Identification of Spurious Labels in Machine Learning Data Sets using N-Version Validation

Malte Mues
Dortmund University of Technology
Dortmund, Germany
Email: malte.mues@tu-dortmund.de

Sebastian Gerard
Dortmund University of Technology
Dortmund, Germany
Email: sebastian.gerard@tu-dortmund.de

Falk Howar
Dortmund University of Technology
Dortmund, Germany
Email: falk.howar@tu-dortmund.de

***Abstract*— Machine learning components are becoming popular for the automotive industry. More and more data sets become available for training machine learning components. All of them provide ground truth labels for images. The labeling process is expensive and potentially error-prone. At the same time, label correctness defines the business value of a data set. In this paper, we use N-Version approach to assess the label quality in a data set. The approach combines N state-of-the-art neural networks and aggregates their results in a single verdict using majority voting. We analyze this majority vote against the ground truth label and compute the percentage of disagreeing pixels along with other metrics, enabling the automated and detailed analysis of label quality on data sets. We evaluate our methodology by classifying the BDD100K drivable area data set. The evaluation shows that the approach identifies misclassified scenes or inconsistencies between label semantics for similar scenes.**

***Index Terms*—Machine Learning Validation, Data Set Value Assessment, N-Version Redundancy.**

I. Introduction

Robot cars, or fully autonomous vehicles (FAV), that transport people and goods are a dream that has been fueling different areas of research for decades. Some prototypes made remarkable progress, for example, the Bertha-project [1] or the Waymo project[1]. Those prototypes have in common that they rely on machine learning algorithms in components responsible for world perception. Andrej Karpathy, current Director of AI at Tesla, identified a new trend in the way we build software, which he called Software 2.0[2]. As neural networks have a constant processing time for segmenting an image and often beat rule-based systems [2], they are mainly used as an intelligent ingredient for computer vision components. Software 2.0 is all about tuning the data sets instead of programming the algorithms. The question how engineers verify the safety of autonomous systems that use such components is still open. Current approaches [3]–[5] suggest that a test data set of relevant situations is constructed manually. An autonomous driving function has to demonstrate an acceptable performance during the accreditation process instead of a formal verification.

Koopman et al. [6], [7] make a clear point, that testing alone will not demonstrate the safety of an intelligent system in the automotive sector. In addition, they highlight the conflict with the current practice that assumes any part of a system as unsafe as long as a verification does not demonstrate convincingly safety. Given the fact that a car drives around in an open world, it is impossible to base this safety demonstration on an enumeration of all situations the car is going to encounter. Koopman et al. argue that in a certification process it is therefore insufficient for a test oracle to only judge the segmentation results of the neural networks for a fixed set of training situations. Instead, they bring up the argument that the only chance for making the used neural network safe is longer training, better architectures or better input data. All three steps are part of the new Software 2.0 trend and do not match traditional software quality assurance.

Major companies paired up with universities releasing benchmark data for tackling the challenge of better neural networks (e.g. AppolloScape [8], BDD100K [9] and Cityscapes [10]). See [11], [12] for two surveys of other benchmarks. The main value proposition in these benchmarks is that they provide images but along with suitable labels serving as a teacher for learning the image segmentation. The Cityscape data set alone consumed time in the magnitude of 410 entire days $(5.000 * 90min + 20.000 * 7min)$ for the labeling of the images. This is a major investment often not feasible for a single research group.

Reducing the cost of labeling is a key aspect for a research project that aims to introduce a new data set. Hence, the research group behind BDD100K developed scalable annotation tooling that supports the user in the task of annotating the images in the benchmark. One consequence of these publicly available data sets is a gold rush for the best neural network getting the best recall percentage on those data sets. A substantial amount of research today addresses better training and architectures (e.g. [13]–[17]) improving the achieved results on a given data set (c.f. Semantic Segmentation on Cityscapes test[3]). Surprisingly, very few publications discuss the relationship between data sets and suitability for reproducing the real world. While for every newly introduced data set it is demonstrated that more weather situations, more complex scenes, or simply more samples are provided, and that after training a network's

[1]https://waymo.com

[2]https://medium.com/@karpathy/software-2-0-a64152b37c35

[3]https://paperswithcode.com/sota/semantic-segmentation-on-cityscapes

prediction matches the labelling, the quality of labels and its influence on a network's performance in the real word is usually not discussed.

To the best of our knowledge, no uniform metric for measuring the quality of input labels for training an artificial intelligence component currently exists. From our point of view, this metric must have two dimensions: Label consistency and correctness within one data set for ensuring that the teacher has a uniform voice during training and suitability of the input data set for reproducing the real world.

In this paper, we use the n-version methodology to assess the label quality regarding consistency and correctness. The n-version design methodology [18] is used for detecting errors by increased redundancy in the design phase. The wisdom of the crowd has been successfully applied in other research areas, e.g. [19], [20]. We demonstrate and evaluate how the knowledge of different neural networks can be combined with majority votes to assess label correctness and consistency. Moreover, we show how a group of neural networks can pair up for creating a baseline for a potential replacement of an inconsistent label. The approach is evaluated on the drivable area detection challenge part of the BDD100K [9] data set.

Potential Impact. For this paper, we just assess label correctness and consistency using the n-version design methodology. The results we obtained with this methodology on the BDD100K data set are promising. A potential practical application for the method may be the identification of challenging situations in data streams: Whenever there is a sequence of images in the online data that leads to a disagreement quote above a predefined threshold between the neural networks, this scene should be recorded. This way, the uninteresting scenes from test drives are filtered out and the interesting scenes can be added to the training data after manual labeling.

The number of encountered scenes allows to assess how good a given training data set is suitable to reproduce the real world over time. Koopman et al. requested that not only predefined test scenarios are added to the training data set. But extending the training data set with recorded situations from a test car driving in the real world should lead to a decreasing rate of challenging situations identified in new test drives. This marks the point in time the training data set starts to saturate with sufficient scenes for modeling the real world.

The number of encountered challenging situations in a given time period or on a given distance might be used as a metric for the suitability of a data set to model the real world as second dimension of the quality metric. Moreover, the combined knowledge from multiple neural networks might become the missing test oracle in the open world for monitoring intelligent components in the automotive industry.

Related Work. B-Snakes have been demonstrated suitable for detecting lanes by Wang et al. [21] presenting good results using the CHEVP algorithm. Derivations of this approach are widely spread in literature before neural networks appeared. First, we tried to reuse them as an algorithmic test oracle instead of majority voting, but had to give up as guessing parameter is not feasible for arbitrary cases [22].

For faster navigation in large image databases, Hornauer et al. [23] presented a query approach for selecting similar images fast from a database. This might be useful during debugging of neural networks predictions for finding similar images and comparing the labeling of situations our approach identifies to be tricky.

Augmentation is often used for increasing the training label amount in a data set artificially. Prakash et al. [14] present Structured Domain Randomization, a technique that preserves the context of the image during the augmentation phase outperforming normal Domain Randomization. Such optimizations are not yet used during our training phase but might strengthen the prediction results.

Outline. Section II describes the idea for n-version validation. The concrete implementation follows in Section III. Results of the evaluation are presented in Section IV. Finally, Section V concludes the paper and outlines future work.

II. Method

In this section, we first present our approach for automatically identifying images with potentially wrong ground truth labels. Then, we describe the high-level architecture used for implementation and finally the classification targets we are looking for in the data set, inspired by scenes we have analyzed in a prestudy.

When we started this research project, we were interested in neural network prediction correctness. While skimming through images with low predicted intersections over union (IOU), we found out that they had different labels for scenes compared with similar images in the dataset. For example, the images in Figure 2 are similar scenes that are not all labeled in a consistent way. This inconsistency has different effects on the predicted label that harm the IOU results. Therefore, we shifted our focus to assessing the label correctness and consistency automatically. The task at hand is implementing an expert system that is able to segment an image into different features with reliable and constant performance.

A. Approach and Architecture

In the presented case, we are interested in segmenting a given image into three label classes: background, ego drivable area and other drivable area. Humans tend to make mistakes in the task execution. To overcome them, one strategy is to hand out the same task to N persons and aggregate all those N results as has been done for labeling the Cityscapes data set with fine-grained annotations [10]. This redundancy-based strategy exploits the smaller likelihood of making the same mistake repeatedly, if a task is assigned to different individuals. It is called N-version redundancy. We will use the N-version redundancy to exploit the knowledge of our experts, the neural networks predicting a segmentation. This aggregated prediction can be compared to the ground truth label and therefore serves as a quality gate. If the prediction does not fit the ground truth label, a potentially wrong ground

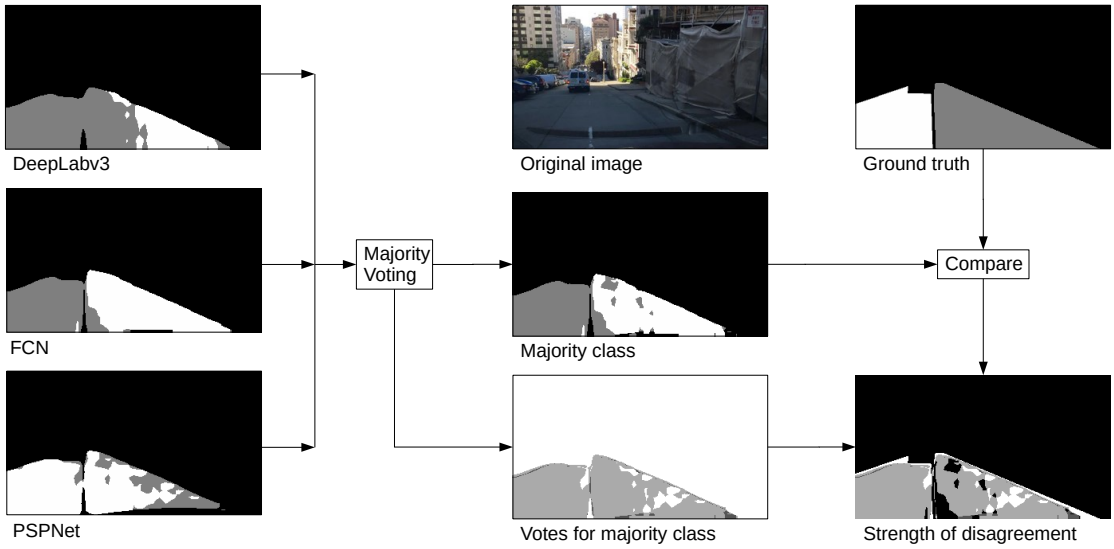

Fig. 1: Segmentations computed by three different neural networks (DeepLabv3, FCN, PSPNet) are aggregated via pixel wise majority voting. For pixels where the ground truth and majority class disagree, the number of votes for the majority class is indicated in the 'strength of disagreement' image. In case of no disagreement, the respective pixel is black, indicating no disagreeing votes. In the case of all neural networks disagreeing with the ground truth, the resulting pixel color is white. Accordingly, one disagreeing vote is represented as dark grey, two votes as light grey. For the images representing a classification, black represents the background class, grey represents the vehicle's ego lane, and white represents other lanes.

truth has been identified. For now, a human has to validate the result manually.

Figure 1 shows the approach we applied for merging the predictions of different neural networks into a single result. On the left are the different implementations, each executing a single segmentation task. In the center is the majority voting component, calculating the aggregated label and how many neural networks have voted for the pixel. Each pixel is classified as the class with most votes. On the right is the ground truth label represented and the result of the comparison between the ground truth and the aggregated label. We call the resulting metric the strength of disagreement. Next, we define the different classification groups.

B. Classification of Label Quality

In total, we defined four classification classes after a prestudy. One for positive match and three for redefining disagreement. We evaluate them later in Table I:

Strong Agreement. This group represents a strong agreement between ground truth and the aggregated prediction. An image classified in this group has at least $93.5\,\%$ of all pixels classified overlapping between both labels. This threshold is the result of analyzing a histogram with percent of disagreeing pixel in the images. There is always some disagreement around the object boundaries that led to a cluster in the section of up to $6.5\,\%$ disagreeing pixel.

Mostly Background. A common situation for which an image is labeled as mostly background is shown in the left of Figure 2. The car is positioned closely behind and between other cars. Therefore, the lanes are only partially visible. The classification rule for 'mostly background' identifies many pictures that are similar, but contain labeled drivable areas in the ground truth (c.f. right of Figure 2). Therefore, it is unclear what a correct learned result would be. We divided this group into three categories: The ground truth and the majority vote agree, the ground truth defines all as background and the majority vote disagrees, and the majority vote defines all as background but the ground truth does not.

Difficult Situations. When aggregating the majority vote classification (see Figure 1), there are some areas for which

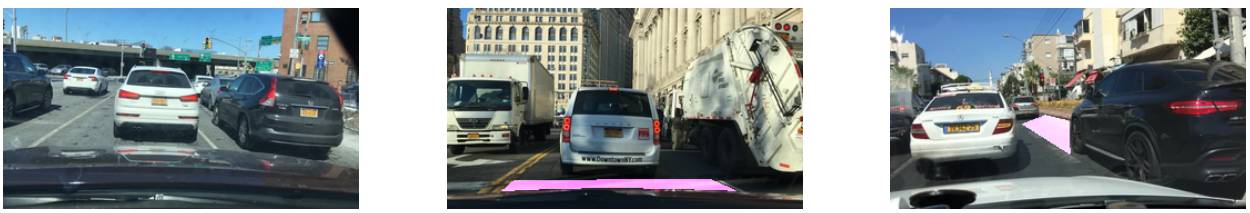

Fig. 2: Some traffic situations appear to be labeled inconsistently in the ground truth. The leftmost image is completely labeled as background, even though lanes (and therefore drivable areas) are clearly visible. The center and the right image show very similar situations, but do have parts labeled as drivable area (marked in pink).

the neural networks cannot find unanimous consent among each other. These areas are apparently hard to classify for the neural networks. This might either be due to the situation being inherently difficult (low lighting, very unusual situation, etc.), or possibly due to similar situations being labeled inconsistently in the training data. With three neural networks and three classes in the segmentation, these situations will usually manifest as 2 vs. 1 vote. The alternative, a three-way tie, would require one network to label a pixel as background, while the other two predict it to be the ego area, and other area, respectively.

Swapped Drivable Areas. A lot of images rated with low per-class IOU scores had the simple problem of judging differently than the ground truth which drivable area is the ego area and which is the area of other drivers. However, the existence of this category is mostly an expression of how difficult a binary classification is. The problem usually occurs when the car is in the process of switching between lanes. In this case, it is not clear when the area changes roles between ego and other area, or whether at some point both areas should be labeled as ego or other.

III. Implementation

This section introduces the three main components that we used for evaluating the suggested methodology: An image segmentation task, a set of prediction methods and a majority voting procedure.

A. Segmentation Task: BDD100k Drivable Area

We use the BDD100K [9] data set, which introduces the drivable area segmentation challenge. The full data set consists of 100.000 labeled images. It is split into 70.000 training, 20.000 validation and 10.000 test images.

The images are labeled with three different classes: 1. a drivable area for the ego vehicle, 2. additional drivable area involving a lane change or interaction with other traffic participants, 3. anything else. Since the test images do not have any associated ground truth labels, we do not use them. The networks are trained on the training images and the classification is evaluated on the validation images.

We resized all images and labels from 1280x720 to a size of 321x185 pixels to reduce computational resource needs. The unusual width and height are the result of the size limitations imposed by the PSPNet. Long et al. [13] have shown, that downsizing images by a factor of 4 does not harm the prediction capacity of a neural network significantly. Therefore, this constraint is not a threat to the approach.

B. Neural Networks for Semantic Segmentation Predictions

We will use three different neural networks as the prediction method in our N-version setup. The architectures we use share a very similar structure. The first part is a feature extractor, which computes general image features. As a feature extractor, or backbone, we use the Resnet101 [15] architecture in all neural networks. The second part of the architecture uses these image features to form a prediction. A common approach for this second part is to use non-local information (context information) to improve the prediction for each pixel.

The used networks are:

- a Fully Convolutional Network (FCN) [13]
- the Pyramid Scene Parsing Network (PSPNet) [16]
- the DeepLabv3 architecture [17].

Fully Convolutional Network. The fully convolutional network we employ consists only of the backbone network, and a *head* that uses the backbone features for straight forward classification. The head consists solely of a convolution, followed by BatchNorm [24], ReLU activation function, a dropout layer, and a second convolution that outputs the classification results. Therefore, it has no access to non-local information.

Pyramid Scene Parsing Network. The PSPNet architecture uses a multi-scale pooling approach to access global context information. It employs four different scale factors: 1, 2, 3, and 6, for which the context features are generated. For each scale factor s, the feature map generated by the feature extractor is divided along height and width by the scale factor. This way, $s * s$ different sub-areas are created. Max-pooling is applied to each of them, generating between 1 and 36 different features, depending on the scale factor. The resulting features are then up sampled to the size of the initial feature map and concatenated with it. This way, the existing features are enriched with information collected at the level of the different sub-areas.

DeepLabv3 DeepLabv3 uses image-level features via global average pooling. Additionally, it generates multi-scale features via *atrous convolutions*, also known as *dilated convolutions*. These are very similar to regular convolutions, except that their field of view is larger, thereby giving access to non-local features. For example, a $3x3$ atrous convolution with rate $r = 3$ inspects nine points on a $7x7$ area. A regular $3x3$ convolution (equivalent to an atrous convolution with $r = 1$) also inspects nine points, but on a $3x3$ area. As can be seen, the number of required parameters is not increased. Only the area from which the points are sampled is increased, thereby allowing for non-local features, without changes in cost.

Both PSPNet and DeepLabv3 aggregate their non-local features by concatenating them to the original image features. They then produce a classification for each pixel by finishing their architecture with a comparable structure used for the head of the FCN architecture. Due to these differences, each of the network solves the task at hand differently. These differences are sufficient to qualify the approaches as different solutions to the given problem, a requirement of the N-version approach. Otherwise, structural errors might be hidden by too much redundancy. The left part of Figure 1 shows an example of the different predictions produced by the three networks for the same image.

C. Training Phase.

To train the neural networks, we use Zhao's Pytorch [25] implementation [26] of PSPNet [16], and the torchvision [27] implementation of a version of a fully convolutional network

Category	Classification rule	Images
Strong agreement	DisagRate<= 0.065	8488
MV: all BG, GT disagrees	(BgRateMv >0.999) and (DisagRate >0.0)	61
GT: all BG, MV disagrees	(BgRateGt = 1.0) and (DisagRate >0.0)	159
GT: all BG, MV agrees	(BgRateGt = 1.0) and (DisagRate = 0.0)	295
Difficult situations	DivVotes >= 0.15	792
Swapped Areas	(DisagRate - SwappedDisag) >= 0.01	256

TABLE I: Based on different measures, images are sorted into categories. The first category represents images for which the majority vote predictions mostly agree with the ground truth. The second category contains images that have been classified as mostly background by the majority vote (MV), but do contain non-background labels in the ground truth (GT). The next two categories contain images that are labeled completely as background in the ground truth. The majority voting result disagrees with this background label in the first category, and agrees with it in the second one. Images in the 'Difficult situation' category have strong uncertainty in the majority voting process, with larger parts of the image not being unanimously agreed upon by the three models. The last category contains images, for which the disagreement with the ground truth sinks by at least 1% when swapping the area labels in the majority vote prediction.

(FCN) [13] and DeepLabv3 [17]. Each of those networks is instantiated with a Resnet101 [15], which is used as a feature extractor. For preprocessing training images before feeding them into the network, we used the transformation steps included in Zhao's project, i.e., random scaling, rotation, Gaussian blur, horizontal flipping, and normalization via mean and standard deviation. During the validation phase, only the normalization was applied. For details on the respective preprocessing parameters, see [26].

After every five epochs, we saved the current model state and evaluate it on the validation data set. For each architecture, we then chose the model with the highest mean IOU on the validation set. This way, we selected one model per architecture and used them in the majority vote method. The code we used, including the specific hyperparameters used for training each of the models, are available online.[4]

D. Majority Vote Procedure

This section introduces majority voting to combine the predictions of the different neural networks into a single prediction. We assume that sections of images that are difficult to judge will receive varying predictions by different neural networks, while simple-to-predict areas should receive the same predictions. To distinguish these kinds of areas, we aggregate the different predictions via majority voting. For every pixel, we consider the class predicted by each neural network. An eventual probabilistic prediction is flattened to a single vote ignoring any uncertainty metric in the prediction. A complex voting procedure respecting uncertainty might be a vector for improving the approach. A *vote* is the class a network predicts. The resulting class with most votes is called *majority class*. For a tie, the numerically lower class is used as a tie-break. The numerically order of classes is: background (0), ego area (1), other area (2). Hence, background beats all areas and ego the other area in a tie.

[4]https://github.com/tudo-aqua/n-version-label-validation.git

Architecture	Background	Ego lane	Other lanes	Mean IOU
FCN	0.950	0.692	0.570	0.733
Deeplabv3	0.949	0.697	0.563	0.737
PSPNet	0.973	0.828	0.717	0.840

TABLE II: Per-class and mean intersection over Union (IOU), as measured on the validation set, for the best models within each of the three architectures. The best model was determined by choosing the one with the highest mean IOU on the validation set.

To categorize the images (see Table I), we use different measures. The *ratio of background pixels in majority vote* (BgRateMv) is the percentage of pixels majority-predicted to belong to the background class. The *ratio of background pixels in ground truth* (BgRateGt) is the same for the ground truth. The *ratio of disagreeing pixels* (DisagRate) is the percentage of pixels that are different between the ground truth and the majority vote prediction. The *mean divergent votes* measure (DivVotes) is the mean number of votes per pixel that do not agree with the majority class. The *swapped lanes disagreement* (SwappedDisag) represents the percentage of disagreeing pixels for swapped area labels.

IV. Experiment and Discussion

We have evaluated the majority vote method described in Section II on the BDD100K drivable area data set. In this section, we present the obtained results and discuss them. We first describe network performance using IOU, then briefly the obtained classification results and give two examples of label inconsistency.

Setup. We trained three different network architectures on the BDD100k drivable area data set, described in section III-A. The resulting values of the IOU measure are presented in Table II. The PSPNet architecture clearly outperforms FCN and DeepLabv3. This might be caused by the PSPNet architecture already using training parameters optimized for use with the Cityscapes data set, which is rather similar to the BDD100k data set. We did not perform any hyper parameter optimization for the FCN and DeepLabv3 architectures provided by the torchvision package.

Results. Table I presents the results of the label classification. From the 20.000 images in the validation set, roughly half of them are classified by our rule. Only 8488 are classified in the *Strong Agreement* class. Given the mean IOUs ranging from 0.73 to 0.84 for the single networks, we are surprised that around $42\,\%$ of the aggregated prediction reaches an accuracy greater 0.935. The implication is, that for $58\,\%$ of those pictures, a reliable prediction on the validation test data has not been possible even for the aggregated vote. The high number of unclassified labels points out, that the few identified situations in the prestudy are not sufficient to explain misclassifications completely. Further research is required here.

Fig. 3: On the left, we see a three lane one-way street labeled completely as drivable ego area in red. In the center a similar street is labeled as three different lanes, blue marking other drivable area. In variation from the label, the complete street is predicted by the networks as ego lane. On the right side, an inner roundabout lane has a large label for drivable ego area. The group of networks identified some areas which are potentially not part of the ego roundabout lane marked in blue.

We found 515 instances that belong to scenes similar to Figure 2. The results demonstrate that the inconsistency leads to uncertainty in how labels should predict drivable areas in this situation. Further, we could identify 792 images that are hard to segment for the neural networks. Manual inspection shows that different camera angels and sometimes obstacles in the field of view might explain some of them. Further, situations that are hard to predict due to lighting, shadows or weather conditions end up in this class. This group creates many insights as it provides hints for conditions making a prediction system fail.

From the 256 instances in the swapped areas class, we will discuss two examples that revealed questionable ground truth labels. The first example is in the left and center columns of Figure 3. Both images present a similar situation: A three lane street, all lanes heading in the same direction. The neural networks generated a comparable prediction for the scene, but the ground truth labels do not. Therefore, the swapped areas rule triggers on the center image. From our point of view, the center image is more precisely labeled in the ground truth than the image on the left. It seems the predictions from the networks are stronger influenced from labels similar to the one used in the left column. The majority label clearly shows that the networks in the prediction do not consider the other lanes as an option in such situations. This is a hint for an inconsistency in the data set with potentially tremendous effects for real world segmentation.

The second example, in the right of Figure 3, shows the inverse case. For a lane in a roundabout the complete area is marked as drivable by the ego vehicle. Somehow, the networks could construct from the other ground truth label sufficient evidence to conclude that the ground truth is wrong. The majority voted label show that the other lanes in the roundabout and the exit lanes are other drivable area. We think, the networks made a prediction better than the ground truth. Apparently, the weaker ground truth could not dominate the learned effects through negative discrimination during the training phase. With joint forces, the network identified a potentially deadly mistake in segmenting the roundabout situation.

Discussion. Overall, the obtained result proves that the proposed method is suitable for guiding human attention during the label development for a dataset. It constitutes a metric that can be computed automatically and gives more insights in the network performance than the IOU alone. Moreover, the classification helps to get a better understanding of potential inconsistencies and semantically wrong labels by showing only problematic instances as demonstrated in the examples. It is worth to investigate the unclassified images to learn more about potential other sources of misclassification and extend the classification rule set.

Threats to Concept Validity. If two networks perform very similar, it could occur that the majority vote reduces to a constant 2 vs. 1 situation, with two networks always overruling the third one. We validate our approach by computing how often the networks agree with each other. FCN and PSPNet agree on 96.3% of pixels, FCN and DeepLabv3 on 98.1%, and PSPNet and DeepLabv3 on 96.2%. This

shows that the two networks that have a similarly strong performance (FCN and DeepLabv3) also tend to agree more often in their predictions, which is to be expected. However, the difference to the agreement with the PSPNet predictions is not very large, so we do not believe that this should cause any problems in the majority vote.

V. Conclusion and Future Work

We show that aggregating the verdicts of different neural networks is a suitable approach for identifying potential inconsistencies between labels of similar features in a data set, following the N-Version redundancy strategy. We implemented the approach using 3 neural networks and evaluated it on the BDD100K drivable area data set. The evaluation goal is the classification of the data set into labels where the aggregated prediction strongly agrees with the ground truth label, where the image seems to be labeled as background only, where the neural networks disagree on the verdict, or where swapping the drivable areas in the ground truth label leads to less disagreement.

The initial results demonstrate that the aggregated labels of multiple predictions sometimes contain better knowledge of the feature of interest in the label than the ground truth labels. Only $42\,\%$ of the 20.000 validation images reach a strong agreement with the ground truth label. We could further identify 515 images as background only, 792 images as difficult for segmentation, and 256 instances that achieve better IOU values after swapping the areas in the ground truth. We show how examples from these buckets might be used to find inconsistencies in the ground truth label.

In the future, such an approach might be used as a filter for identifying complicated situations during test drives or assessing the label quality of a new and unknown data set. We are convinced that the label quality influences the monetary value of a data set in the long term. From our point of view, this assessment of existing labels is a first step towards automated label repair because it can identify wrong labels based on the learning progress of system components. The missing technique for a fully autonomous approach is an automated label correction. However, the majority voted label might serve as a baseline for a corrected version.

References

[1] U. Franke, D. Pfeiffer, C. Rabe, C. Knoeppel, M. Enzweiler, F. Stein, and R. Herrtwich, "Making bertha see," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2013, pp. 214–221.

[2] N. Monot, X. Moreau, A. Benine-Neto, A. Rizzo, and F. Aioun, "Comparison of rule-based and machine learning methods for lane change detection," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 198–203.

[3] A. Pütz, A. Zlocki, J. Bock, and L. Eckstein, "System validation of highly automated vehicles with a database of relevant traffic scenarios," *situations*, vol. 1, pp. 19–22, 2017.

[4] C. Amersbach and H. Winner, "Functional decomposition: An approach to reduce the approval effort for highly automated driving," in *8. Tagung Fahrerassistenz*, 2017.

[5] H. Winner, K. Lemmer, T. Form, and J. Mazzega, "Pegasus—first steps for the safe introduction of automated driving," in *Road Vehicle Automation 5*, G. Meyer and S. Beiker, Eds. Cham: Springer International Publishing, 2019, pp. 185–195.

[6] P. Koopman and M. Wagner, "Challenges in autonomous vehicle testing and validation," *SAE International Journal of Transportation Safety*, vol. 4, no. 1, pp. 15–24, 2016.

[7] ——, "Autonomous vehicle safety: An interdisciplinary challenge," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 1, pp. 90–96, 2017.

[8] P. Wang, X. Huang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, "The apolloscape open dataset for autonomous driving and its application," *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[9] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving video database with scalable annotation tooling," *arXiv preprint arXiv:1805.04687*, 2018.

[10] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[11] J. Guo, U. Kurup, and M. Shah, "Is it safe to drive? an overview of factors, metrics, and datasets for driveability assessment in autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[12] H. Yin and C. Berger, "When to use what data set for your self-driving car algorithm: An overview of publicly available driving datasets," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Oct 2017, pp. 1–8.

[13] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[14] A. Prakash, S. Boochoon, M. Brophy, D. Acuna, E. Cameracci, G. State, O. Shapira, and S. Birchfield, "Structured domain randomization: Bridging the reality gap by context-aware synthetic data," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 7249–7255.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[16] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2881–2890.

[17] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[18] A. Avizienis, "The n-version approach to fault-tolerant software," *IEEE Transactions on software engineering*, no. 12, pp. 1491–1501, 1985.

[19] A. Kittur and R. E. Kraut, "Harnessing the wisdom of crowds in wikipedia: quality through coordination," in *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, 2008, pp. 37–46.

[20] W. Pan, Y. Altshuler, and A. Pentland, "Decoding social influence and the wisdom of the crowd in financial trading network," in *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*. IEEE, 2012, pp. 203–209.

[21] Y. Wang, E. K. Teoh, and D. Shen, "Lane detection and tracking using b-snake," *Image and Vision computing*, vol. 22, no. 4, pp. 269–280, 2004.

[22] R. Heij, M. Helgers, W. Kockelkorn, and R. Smelik, "Snakes for lane detection," https://pdfs.semanticscholar.org/e81c/0309564942ec4297d06dca1f7c49f92283cf.pdf, 2006.

[23] S. Hornauer, B. Yellapragada, A. Ranjbar, and S. Yu, "Driving scene retrieval by example from large-scale data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 25–28.

[24] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[26] H. Zhao, "semseg," https://github.com/hszhao/semseg, 2019.

[27] PyTorch, "Torchvision," https://github.com/pytorch/vision, 2019.