Constructive Interpretability with CoLabel: Corroborative Integration, Complementary Features, and Collaborative Learning

Abhijit Suprem
School of Computer Science
Georgia Institute of Technology
Atlanta, USA
asuprem@gatech.edu

Sanjyot Vaidya
School of Computer Science
Georgia Institute of Technology
Atlanta, USA

Suma Cherkadi
School of Computer Science
Georgia Institute of Technology
Atlanta, USA

Purva Singh
School of Computer Science
Georgia Institute of Technology
Atlanta, USA

Joao Eduardo Ferreira
Institute of Mathematics and Statistics
University of Sao Paulo
Sao Paulo, Brazil

Calton Pu
School of Computer Science
Georgia Institute of Technology
Atlanta, USA

IEEE 4th International Conference on Cognitive Machine Intelligence (CogMI) | 978-1-6654-7406-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/CogMI

***Abstract*—Machine learning models with explainable predictions are increasingly sought after, especially for real-world, mission-critical applications that require bias detection and risk mitigation. Inherent interpretability, where a model is designed from the ground-up for interpretability, provides intuitive insights and transparent explanations on model prediction and performance. In this paper, we present CoLabel, an approach to build interpretable models with explanations rooted in the ground truth. We demonstrate CoLabel in a vehicle feature extraction application in the context of vehicle make-model recognition (VMMR). By construction, CoLabel performs VMMR with a composite of interpretable features such as vehicle color, type, and make, all based on interpretable annotations of the ground truth labels. First, CoLabel performs corroborative integration to join multiple datasets that each have a subset of desired annotations of color, type, and make. Then, CoLabel uses decomposable branches to extract complementary features corresponding to desired annotations. Finally, CoLabel fuses them together for final predictions. During feature fusion, CoLabel harmonizes complementary branches so that VMMR features are compatible with each other and can be projected to the same semantic space for classification. With inherent interpretability, CoLabel achieves superior performance to the state-of-the-art black-box models, with accuracy of 0.98, 0.95, and 0.94 on CompCars, Cars196, and BoxCars116K, respectively. CoLabel provides intuitive explanations due to constructive interpretability, and subsequently achieves high accuracy and usability in mission-critical situations.**

DOI 10.1109/CogMI56440.2022.00021

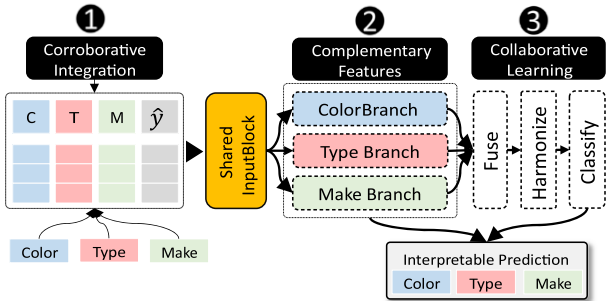


Fig. 1: CoLabel Architecture and Dataflow: CoLabel takes in multiple vehicle datasets, each with a subset of desired annotations. Corroborative Integration integrates annotations into a single training set. Complementary Features extracts interpretable features with complementary branches. Collaborative Learning fuses complementary features to yield interpretable predictions for vehicle classification.

I. Introduction

Machine learning models that are interpretable and explainable are increasingly sought after in a wide variety of industry applications [1], [2], [3], [4]. Explainable models augment the black-box of deep networks by providing insights into their feature extraction and prediction [5]. They are particularly useful in real-world situations where accountability, transparency, and provenance of information for mission-critical human decisions are crucial, such as security, monitoring, and medicine [1], [6]. Interpretable indicates model features designed from the get-go to be human-readable. Explainable indicates *post-hoc* analysis of models to determine feature importance in prediction.

Inherently interpretable models [7] are designed from the getgo to provide explainable results. This contrasts with *post-hoc* explainability, where a black box model is analyzed to obtain potential explanations for its decisions. Inherently interpretable models provide more transparent explanations [7], since these are directly dependent on model design. Such models also bypasses some limitations of *post-hoc* explainability, such as unjustified counterfactual explanations [8]. Inherently interpretable models have higher trust under adversarial conditions [9] since their predictions can be directly tied to training ground truth through model-generated explanations.

978-1-6654-7406-1/22/$31.00 ©2022 IEEE

There are several challenges, however, in building models that are inherently interpretable. There is no one-size-fits-all solution since interpretability is domain-specific [7]. Existing datasets may not have completely interpretable annotations; instead, most datasets only have the ground truth annotations without interpretable subsets. For example, existing vehicle classification datasets label vehicle make and model, but not vehicle color, type, or decals [10]. Furthermore, deep networks are biased during training towards strong signals, and may ignore more interpretable weaker signals. For example, person re-id models focus primarily on a person's shirt, and need guidance to focus on more interpretable accessories such as hats, handbags, or limbs [1].

CoLabel. In this paper, we present CoLabel: a process for building inherently interpretable models. We use CoLabel to build end-to-end interpretable models that provide explanations rooted in the ground truth. By construction, CoLabel provides predictions along with a composite of interpretable features that comprise the prediction with a combination of Corroborative Integration, Complementary Features, and Collaborative Learning. We call this this approach to achieve interpretable models from design and implementation *constructive interpretability*.

We demonstrate the inherent interpretability and superior accuracy of CoLabel in vehicle feature extraction, an important challenge in monitoring, tracking, and surveillance applications [9], [7]. Specifically, vehicle features are crucial for re-id, traffic monitoring and management, tracking, and make/model classification. Current state-of-the-art vehicle classification models employ black-box models.

These mission-critical applications require interpretable predictions for aiding human decisions, particularly for borderline cases where explanations aligned with human experience can benefit human decisions much more than algorithmic internal specifics. The goal of constructive interpretability is to design and build inherently interpretable models aligned with human understanding of applications. This is where CoLabel comes in. As an inherently interpretable model with constructive interpretability in mind, CoLabel has state-of-the-art accuracy as well as interpretable predictions.

We show CoLabel's dataflow with respect to vehicle feature extraction in Figure 1. Our constructive interpretability approach for inherently interpretable models begins from selection of interpretation annotations for vehicle features: color, type, and make. CoLabel uses these annotations to generate interpretable vehicle features. These features are usable in a variety of applications, such as vehicle make and model recognition (VMMR), re-id, tracking, and detection [10]. In this work, we focus CoLabel on VMMR.

Dataflow. Given our desired annotations of color, type, an make, as well as datasets that each carry a subset of these annotations, CoLabel's dataflow involves the following three steps:

Corroborative Integration integrates multiple datasets and corroborates annotations of 'natural' vehicle features across them. It then builds a robust training set with ground truth as well as interpretable annotations.

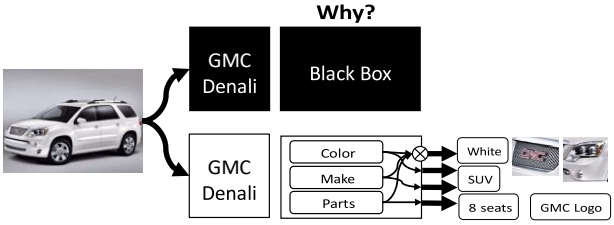


Fig. 2: Constructive Interpretability: Models with constructive interpretability provide explanations for their predictions. This is achieved by incorporating human knowledge. Here, an interpretable model explains its prediction is based on vehicle color, type, seats, and vehicle part similarities.

The Complementary Features module extracts features corresponding to the interpretable annotations. The goal is to maintain interpretable knowledge when integrated. Each complementary feature is extracted with its own branch in the CoLabel model.

Finally, Collaborative Learning harmonizes complementary features, ensuring features from different branches can be fused more effectively. With harmonization, branches collaborate to exploit correlations between complementary features.

Contributions. We show that CoLabel can achieve excellent accuracy on feature extraction while simultaneously providing interpretable results by construction. CoLabel's explanations align with human knowledge of vehicles. The contributions are:

- CoLabel [1]: Constructive interpretability approach to design and build an inherently interpretable vehicle feature extraction system by integrating diverse interpretable annotations that are aligned with human knowledge of vehicles.
- Model: Experimental evaluation and demonstration of the superior accuracy and interpretability achieved by CoLabel across common VMMR datasets: on CompCars, Cars196, and BoxCars116K , CoLabel achieves accuracies of 0.98, 0.95, and 0.94, respectively.

[1] 1We provide an anonymized repository at https://anonymous.4open.science/r/GLAMOR-024D/readme.md

II. Related Work

We first cover recent work in inherently interpretable models and *post-hoc* explanations. We will then cover vehicle feature extraction.

A. Interpretability and Explainability

Interpretability. Interpretable models are designed from the ground up to provide explanations for their features. Intuitively, interpretability is deeply intertwined with human understanding [8]. Models that are inherently interpretable directly integrate human understanding into feature generation.

Such models are more desired is mission-critical scenarios such as healthcare, monitoring, and safety management[7], [11], [12]. The prototype layers in [13], [11] provide interpretable predictions: the layers compare test image samples to similar ground truth samples to provide explanations of the model classification. The decomposable approaches in [14], [15] build component-classifiers that are integrated for the overall task, e.g. image and credit classification, respectively.

Explainability. Currently, most approaches perform *posthoc* explanation in a bottom-up fashion, where an existing model's black-box is 'opened' [16]. These include examining class activations [17], [18], concept activation vectors [4], neuron influence aggregation [19], and deconvolutions [5]. In *post-hoc* explanation, second model is used to model the original model's predictions by projecting model features along human-readable dimensions, if possible [18], [20], [19], [21]. However, these approaches do not build interpretable models from the ground truth; they merely enhance existing models for explanation. For example, Grad-CAM's [17] outputs are used with human labeling to determine 'where' and 'what' a model is looking at [18], [20]. Similarly, the embedding and neuron views in [19], [21] make it easier to visually characterize an class similarity clusters. However, there are risks to explainability when it is disconnected from the ground truth [7], [8], [9]. Such explanations may not be accurate, because if they were, the explainer model would be sufficient for prediction [8]. Furthermore, interpretable models are usually as accurate as black-box models, with the added benefit of interpretability, with several examples provided in [22]. Thus, the challenges of bottom-up approaches are bypassed with *constructive interpretability*, since it is a top-down approach for interpretability, as in Figure 2. In constructive interpretability, inherently interpretable models are built with features aligned with human knowledge of application domains, forming intuitively understandable and interpretable models as in [7], [11], [8], [13], [14], [15].

B. Vehicle Feature Extraction

We demonstrate CoLabel's interpretability with vehicle feature extraction. This encompasses several application areas, from VMMR[23], [24], [25], re-id [26], [27], [28], [29], tracking [30], [31], [32], [33], and vehicle detection[34], [35]. We cover recent research on feature extraction for these application areas.

CNN and SSD models are used together for logo detection in high resolution images[36]. Logo-Net [37] uses such a composition to improve logo detection and classification. Wang et. al. [38] develop an orientation-invariant approach that uses 20 engineered keypoint regions on vehicles to extract representative features. Liu et. al. propose RAM [39], which has sub-models that each focus on a different region of the vehicle's image, because different regions of vehicles have different relevant features.

Additionally, there have been recent datasets with varied annotations for feature extraction. Boukerche and Ma [10] provide a survey of such datasets for feature extraction. Yang et. al. [40] propose a part attributes driven vehicle model recognition and develop the CompCars dataset with VMMR labels. BoxCars116K [34] provides a dataset of vehicles annotations with type, and uses conventional vision modules for vehicle bounding box detection.

Summary. Since each application area for vehicle features remains sensitive and mission-critical, interpretable features are crucial for deployment. The above approaches have improved on feature extraction. We augment them with CoLabel to demonstrate interpretability. We will further show that such inherently interpretable models offer additional avenues for increasing model accuracy.

III. CoLabel

CoLabel, our approach for interpretable feature extractions (Figure 3). We have given an overview of CoLabel's dataflow in Figure 1. Here, we describe CoLabel's components in details. We first describe Corroborative Integration in §III-A. Then we present Complementary Features for interpretable annotations in §III-B. Finally, we present Collaborative Learning in §III-C, where CoLabel fuses complementary features for interpretable predictions. We implement CoLabel for vehicle feature extraction, which has a need for interpretability in a variety of mission-critical applications in traffic management, safety monitoring, and multi-camera tracking. Specifically, we apply CoLabel for interpretable vehicle make and model recognition (VMMR), where the task is to generate features from vehicle images to identify the make and model.

A. Corroborative Integration

Constructive interpretability starts from a judicious decomposition of the application domain into interpretable annotations. For vehicle classification with CoLabel, we identified three interpretable annotations for model training in addition to VMMR labels: vehicle color, type, and make. Vehicle color is the overall color scheme of a vehicle. Type is the body type of the vehicle, such as SUV, sedan, or pickup truck. Finally, make is the vehicle brand, such as Toyota, Mazda, or Jeep. We select these annotations as they are broadly common across vehicle feature extraction datasets [10].

One advantage of CoLabel is a very inclusive classifier training process. Of the several VMMR datasets (discussed in §IV-A), each has a subset of the three desired interpretable annotations. CompCars [40] labels make, model and type. VehicleColors [41] labels only the colors. BoxCars116K [34] labels make and model. CoLabel is capable of integrating the partial knowledge in all of labeled data sets with Corroborative Integration.

Labeling Overview. So, CoLabel uses Corroborative Integration to 'complete' partial annotations: in this case by adding color labels to each image in CompCars. Let there be k interpretable annotations, and a set D of datasets, all of which contain some subset of k annotations/labels $\hat{y}_k$. Datasets in D also contain the overall ground truth label $\hat{y}$, *i.e.* VMMR. For CoLabel, $k = 3$: color ($\hat{y}_{color}$), type ($\hat{y}_{type}$) and make ($\hat{y}_{make}$). We show Corroborative Integration for a single interpretable annotation ($\hat{y}_{color}$) in Figure 4.

Fig. 4: Corroborative Integration: Given a desired annotation k (e.g. color), a subset $d^s_{color} \in D$ of VMMR datasets contains this annotation. Corroborative Integration employs a team of labeling models M_{color^s}, each trained on a corresponding d^s_{color}. Each model employs JPEG compression ensemble to improve labeling agreement. Using Corroborative Integration, CoLabel can label the remaining $D-s$ datasets with the color annotations.

So, a subset $\{d_{color}\}_s \in D$ contains desired annotation $\hat{y}_{color}$. We train a set of models M_{colors}, one for each of the s datasets in $\{d_{color}\}_s$, with the datasets' corresponding $\{\hat{y}_{color}\}_s$ as the ground truth. Then, we build an team of $\{M_{color}\}_s$ to label the remaining datasets in D without color annotation, *e.g.* the $\hat{y}_{color}$-unlabeled subset $\{d^*_{color}\}_{|D|-s}$, with weighted voting. Since d^*_{color} has a subset of desired annotations, we call it a partially unlabeled dataset; this subset is the complement of labeled subset d_{color}.

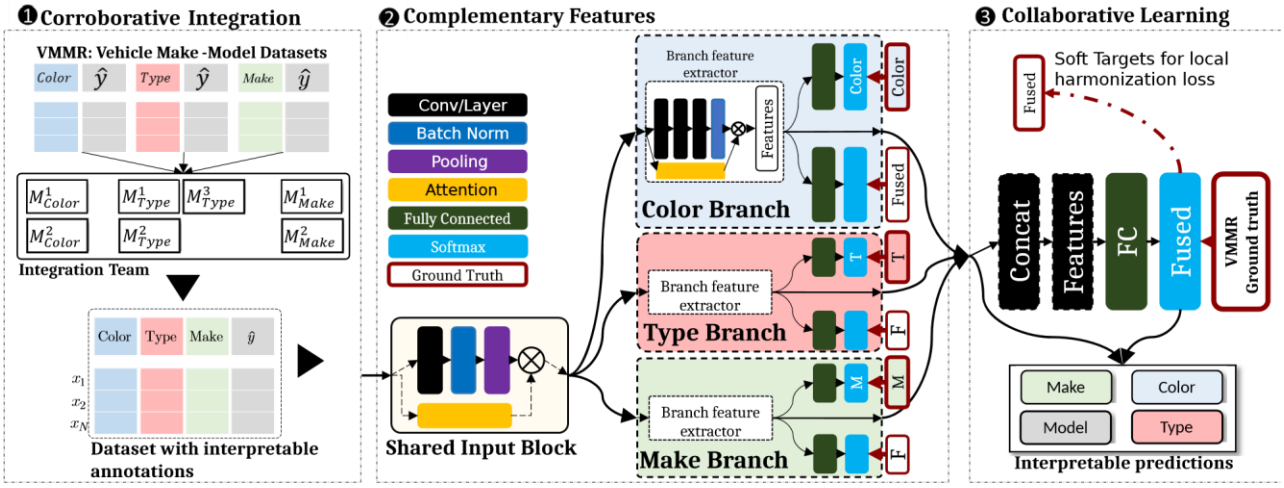
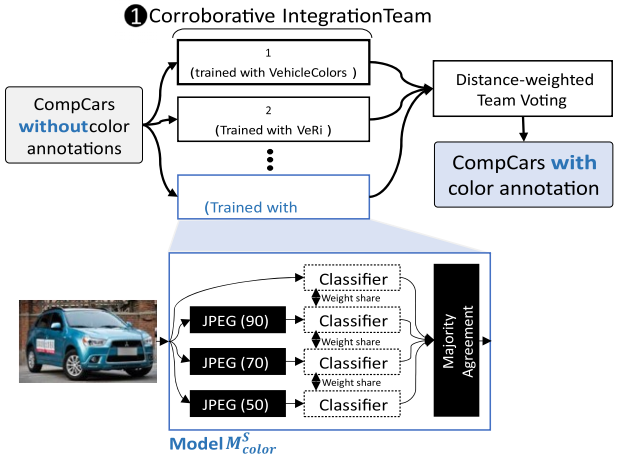


Fig. 3: CoLabel for VMMR: For VMMR, we use color, type, and make labels as our interpretable annotations. Corroborative Integration combines various VMMR datasets, each with a subset of desired interpretable annotations, with a labeling team to generate a single dataset with all desired annotations (§III-A). Then, Complementary Features uses 3 branches to extract color, type, and make features. Each branch contains a feature extractor backbone. A dense layer converts features to branch-specific predictions. A second dense layer yields harmonization features (§III-B). Finally, Collaborative Learning fuses features for VMMR classification. Simultaneously, a harmonization loss ensures branch features collaborate on feature correlations (§III-C). Predictions are combined from Collaborative Learning and Complementary Features to generate interpretable classification that correspond to annotations from Corroborative Integration.



Labeling Team. Team member votes are dynamically weighted for each partially unlabeled dataset d^*_{color}. To obtain these weights, we first apply tSNE to the union $\{d_{color}\}_s \cup \{d^*_{color}\}_i$, where i is the current partially unlabeled dataset. After dimensionality reduction, we obtain the distance between each d_{color} cluster center to the $\{d^*_{color}\}_i$ cluster center.

The closest labeled datasets get proportionally higher weights, similar to the approach in [42].

Each trained model generalizes to the task for cross-domain datasets, as we will show in §IV-B. However, models still encounter edge cases due to dataset overfitting [43]. We address cross-domain performance deterioration with early stopping, a JPEG compression ensemble, and greater-thanmajority agreement among models.

Early Stopping. During training of each model in $\{M_{color}\}_s$, we compute validation accuracy over the validation sets in $\{d_{color}\}_s$. With early stopping tuned to cross-dataset validation accuracy, we can ensure models do not overfit to their own training dataset.

JPEG Compression. During labeling of any d^*_{color}, each labeling model M_{color} takes 4 copies of an unlabeled image. The first is the original image. Each of the three remaining copies is the original image compressed using the JPEG protocol, with quality factors of 90, 70, and 50. We use the majority predicted label amongst the four copies as the model's final prediction. This is similar to the 'vaccination' step from [44], where JPEG compression removes high-frequency artifacts that can impact cross-dataset performance.

Team Agreement. If an image has no majority label from a model's JPEG ensemble, we discard that model's label. Further, if we only have < 50% of models in the labeling team after discarding models without JPEG ensemble majority label, we leave the annotation for that image blank.

Summary. Using these steps, we can label most partially unlabeled samples datasets without the desired annotation. We evaluate labeling accuracy given these strategies in §IV-B, where we test on held-out labeled datasets. Our results show average accuracy improvement of almost 20% from 0.83 to 0.98 for held-out unlabeled samples when we use labeling teams with early stopping for team members and JPEG compression ensemble for each model.

B. Complementary Features

CoLabel's next step is Complementary Features extraction, which propagates the three interpretable annotations from Corroborative Integration for feature extraction. Complementary Features comprises of 2 stages: a shared input stage, followed by $k = 3$ complementary feature branches. The shared input stage performs shallow convolutional feature extraction. These features are common to each branch's input. Then, the complementary branches extract their annotationspecific interpretable features and propagate them to Collaborative Learning.

Shared Input Block. Multi-branch models often use different inputs for each branch [45], [46]. CoLabel uses a shared input block to create common input to each branch to improve feature fusing in Collaborative Learning. Feature fusing requires integration of branches that carry different semantics and scales. This is accomplished with additional dense layers after concatenation [38], longer training to ensure convergence or appropriate selection of loss functions [45]. CoLabel's branches perform complementary feature extraction, where the features are semantically different. So, we use a shared input block to ensure branches have a common starting point in shallow convolutional features and use attribute features from those layers [1], [47]. It consists of early layers in a feature extractor backbone such as ResNet, along with attention modules (we use CBAM [48]). For CoLabel, we use the first ResNet bottleneck block in the shared input layer, and use the remaining bottleneck blocks in the branches. We show the the impact of the shared input block for training in §IV-C.

Complementary Feature Branches. Each interpretable annotation from Corroborative Integration is matched to a corresponding branch in CoLabel. We describe a single Complementary Features branch here. The features x_{shared} from the shared input block are passed through a feature extractor backbone comprising of conv layers, normalization, pooling, and CBAM. This yields intermediate features x_k, *e.g.* x_{color} for the color branch. A fully connected layer F_{color} projects x_{color} to predictions y_{color}. A second fully connected layer $F_{color-fused}$ projects x_{color} to tentative fused predictions $y_{color-fused}$ These tentative fused predictions are only used during training to improve feature harmonization in Collaborative Learning. We defer their discussion to §III-C. Finally, x_{color} is also sent to Collaborative Learning for fusion with the other branch features x_{type} and x_{make}.

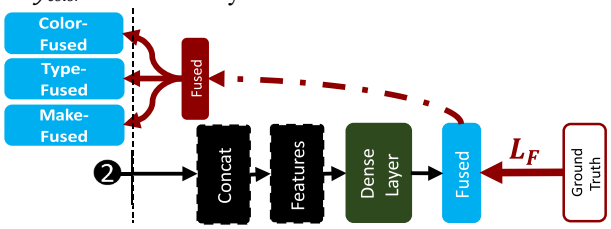


Fig. 5: Collaborative Learning: Features from Complementary Features are fused with concatenation. Branches collaborate on feature correlations and interdependencies with the harmonization losses L^k_H. L^k_H uses the fused features as soft-targets for $x_{k-fused}$ from §III-B. A fused loss L_F on the cross-entropy loss between VMMR targets and fused prediction also backpropagates over the entire model.

Training. During training, we compute 2 local losses to train the branch feature extractors. L^k_B is the branch specific loss for branch k, computed as the cross-entropy loss between y_k and $\hat{y}_k$:

$$L_{colorB} = \mathrm{H}(y_{color}, \hat{y}_{color}) \quad (1)$$

L^k_H, the local harmonization loss, is discussed in §III-C.

Impact of Complementary Features. CORROBORATIVE INTEGRATION generates interpretable features in x_{color}. As such, for any prediction, we can decompose COLABEL into the k complementary branches and explain predictions with the component annotations. For prediction errors, COLABEL provides provenance of its classification, so that the specific branch that caused the error can be updated. Further, the interpretable branches are extensible: any new desired annotations need only be added to the training set with CORROBORATIVE INTEGRATION. Subsequently, COMPLEMENTARY FEATURES will deploy a branch to generate interpretable features for the corresponding annotations. We discuss the impact of COMPLEMENTARY FEATURES in accuracy and interpretability in §IV-C

C. Collaborative Learning

Finally, COLABEL performs feature fusion to generate interpretable predictions. The branches of COMPLEMENTARY FEATURES yield their corresponding features x_{color}, x_{type}, and x_{make}. These are concatenated to yield fused features x_F. A fully connected layer projects x_F to final predictions y_F. COLABEL is trained end-to-end with the cross-entropy loss between predictions and ground truth:

$$L_F = \mathcal{H}(y_F, \hat{y}) = -\frac{1}{N_B}\sum^{N_B} p_i^{y_F} \log(p_i^{\hat{y}}) \quad (2)$$

Local Harmonization Loss. COLABEL employs a local harmonization loss for each branch in COMPLEMENTARY FEATURES. Since the branches are extracting complementary features, we need a way for branches to exploit correlation and interdependency between features. We accomplish this with weak supervision on the branch features using the fused feature predictions y_F. Intuitively, we want branches to agree on the overall VMMR task. So, branch features should also accomplish VMMR, in addition to their branch-specific annotation. Using this fused-feature knowledge distillation, we ensure that branch features harmonize on the final VMMR prediction labels y_F. In effect, y_F is a soft target for each branch. The local harmonization loss is computed for each branch as the cross-entropy loss between the tentative fused predictions $y_{k-fused}$ from COMPLEMENTARY FEATURES and the final fused predictions y_F. For the color branch:

$$L_{colorH} = \mathrm{H}(y_{color-fused}, y_F) \quad (3)$$

Losses. COLABEL employs 3 losses during training, shown as red arrows in Figure 3 and Figure 5. The fused loss L_F in Eq. (2) backpropagates through the entire model. COLABEL's branches are trained with branch annotation loss L_B^k and local harmonization loss $L^k{}_H$:

$$L_k = L_B^k + L_H^k = \mathcal{H}(y_k, \hat{y}_k) + \mathcal{H}(y_{k-fused}, y_F)$$
$$= -\frac{1}{N_C}\sum^{N_C} p_i^{y_k} \log p_i^{\hat{y}_k} - \frac{1}{N_B}\sum^{N_B} p_i^{y_{kfused}} \log p_i^{y_F} \quad (4)$$

Here, C is the subset of mini-batch B that has the annotations for x_k. We need this because during CORROBORATIVE INTEGRATION, COLABEL leaves unlabeled samples without team agreement as unlabeled. These unlabeled samples can be processed under an active learning scheme. For this paper, we compute loss using the subset of samples for which the annotation is known.

IV. RESULTS

Now, we show the effectiveness and interpretability of COLABEL. First, we will cover the experimental setup. Then we will evaluate each of COLABEL's components and demonstrate efficacy of design choices. Finally, we demonstrate interpretability as well as high accuracy with the end-to-end model for VMMR.

A. Experimental Setup

We cover system setup, as well as datasets.

System Details. We implemented COLABEL in PyTorch 1.4 on Python 3.8. For our backbones, we use ResNet with IBN [49], with pretrained ImageNet weights. Experiment are performed on a server with NVIDIA Tesla P100, and an Intel Xeon 2GHz processor.

Datasets. We use the following datasets: CompCars[40], BoxCars116K[34], Cars196[50], VehicleColors[41], and VeRi[51]. We also obtained our own dataset of vehicles labeled with color and type annotations using a web crawler on a variety of car-sale sites, called CrawledVehicles. Datasets are described in Table I.

TABLE I: Datasets: We use the boldfaced datasets in our final evaluations. They are partially annotated. To complete the *No* annotations, we use the underlined datasets for each annotation in CORROBORATIVE INTEGRATION.

Dataset	Make	Model	Color	Type
CompCars[40]	Yes	Yes	*No*	Yes
BoxCars116K[34]	Yes	Yes	*No*	*No*
Cars196[50]	Yes	Yes	*No*	*No*
VehicleColors[41]	No	No	Yes	No
VeRi[51]	No	No	Yes	Yes
CrawledVehicles (ours)	No	No	Yes	Yes

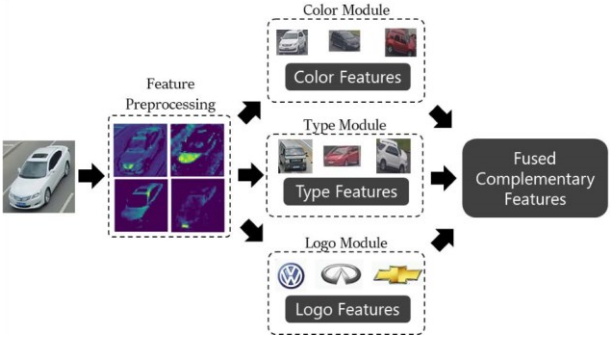


Fig. 6: COLABEL Dataflow: Feature preprocessing extracts coarse feature maps for each branch. Branches extract branch specific features. These complementary features are fused for vehicle classification. In addition, branch specific features provide important metadata for the images.

We use CompCars, BoxCars116K, and Cars196 for end-toend evaluations. Their annotations are incomplete, since none contain all three desired annotations. We complete the ground truth for these datasets with CORROBORATIVE INTEGRATION.

Dataflow. We show CoLABEL's implementation dataflow in Figure 6. We obtain attention maps from the initial attention modules, described in §III-B. Attention maps are provided to each branch, which select best-fit attention map for their corresponding feature extractors. For example, color module clusters vehicles based on color, while the type module clusters on vehicle type according to labels from Table I. Finally, we fuse features to obtain predictions.

B. Corroborative Integration

CompCars, Cars196, and BoxCars116K are missing annotations from our desired interpretable annotation list of make, color, and type (see Table I). We use CORROBORATIVE INTEGRATION to augment these datasets. Specifically, we use VehicleColors, CrawledVehicles, and VeRi to label Cars196, BoxCars116K, and CompCars with color annotations. Then, we use CompCars and CrawledVehicles to label Cars196 and BoxCars116K with type annotations.

Color Model (Color-CM). Color-CM is a team of 3 models, where each model is trained with VehicleColors, CrawledVehicles, or VeRi, respectively. During training of each model, we use horizontal flipping, random erasing, and cropping

TABLE II: Color-CM: Accuracy of Color-CM teams in labeling heldout datasets with color annotations. For each column's evaluation, the team member models are trained with the datasets of the other 2 columns.

	VehicleColors	VeRi	CrawledVehicles
Initial	0.87	0.84	0.86
+Early Stop	0.89	0.9	0.92
+Compression (90)	0.91	0.91	0.92
+Compression(90, 70, 50)	0.94	0.93	0.94
+Labeling Team	0.95	0.95	0.96
+Agreement	0.98	0.97	0.98

TABLE III: Type-CM: Accuracy of Type-CM team in held-out dataset with type annotations.

	CompCars	CrawledVehicles	VeRi
Initial	0.86	0.88	0.86
+Early Stop	0.89	0.91	0.89
+Compression (90)	0.91	0.91	0.91
+Compression(90, 70, 50)	0.93	0.94	0.94
+Labeling Team	0.96	0.95	0.95
+Agreement	0.98	0.97	0.98

augmentations to improve training accuracy. For the JPEG compression ratios of each model, we test with 2 schemes: (a) with a single additional ratio of quality factor 90, and (b) 3 additional ratios with quality factors 90, 70, and 50. We use majority voting from the JPEG compression ensemble. Then with majority weighted voting from team member models, we arrive at the final prediction.

We evaluate with held-out test sets from the labeled subset of datasets. Specifically, we conduct three evaluations. For each of the 3 labeled datasets VehicleColors, CrawledVehicles, and VeRi, we use one for testing and the remaining 2 for building the team. Results are provided in Table II.

On each held-out dataset, initial accuracy is ~0.86. With cross-validation early stopping, we can increase this to ~0.91 With JPEG compression with 3 ratios, we can increase accuracy by a further 3%. By teaming several models, we further increase accuracy to ~0.95 Finally, we add the agreement constraint, where we accept a label only if > 50% of models have agreed on the label. This improves labeling accuracy by an additional 3%, to 0.98. On the held-out test set, we can thus label 90% of the samples, with the other 10% remaining unlabeled due to disagreements.

With these strategies, we label color for BoxCars116K and Cars196 using CORROBORATIVE INTEGRATION. CoLABEL can label 88% of the images in these datasets. Figure 7 shows a sample of these images and their assigned labels in Cars196.

Type Corroborative Model (Type-CM). The team for type annotation labeling is trained with ground truth in CompCars, CrawledVehicles, and VeRi to label BoxCars116K and Cars196. The training process is similar to Color-CM.

Table III shows held-out accuracy on test sets. The initial accuracy is 0.87, and with early stopping and JPEG compression, we can increase accuracy by 4%, to 0.94. Team of models

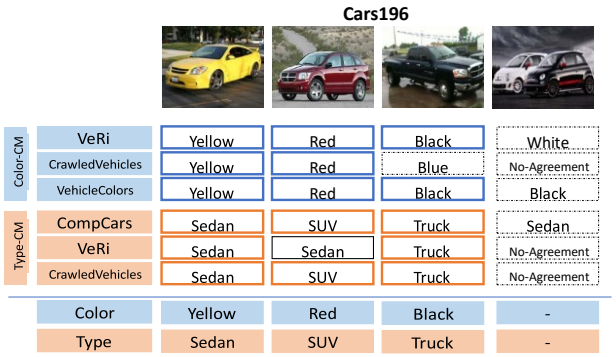


Fig. 7: Labeled Cars196 Images: Using CORROBORATIVE INTEGRATION, we can assign color and type annotations to Cars196 images. For some images with multiple vehicles or occlusions, we leave blank annotations if CORROBORATIVE INTEGRATION cannot find agreement on the labels.

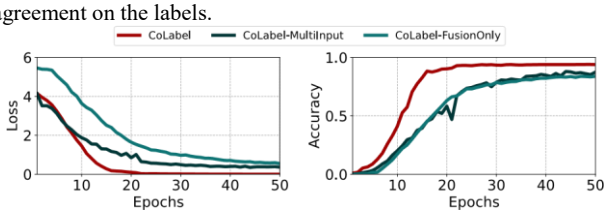


Fig. 8: Convergence of CoLABEL: We compare convergence with respect to loss minimization and accuracy of CoLABEL against CoLABEL-MULTIINPUT and CoLABEL-FUSIONONLY on CompCars.

CoLabel-MultiInput has multiple inputs, which slows learning of fuseable branch features. CoLabel-FusionOnly uses only the fused feature loss without local harmonization, which reduces effectiveness of feature fusion.

increases accuracy to 0.96. By adding JPEG compression agreement to team members, we arrive at a final accuracy of 0.98. On our desired ground truth of BoxCars116K and Cars196, the team of Type-CM models corroboratively labels 91% of samples.

C. Complementary Features

After Corroborative Integration, we have our desired interpretable annotations in the BoxCars116K, CompCars, and Cars196 datasets. Here, we demonstrate interpretable classification with Complementary Features. Shared Input Block. We first cover the shared input block. The shared block is important for feature harmonization to ensure complementary features are extracted from a common set of convolutional features to improve semantic agreement. Further, the shared block is updated with backpropagated loss from all branches, ensuring the shallow features it extracts are usable by all branches. In turn, this improves convergence and training time for CoLabel. We show in Figure 8 the impact of the shared input block in convergence by comparing loss over time between CoLabel and CoLabel-MultiInput, a model with an input for each branch. The shared input block improves weight convergence and training time; we will discuss
CoLabel-FusionOnly in the figure in §IV-D

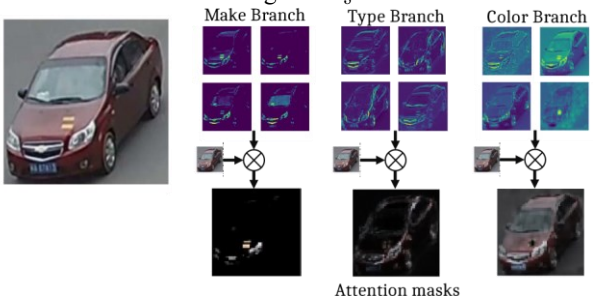


Fig. 9: Attention masks: We obtain attention masks from the attention modules in Corroborative Integration. We show a few of the attention masks in shallow layers of each branch that help the branch extract their annotation-specific features.

TABLE IV: Impact of attention: CoLabel with attention outperforms model without attention (NoAtt) across all classification tasks. For VMMR, attention yields between 4-5% improvement across all datasets.

	CompCars		Cars196		BoxCars116K	
	CoLabel	NoAtt	CoLabel	NoAtt	CoLabel	NoAtt
Classification	0.96	0.89	0.94	0.91	0.89	0.84
Color	0.95	0.94	0.96	0.91	0.93	0.89
Type	0.96	0.92	0.96	0.92	0.91	0.85
Make	0.95	0.91	0.92	0.87	0.87	0.81

Attention Modules. Complementary Features also uses attention modules in both the shared input block and the complementary feature branches. Attention improves feature extraction in both cases. For the shared input layer, attention masks identify image region containing relevant features for each branch. For complementary features, attention further improves feature extraction accuracy, shown in Table IV. The make branch improves classification accuracy from 0.91 without attention (NoAtt) to 0.95 by including attention in the branch.

Attention Masks. We show attention masks of each branch in Figure 9. While attention masks are generally black-boxes, CoLabel's interpretability allows us to make educated guesses about the masks. Masks for each branch are visually similar to each other, indicating attention has been clustered by
Corroborative Integration. Further, the make branch masks indicate the branch focuses on the logo area of vehicles. Similarly, the type branch masks focus on the general shape of the vehicle at the edges. The color branch masks extract overall vehicle color information.

D. Collaborative Learning

Finally, CoLabel fuses complementary features to generate final features for VMMR classification. We evaluate CoLabel end-to-end to demonstrate the feasibility of inherently interpretable models with several experiments.

Impact of Loss Functions. First, we show the impact of loss functions on training convergence and accuracy on the CompCars dataset. Figure 8 compares CoLabel to a CoLabel-FusionOnly, which uses only the final fused loss and without the local harmonization losses. We also show accuracy across datasets in Table V. By adding the local harmonization loss to improve feature fusion, we can increase accuracy by almost 10% on average; on CompCars, we increase accuracy from 0.87 to 0.96 for VMMR. Without the harmonization loss, CoLabel-FusionOnly converges slower and has lower accuracy.

TABLE V: Impact of Harmonization Loss: Across all datasets, using both local harmonization loss and final fused loss significantly improves accuracy.

	CompCars	Cars196	BoxCars116K
CoLabel	0.96	0.94	0.89
FusionOnly	0.87	0.84	0.81

TABLE VI: CoLabel vs Non-Interpretable Models: We show performance of CoLabel against several states-of-the-art. CoLabel achieves similar performance with the added benefit of interpretability. Further, interpretability allows us to exploit disagreements between classifications and existing knowledgebases to further improve accuracy. We show this with CoLabel-Match, a model that selfdiagnoses mistakes using existing knowledge about vehicle makes, models, and types.

	CompCars	BoxCars116K	Cars196
R50-Att [49]	0.90	0.75	0.89
R152 [24]	0.95	0.87	0.92
D161-CMP [24]	0.97	-	0.92
R50-CL [23]	0.95	0.86	-
CoLabel	0.96	0.89	0.94
CoLabel-Match	0.97	0.93	0.94

Interpretability and Accuracy. Now, we evaluate CoLabel against several non-interpretable models: (a) R50-Att, a ResNet50 backbone with a single branch with IBN and attention [49], (b) R152, a ResNet152 with benchmark results from [24], (c) D161-CMP, a DenseNet with channel pooling from [24], and (d) R50-CL, a ResNet50 with unsupervised co-occurrence learning [23].

For CoLabel, we use a ResNet34 backbone. The first bottleneck block resides in the shared input block. The remaining three bottleneck blocks are copied to each branch, as described in §III-B. For each model, we use image size 224×224, and train for 50 epochs with lr=1e-4, with a batch size of 64.

Results are shown in Table VI. CoLabel achieves slightly higher accuracy than both R152 and R50-Att, with accuracy of 0.96 on VMMR on CompCars. For Cars196 and BoxCars116K, CoLabel achieves accuracy of 0.94 and 0.89, respectively. In each case, CoLabel achieves similar or slightly better performance than existing non-interpretable approaches.

However, CoLabel's results are also interpretable, allowing us to further increase accuracy by *retroactive corrections*. Given vehicle models and their ground truth types from existing vehicle databases [52], we can check where CoLabel's type detection and vehicle model predictions do not agree. This occurs when the image is difficult to process, either due to occlusion, blurriness, or other artifacts (an example such disagreement in Corroborative Integration with 2 cars in the same image is shown in Figure 7). As such, CoLabel generates conflicting interpretations, which are themselves

TABLE VII: Single-Branch Multi-Labeling: With multi-labeling output from a single branch in CoLabel-SMBL, we can maintain interpretability for a single-branch while sacrificing accuracy.

	CompCars	BoxCars116K	Cars196	Params
HML [53]	0.65	-	-	-
CoLabel-SMBL	0.91	0.84	0.89	25M
CoLabel	0.96	0.89	0.94	60M

TABLE VIII: All-v-All vs 2-Stage Cascade: We compare CoLabel under all-v-all and 2-stage cascade. With CoLabel-2SC, we use a specialized submodel for each make, simplifying the VMMR problem. We can benefit from interpretability by including retroactive correction to further increase VMMR classification accuracy.

	CompCars	BoxCars116K	Cars196
D161-SMP	0.97	-	0.92
CoLabel (AVA)	0.96	0.89	0.94
CoLabel-Match	0.97	0.93	0.94
CoLabel-2SC	0.97	0.93	0.95
CoLabel-2SC-Match	0.98	0.94	0.95

useful in analyzing the model. Using this variation called CoLabel-Match, we can further increase accuracy solely due to interpretability, to 0.97, 0.95, and 0.93 on CompCars, Cars196, and BoxCars116K, respectively.

Single-Branch Multi-Labeling. Since CoLabel uses multiple branches, a natural question is: could branches be removed while maintaining interpretability? We compare CoLabel's multi-branch interpretability with Single-Branch Multi-Labeling approach, called CoLabel-SMBL. In CoLabelSMBL, we use a single branch for feature extraction. The features are then used in 4 parallel dense layers: color, type, make, and VMMR detection. With CoLabel-SMBL, we could reduce model parameters, since we use a single branch. We compare CoLabel-SMBL against CoLabel and HML [53] in Table VII. CoLabel-SMBL sacrifices accuracy with the reduced parameters. Further, we also found CoLabelSMBL more difficult to converge, as it needed fine-tuning of learning rates to contend with the multiple backpropagated losses. We leave further exploration of CoLabel-SMBL with other architecture choices to future work.

All-v-All vs 2-Stage Cascade. Here, we evaluate CoLabel as a 2-stage cascade (CoLabel-2SC) and compare to all-v-all (AVA) in Table VIII. AVA is the method we have described in CoLabel, where final features are used for make and model classification. In essence, this is a complex problem where CoLabel's fused features are trained with every vehicle model in our datasets. In CoLabel-2SC, we simplify the problem by creating classifier submodel (*i.e.* a dense layer) for each make. The submodels use CoLabel' fused features for prediction. So, given the 78 makes in CompCars, we create 78 submodels. For each image, CoLabel-2SC's vehicle make prediction activates the corresponding classification submodel.

Since CoLabel-2SC works on a simpler problem, we can improve accuracy from CoLabel, with a trade-off of increased parameters due to the submodels. CoLabel-2SC achieves accuracy of 0.97 on CompCars, 0.95 on Cars196, and
0.93 on BoxCars116K, comparable to D161-CMP [24]. With CoLabel-2SC-Match, we apply *retroactive correction* to further improve accuracy on CompCars to 0.98 by verifying predictions with make-model-type knowledgebase [52].

V. Conclusion

In this paper, we have presented constructive interpretability with CoLabel, an inherently interpretable model for feature extraction and classification. CoLabel contains 3 components for interpretability. Corroborative Integration allows us to complete interpretable annotations in datasets using a variety of corroborative datasets. Complementary Features perform feature extraction corresponding to interpretable annotations. Finally, Collaborative Learning lets CoLabel fuse features effectively during training using local harmonization losses for each branch. Per [7], there is no tradeoff between interpretability and accuracy. Our evaluations show that CoLabel has superior accuracy to state-of-the-art black box models. We are also able to exploit interpretability to selfdiagnose mistakes in classification, further increasing accuracy with CoLabel-Match and CoLabel-2SC-Match.

Acknowledgement

This research has been partially funded by National Science

Foundation by CISE/CNS (1550379, 2026945, 2039653), SaTC (1564097), SBE/HNDS (2024320) programs, and gifts, grants, or contracts from Fujitsu, HP, and Georgia Tech Foundation through the John P. Imlay, Jr. Chair endowment. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or other funding agencies and companies mentioned above.

REFERENCES

[1] X. Chen, X. Liu, W. Liu, X.-P. Zhang, Y. Zhang, and T. Mei, "Explainable person re-identification with attribute-guided metric distillation," in *ICCV*, 2021, pp. 11813–11822.

[2] Y. Zhao, S. Luo, Y. Yang, and M. Song, "Deepssh: Deep semantic structured hashing for explainable person re-identification," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 1653–1657.

[3] M.-L. Nguyen, T. Phung, D.-H. Ly, and H.-L. Truong, "Holistic explainability requirements for end-to-end machine learning in iot cloud systems," in *2021 IEEE REW Workshop*, 2021, pp. 188–194.

[4] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, "Explainable ai: A review of machine learning interpretability methods," *Entropy*, vol. 23, no. 1, p. 18, 2021.

[5] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV*. Springer, 2014, pp. 818–833.

[6] L. Zheng, Y. Yang, and A. G. Hauptmann, "Person re-identification: Past, present and future," *CoRR*, 2016.

[7] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019.

[8] T. Laugel, M.-J. Lesot, C. Marsala, X. Renard, and M. Detyniecki, "The dangers of post-hoc interpretability: Unjustified counterfactual explanations," in *IJCAI*, 2019, pp. 2801–2807.

[9] Z. C. Lipton, "The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery." *Queue*, vol. 16, no. 3, pp. 31–57, 2018.

[10] A. Boukerche and X. Ma, "Vision-based autonomous vehicle recognition: A new challenge for deep learning-based systems," *ACM Computing Surveys (CSUR)*, vol. 54, no. 4, pp. 1–37, 2021.

[11] C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, and J. K. Su, "This looks like that: deep learning for interpretable image recognition," *NeurIPS*, vol. 32, 2019.

[12] Z. Obermeyer, B. Powers, C. Vogeli, and S. Mullainathan, "Dissecting racial bias in an algorithm used to manage the health of populations," *Science*, vol. 366, no. 6464, pp. 447–453, 2019.

[13] O. Li, H. Liu, C. Chen, and C. Rudin, "Deep learning for casebased reasoning through prototypes: A neural network that explains its predictions," in *AAAI*, vol. 32, no. 1, 2018.

[14] S. Saralajew, L. Holdijk, M. Rees, E. Asan, and T. Villmann, "Classification-by-components: Probabilistic modeling of reasoning over a set of components," in *NeurIPS*, vol. 32, 2019.

[15] C. Chen, K. Lin, C. Rudin, Y. Shaposhnik, S. Wang, and T. Wang, "An interpretable model with globally consistent explanations for credit risk," *CoRR*, 2018.

[16] M. Hardt, X. Chen, X. Cheng, M. Donini, J. Gelman, S. Gollaprolu, J. He, P. Larroy, X. Liu, N. McCarthy *et al.*, "Amazon sagemaker clarify: Machine learning bias detection and explainability in the cloud," *CoRR*, 2021.

[17] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *ICCV*, 2017, pp. 618–626.

[18] X. Zhang, R. Zhang, J. Cao, D. Gong, M. You, and C. Shen, "Part-guided attention learning for vehicle instance retrieval," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[19] F. Hohman, H. Park, C. Robinson, and D. H. P. Chau, "Summit: Scaling deep learning interpretability by visualizing activation and attribution summarizations," *IEEE TVCG*, vol. 26, no. 1, pp. 1096–1106, 2019.

[20] L. Chen, J. Chen, H. Hajimirsadeghi, and G. Mori, "Adapting grad-cam for embedding networks," in *WACV*, 2020, pp. 2794–2803.

[21] N. Das, H. Park, Z. J. Wang, F. Hohman, R. Firstman, E. Rogers, and D. H. P. Chau, "Bluff: Interactively deciphering adversarial attacks on deep neural networks," in *IEEE VIS*. IEEE, 2020, pp. 271–275.

[22] C. Rudin and J. Radin, "Why are we using black box models in ai when we don't need to? a lesson from an explainable ai competition," *Harvard Data Science Review*, vol. 1, no. 2, 11 2019, https://hdsr.mitpress.mit.edu/pub/f9kuryi8. [Online]. Available: https://hdsr.mitpress.mit.edu/pub/f9kuryi8

[23] S. Elkerdawy, N. Ray, and H. Zhang, "Fine-grained vehicle classification with unsupervised parts co-occurrence learning," in *ECCV Workshops*, 2018.

[24] Z. Ma, D. Chang, J. Xie, Y. Ding, S. Wen, X. Li, Z. Si, and J. Guo, "Fine-grained vehicle classification with channel max pooling modified cnns," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3224–3233, 2019.

[25] H. C. Sanchez, N. H. Parra, I. P. Alonso, E. Nebot, and D. Fern´ andez-´ Llorca, "Are we ready for accurate and unbiased fine-grained vehicle classification in realistic environments?" *IEEE Access*, vol. 9, pp. 116338– 116355, 2021.

[26] B. He, J. Li, Y. Zhao, and Y. Tian, "Part-regularized near-duplicate vehicle re-identification," in *CVPR*, 2019, Conference Proceedings, pp. 3997–4005.

[27] H.-M. Hsu, T.-W. Huang, G. Wang, J. Cai, Z. Lei, and J.-N. Hwang, "Multi-camera tracking of vehicles based on deep features re-id and trajectory-based camera link models," in *CVPR*, 2019, Conference Proceedings.

[28] X. Liu, W. Liu, T. Mei, and H. Ma, "A deep learning-based approach to progressive vehicle re-identification for urban surveillance," in *ECCV*, 2016, Conference Proceedings.

[29] A. Suprem and C. Pu, "Looking glamorous: Vehicle re-id in heterogeneous cameras networks with global and local attention," *CoRR*, 2020.

[30] E. Ristani and C. Tomasi, "Features for multi-target multi-camera tracking and re-identification," in *CVPR*, 2018, Conference Proceedings, pp. 6036– 6046.

[31] Z. Tang, G. Wang, H. Xiao, A. Zheng, and J.-N. Hwang, "Single-camera and inter-camera vehicle tracking and 3d speed estimation based on fusion of visual and semantic features," in *CVPR Workshops*, 2018, Conference Proceedings, pp. 108–115.

[32] Z. Xu, H. Gupta, and U. Ramachandran, "Sttr: A system for tracking all vehicles all the time at the edge of the network," in *DEBS*.ACM, 2018, Conference Proceedings, pp. 124–135.

[33] A. Suprem, R. A. Lima, B. Padilha, J. E. Ferreira, and C. Pu, "Robust, extensible, and fast: Teamed classifiers for vehicle tracking in multicamera networks," in *IEEE CogMi*, 2019, pp. 23–32.

[34] J. Sochor, J. Spaˇ nhel, and A. Herout, "Boxcars: Improving fine-grainedˇ recognition of vehicles using 3-d bounding boxes in traffic surveillance," *IEEE transactions on intelligent transportation systems*, vol. 20, no. 1, pp. 97–108, 2018.

[35] X. Zhang, H. Gao, C. Xue, J. Zhao, and Y. Liu, "Real-time vehicle detection and tracking using improved histogram of gradient features and kalman filters," *International Journal of Advanced Robotic Systems*, vol. 15, no. 1, p. 1729881417749949, 2018.

[36] S. Yang, J. Zhang, C. Bo, M. Wang, and L. Chen, "Fast vehicle logo detection in complex scenes," *Optics & Laser Technology*, 2019.

[37] S. C. Hoi, X. Wu, H. Liu, Y. Wu, H. Wang, H. Xue, and Q. Wu, "Logonet: Large-scale deep logo detection and brand recognition with deep region-based convolutional networks," *CoRR*, 2015.

[38] Z. Wang, L. Tang, X. Liu, Z. Yao, S. Yi, J. Shao, J. Yan, S. Wang, H. Li, and X. Wang, "Orientation invariant feature embedding and spatial temporal regularization for vehicle re-identification," in *ICCV*, 2017, pp. 379–387.

[39] X. Liu, S. Zhang, Q. Huang, and W. Gao, "Ram: A region-aware deep model for vehicle re-identification," in *IEEE ICME*, 2018, pp. 1–6.

[40] L. Yang, P. Luo, C. Change Loy, and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," in *CVPR*, 2015, pp. 3973–3981.

[41] K. Panetta, L. Kezebou, V. Oludare, J. Intriligator, and S. Agaian, "Artificial intelligence for text-based vehicle search, recognition, and continuous localization in traffic videos," *AI*, vol. 2, no. 4, pp. 684–704,

2021. [Online]. Available: https://www.mdpi.com/2673-2688/2/4/41
[42] A. Suprem, J. Arulraj, C. Pu, and J. Ferreira, "Odin: Automated drift detection and recovery in video analytics," *Proc. VLDB Endow.*, vol. 13, no. 12, pp. 2453–2465, jul 2020. [Online]. Available: https://doi.org/10.14778/3407790.3407837
[43] Y. Huang, P. Peng, Y. Jin, J. Xing, C. Lang, and S. Feng, "Domain adaptive attention model for unsupervised cross-domain person reidentification," *CoRR*, 2019.
[44] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, S. Li, L. Chen, M. E. Kounavis, and D. H. Chau, "Shield: Fast, practical defense and vaccination for deep learning using jpeg compression," in *SigKDD*, 2018.
[45] M.-T. M. L. for Vehicle Re-Id, "Kanaci, aytac and li, minxian and gong, shaogang and rajamanoharan, georgia," in *CVPR Workshops*, 2019, Conference Proceedings.
[46] Y. Zhou and L. Shao, "Aware attentive multi-view inference for vehicle re-identification," in *CVPR*, 2018, pp. 6489–6498.
[47] S. S. Basha, S. R. Dubey, V. Pulabaigari, and S. Mukherjee, "Impact of fully connected layers on performance of convolutional neural networks for image classification," *Neurocomputing*, vol. 378, pp. 112–119, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231219313803
[48] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *ECCV*, 2018, pp. 3–19.
[49] H. Luo, Y. Gu, X. Liao, S. Lai, and W. Jiang, "Bag of tricks and a strong baseline for deep person re-identification," in *CVPR Workshops*, 2019, Conference Proceedings.
[50] J. Krause, M. Stark, J. deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *3dRR-13*, 2013.
[51] X. Liu, W. Liu, H. Ma, and H. Fu, "Large-scale vehicle re-identification in urban surveillance videos," in *IEEE ICME*, 2016, pp. 1–6.
[52] "National highway traffic safety admin," 2022. [Online]. Available: vpic.nhtsa.dot.gov
[53] M. Buzzelli and L. Segantin, "Revisiting the compcars dataset for hierarchical car classification: New annotations, experiments, and results," *Sensors*, vol. 21, no. 2, p. 596, 2021.