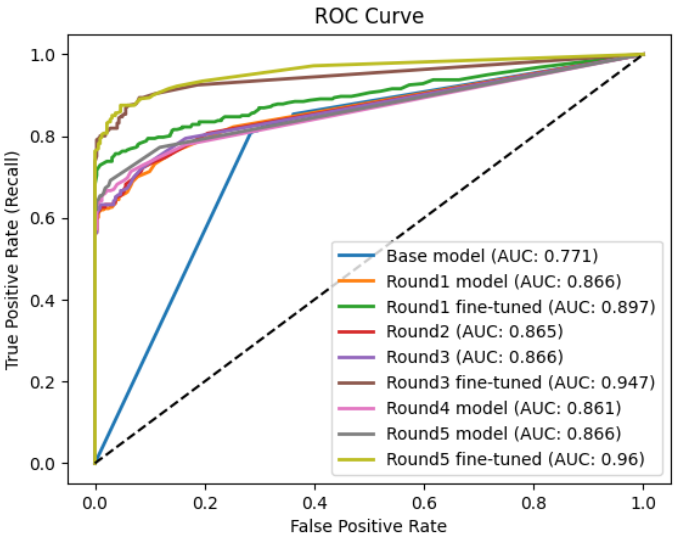

Fig. 11. ROC curve of guardrail models through AL including round 1 and round 3 model counterparts with fine tuned weights without using the shared common backbone weights.

TABLE I
COLLECTED ANNOTATION COUNT INFO

Feature	AL Round					
	1	2	3	4	5	Total
guardrail	9980	11340	7490	4860	6610	40280
pole	3198	5340	3340	5821	N/A	17699

V. CONCLUSION

In this paper we have presented an AI tool for detection of rural roadside safety features from existing video log data. We discussed the utilization of transfer learning to extract features to support the use of a common feature extraction backbone which was then integrated into an iterative AL process. We also presented our web-based annotation tool which not only allowed us to collect annotations effectively throughout the AL process, but also enabled visual analysis and assessment of model prediction performance in the geospatial context. As

TABLE II
MODEL PERFORMANCE INFO EVALUATED ON COLLECTED ANNOTATION

Feature	Model performance				
	total	errors	FPs	FNs	accuracy
guardrail	40280	567	436	131	99%
pole	17699	1861	1431	430	89.5%

TABLE III
MODEL PERFORMANCE INFO EVALUATED ON HOLDOUT TEST SETS

Feature	Model performance				
	total	errors	FPs	FNs	accuracy
guardrail	12057 (whole)	152	89	63	99%
guardrail	642 (balanced)	67	4	63	90%
pole	960	117	58	59	88%

the first step toward enabling automated or partially automated assignment of safety ratings for the road segment being considered, our current approach indicates if a given roadway image has certain safety features of interest or not. Although this type of binary classification is useful for continuous safety features such as guardrails, it is not very useful for point features such as utility poles from a road safety perspective without knowing the geometric distances of the point feature in relation to the camera and the roadway. Future next steps are to extend this approach by adding more safety-related features of interest to be recognized along with geometric distances of the features to the camera and the roadway. While the use of recurrent neural networks (RNNs) [42] can be used to improve the detection accuracy of continuous features, additional future development would explore the use of monocular depth estimation [43] combined with GIS roadway data to assist in tackling the difficult problem of point object geolocation. The ultimate goal with the current AI tool is to provide a detailed geometric description of the road and area immediately surrounding the roadway to unlock predictive insights from video log data for rural road safety.

The source code repository for our AI tool with the entire AL pipeline integrated is publicly available at https://github.com/renci/ncdot-road-safety. Our guardrail and utility pole models can also be downloaded from the github repository to be used directly or as pretrained models for TL.

ACKNOWLEDGMENT

This work was supported by USDOT as part of its Safety Data Initiative.

REFERENCES

[1] F. Chollet, "Xception: deep learning with depthwise separable convolutions," Proc. IEEE CVPR, vol. 1, 2017, pp. 1800–1807.
[2] C.V. Zegeer, J. Hummer, D. Reinfurt, L. Herf, and W. Hunter, "Safety effects of cross-section design for two-lane roads," FHWA-RD-87/008, Washington DC, U.S. Department of Transportation. 1987.
[3] "Update of Crash Modification Factors for the Highway Safety Manual," Unpublished Draft Report from National Cooperative Highway Research Program (project no: 17-72), Transportation Research Board, 2021.
[4] B. Verma, L. Zhang, and D. Stockwell, "Roadside video data analysis: Deep Learning," Springer, 2017.
[5] C. Aggarwal, X. Kong, Q. Gu, J. Han, and P. Yu, "Active learning: a survey," Data Classification: Algorithms and Applications, CRC, 2014.
[6] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models," J. Artif. Intell. Res., vol. 4, 1996, pp. 129–145.
[7] B. Settles, "From theories to queries: active learning in practice," Active Learning and Experimental Design Workshop, 2011.
[8] C. Zhang, W. Tavanapong, G. Kijkul, J. Wong, P. Groen, and J Oh, "Similarity-based active learning for image classification under class imbalance," Proc. IEEE ICDM, 2018, pp. 1422–1427.
[9] A Smailagic, H. Noh, A. Campilho, P. Costa, D. Walawalkar, et. al., "MedAL: accurate and robust deep active learning for medical image analysis," Proc. IEEE ICMLA, 2018, pp. 481–488.
[10] A Smailagic, P. Costa, A. Gaudio, K. Khandelwal, M. Mirshekari, et. al., "O-MedAL: online active deep learning for medical image analysis," Data Mining and Knowledge Discovery, vol. 10, issue 4, 2020.
[11] D. Lewis and J. Catlett, "Heterogeneous uncertainty sampling for supervised learning," Proc. ICML, 1994, pp. 148–156.
[12] A. Khan, A. Sohail, U. Zahoora, and A. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," Artificial Intelligence Review, vol. 53, 2020, pp. 5455–5516.
[13] Y LeCun, K. Kavukcuoglu, C Farabet et al, "Convolutional networks and applications in vision," Proc. IEEE ISCAS., 2010, pp. 253–256.
[14] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y LeCun, "What is the best multi-stage architecture for object recognition?" Proc. IEEE ICCV, 2009, pp. 2146–2153.
[15] A Qureshi and A Khan, "Adaptive transfer learning in deep neural networks: wind power prediction using knowledge transfer from region to region and between different task domains," Computational Intelligence, vol. 35, 2019, pp. 1088–1112.
[16] S. Pan and Q. Yang, "A survey on transfer learning," IEEE Transactions on Knowledge and Data Engineering. vol. 22, 2010, pp. 1345–1359.
[17] Z. Kolar, H. Chen, and X. Luo, "Transfer learning and deep convolutional neural networks for safety guardrail detection in 2D images," Automation in Construction, vol 89, 2018, pp. 58–70.
[18] I. Brkić, M. Miler, M. ˇSevrović, and D. Medak, "An Analytical Framework for Accurate Traffic Flow Parameter Calculation from UAV Aerial Videos," Remote Sens. 12 (22), 2020, 3844.
[19] O. Russakovsky, J. Deng, H. Su, J. Krause, et al., "ImageNet Large Scale Visual Recognition Challenge," International J. of Computer Vision, vol. 115, 2015, pp. 211–252.
[20] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Proc. ICRL. 2015, pp. 1–14.
[21] F. Abodo, R. Rittmuller, B. Sumner, and A. Berthaume, "Detecting work zones in SHRP 2 NDS Videos using deep learning based computer vision," Proc. IEEE ICMLA, 2018, pp. 679–686.
[22] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," IEEE CVPR, 2018, pp. 4510–4520.
[23] M. Kačan, M Oršić, S. Šegvić, and M. Ševrović, "Multi-Task Learning for iRAP Attribute Classification and Road Safety Assessment," Proc. IEEE ITSC, 2020, pp. 1–6.
[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," CoRR, vol. abs/1512.03385, 2015.
[25] G. Huang, Z. Liu, L. Maaten, and K. Weinberger, "Densely connected convolutional networks," IEEE CVPR, 2017, pp. 2261–2269.
[26] W. Beluch, T. Genewein, A. Nürnberger, and J. Köhler, "The power of ensembles for active learning in image classification," IEEE CVPR, 2018, pp. 9368–9377.
[27] K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin, "Cost-effective active learning for deep image classification," IEEE Trans. on Circuits and Systems for Video Technology, vol. 27, no. 12, 2017, pp. 2591–2600.
[28] Y. Gal, R. Islam, and Z. Ghahramani, "Deep Bayesian active learning with image data," Proc. ICML, vol. 70, 2017, pp. 1183–1192.
[29] O. Sener and S. Savarese, "Active learning for convolutional neural networks: a core-set approach," Proc. ICLR, 2018.
[30] C. Sun, H. Sun, and X. Liu, "Robust adversarial active learning with a novel diversity constraint", Proc. IEEE BigData, 2020, pp. 226–231.
[31] Y. Geifman and R. El-Yaniv, "Deep active learning with a neural architecture search," Proc. NIPS, 2019, pp. 5974–5984.
[32] P. Branco, L. Torgo, and R. Ribeiro, "A survey of predictive modeling on imbalanced domains," ACM Comput. Surv., 49 (2), 2016, pp. 1–50.
[33] J. Johnson and T. Khoshgoftaar, "Survey on deep learning with class imbalance," J. Big Data, vol. 6, issue 27, 2019, pp. 1–54.
[34] S. Ertekin, "Adaptive oversampling for imbalanced data classification," Information Sciences and Systems, 2013, pp. 261–269.
[35] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, et al. "Tensorflow: a system for large-scale machine learning," OSDI, 2016, pp. 265–283.
[36] D. Kingma and J. Ba, "Adam: a method for stochastic optimization," Proc. ICLR, 2015, abs/1412.6980.
[37] H. S. Seung, M. Opper, and H. Sompolinsky, "Query by committee," ACM Workshop on Computational Learning Theory, 1992, pp. 287–294.
[38] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ADE20K dataset," CVPR, 2017, pp. 633–641.
[39] B. Zhou, H. Zhao, X. Puig et al. "Semantic Understanding of Scenes Through the ADE20K Dataset," Int J. Comput. Vis., vol 127, 2019, pp. 302–321.
[40] L. Maaten and G. Hinton, "Visualizing data using t-SNE," JMLR, vol. 9, no. 86, 2008, pp. 2579–2605.
[41] Pedregosa et al. "Scikit-learn: Machine Learning in Python," JMLR 12, 2011, pp. 2825–2830.
[42] Q. Zou, H. Jiang, Q. Dai, Y. Yue, L. Chen, and Q. Wang, "Robust Lane Detection from Continuous Driving Scenes Using Deep Neural Networks," IEEE Trans. Veh. Technol., 69 (1), 2019. pp. 41–54.
[43] C. Godard, O. Aodha, G. Brostow, "Unsupervised Monocular Depth Estimation with Left-Right Consistency," CVPR, 2017, pp. 270–279.