Evidential deep learning for arbitrary LIDAR object classification in the context of autonomous driving

Edouard Capellier, Franck Davoine, Véronique Cherfaoui, You Li

Evidential deep learning for arbitrary LIDAR object classification in the context of autonomous driving

Edouard CAPELLIER[1,2], Franck DAVOINE[2], Veronique CHERFAOUI[2], You LI[1]

Abstract— **In traditional LIDAR processing pipelines, a point-cloud is split into clusters, or objects, which are classified afterwards. This supposes that all the objects obtained by clustering belong to one of the classes that the classifier can recognize, which is hard to guarantee in practice. We thus propose an evidential end-to-end deep neural network to classify LIDAR objects. The system is capable of classifying ambiguous and incoherent objects as unknown, while only having been trained on vehicles and vulnerable road users. This is achieved thanks to an evidential reformulation of generalized logistic regression classifiers, and an online filtering strategy based on statistical assumptions. The training and testing were realized on LIDAR objects which were labelled in a semi-automatic fashion, and collected in different situations thanks to an autonomous driving and perception platform.**

I. INTRODUCTION

Detecting and recognizing road users is paramount for autonomous vehicles that are intended to drive on public road. 3D sensors, and especially LIDAR scanners, seem particularly suitable for those tasks. In parallel, unmanned ground vehicles that follow the standard 4D/RCS model [1] rely on processing pipelines that include a segmentation step -to detect objects- and a classification step -to infer the type of each detected object. Using similar design choices in the context of autonomous driving, when working with LIDAR raw data, thus appears natural.

However, a classifier used within such a processing pipeline should be able to cope with any possible object generated during the segmentation step, and always output pertinent results. A naive way to cope with this requirement would be to collect large amounts of data which would be accurately labeled afterwards, and to train a classifier on the resulting dataset. Unfortunately, this method is not guaranteed to cover all the randomness that an autonomous vehicle is likely to meet on public roads. This would then lead to errors in situation understanding. For instance, in the situation in Fig. 1, if a pedestrian detector were to consider that the poles on the sides of the roundabout are pedestrians because it wasn't trained to reject them, this would falsely complexify the situation understood by the vehicle.

A more realistic way to grapple with this randomness might be to use classifiers that are able to classify objects as unknown, while having only been trained on known objects. The evidential theory, or Dempster-Shafer theory, in which the unknown is explicitely represented, seems of particular use. Nevertheless, this approach also has two main limits. First evidential labels, in which the fact of not knowing is quantified, are hard to obtain. Then, evidential classifiers usually rely on a closed-world assumption [2]: objects classified as unknown are usually ambiguous ones with regards to the training dataset.

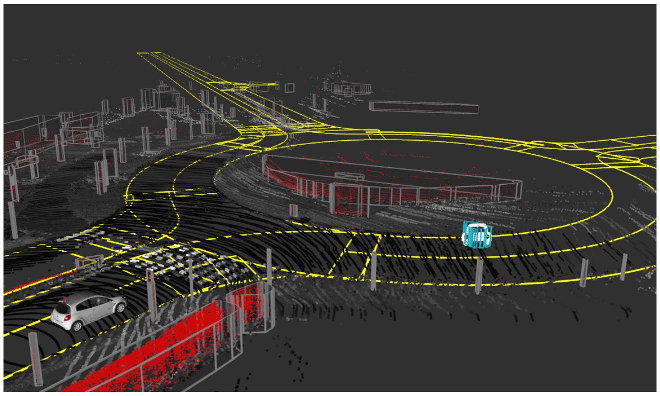

Fig. 1: Example of output from a LIDAR perception pipeline using the proposed classifier. The yellow lines correspond to a pre-existing map of the scene; the red LIDAR points belong to obstacles ; the grey 3D polygons represent objects classified as unknown objects; the blue 3D polygon represents an object classified as vehicle.

To address those two limits, a multi-task multi-layer perceptron (MLP) is trained on labelled LIDAR objects, and its outputs are reinterprated as a fusion of evidential mass functions. This is accomplished via an extension of T. Denoeux's recent work on generalized logistic regression (GLR) classifiers [3], which enables statistically incoherent features to be filtered. This work only aims at classifying vehicles (cars, trucks) and vulnerable road users (pedestrians, two-wheeled vehicles), while classifying other objects as unknown without having represented them explicitely in the training dataset. Fig. 1 highlights the interest of such a system: the object that was not classified as an unknown one was the vehicle in the roundabout, although the classifier wasn't trained on the other objects. The main contributions of the presented work are then:

- A lightweight multi-layer perceptron architecture, to classify LIDAR objects and reject unknown ones
- An end-to-end reinterpretation of GLR classifier as a fusion of mass functions
- A simple online statistical filtering mecanism to detect statistically incoherent objects

The system was evaluated on real-life LIDAR objects that were collected thanks to an autonomous and perception

*This work is supported by a CIFRE fellowship from Renault S.A.S

[1] Renault S.A.S, 1 av. du Golf, 78288 Guyancourt, France. Contact: name.surname@renault.com

[2] Sorbonne Universités, Université de technologie de Compiègne, CNRS, HeuDiaSyc, Centre de recherche Royallieu, CS 60319, 60 203 Compiegne cedex, France. Contact: name.surname@hds.utc.fr

platform, and labelled in a semi-automatic fashion.

II. LITTERATURE REVIEW

A. Classification of LIDAR objects and points

Although evidential theory is widely used to generate and analyze occupancy grids from pre-processed LIDAR scans [4]–[7], very few works use it for LIDAR object classification, as it was done in [8]. Yet, in those works, evidential mass functions are derived from heuristics and prior geometrical assumptions.

Regarding non-evidential LIDAR classifiers, two main state-of-the-art approaches coexist nowadays. First, Wu et al. [9] introduced SqueezeSeg, and proposed to process LIDAR points as a dense range image representing spherical coordinates, thanks to classical convolutional neural networks and conditional random fields. This work was extended in [10], [11]. Yet, before being used within actual autonomous systems, those approaches need coupling with an object detection algorithm. As processing a LIDAR scan twice, first for point-level classification and then for object detection, seems inefficient, the practical interest of these methods for autonomous vehicles is limited.

Another popular option is to directly process raw point-clouds thanks to derivatives of the PointNet architecture, introduced by Qi et al. [12] PointNet applies a multi-layer perceptron to each individual point, and produces a feature vector describing the whole pointcloud by applying a max operator to the features extracted from each point. PointNet was successfully coupled with an image-based object detector and classifier, to perform 3D bounding-box regression [13]. However, the PointNet architecture suffers from several drawbacks. First of all, it requires a fixed number of input points. Secondly, PointNet usually expects normalized and constrained inputs. This makes the architecture improper when aiming to process large-scale LIDAR scans [14] or size-varrying LIDAR objects, and is limitting when trying to build classifiers that have to cope with any possible LIDAR object. Those intrisic limitations of PointNet might be overcome by jointly performing object detection and classification.

B. Joint LIDAR-object detection and classification

Yan et. al. [15] proposed to split a LIDAR scan into equally-sized voxels and to apply PointNet to the points enclosed in each voxel. Additional convolutions followed by a region proposal network are then used to predict bounding box dimensions and an objectness score for each voxel. However, this approach is computationnally challenging. Indeed, the bounding box parameters and regression scores are only calculated for a single class, and different models are needed for each class.

Simon et. al. [16] thus proposed to perform bounding box regression and multi-class classification jointly by training an image-domain object detector on feature grids generated from a LIDAR scan. Albeit this approach runs at a high framerate, its general performances are significantly worse than LIDAR-domain classifiers.

State-of-the-art LIDAR classification algorithms need coupling with performant object detectors, and joint LIDAR object detection and classification is still challenging. Thus, a simpler classifier was adopted in the context of this work, so as to be agnostic to any object detector, and to focalize on unknown object detection.

C. Detecting unknown objects in machine learning

Machine learning algorithms are usually designed to work with data from relatively constrained domains. Extensive research is however being done in uncertainty modelling and outliers rejection. An efficent uncertainty modelling procedure would be valuable when designing machine learning algorithms that should be able to classify unknown objects. Bayesian neural networks (BNN) [17] follow a probabilistic interpretation of neural networks, allowing their classification uncertainty to be modelled. Gal et al. [18] recently proposed to infer the distribution of the weights of a BNN by using dropout during multiple inferences on the same input, as a Monte-Carlo sampling technique [18]. This technique was recently used for vehicle detection in LIDAR feature maps [19]. However, exhaustively sampling the weights of a neural network is computationnally challenging. And worse, this sampling scheme leads to a nondeterministic behavior, as each inference on a given input generates a different output. Machine learning algorithms that are explicitely designed to reject outliers thus seem more usable in practice.

One-class classifiers are particularly of interest for unknown object detection. Those classifiers are trained on a single-class dataset, and are expected to reject objects that do not belong to the class of the training set. Practical one-class learning applications rely on one-class SVMs [20], which lack evidential or probabilistic interpretation. One-class learning was also recently successfully addressed in the image domain, by using a convolutional neural network trained in parrallell on a multi-class dataset and a one-class dataset [21].

Closer to our work, Sensoy et. al. [22] employed the evidential theory while training deep neural networks to perform multi-class classification. Under the assumption that a belief mass assignment follows a Dirichlet distribution, a specific loss function was defined. However, no actual way to classify objects as unknown was available with this approach, and the mass values were not actually used: the sum of the masses on non-singleton sets was just only considered as an uncertainty indicator. The model from T. Denoeux in [3], which reinterprets GLR classifiers as a fusion of evidential mass functions, was thus extended instead.

III. EVIDENTIAL END-TO-END FORMULATION OF BINARY LOGISTIC REGRESSION CLASSIFIERS

Although both multi-class and binary GLR classifiers can be seen as a fusion of mass functions [3], only the binary case was considered. Indeed, the multi-class case leads to more complex models and decision rules, and the main focus of the present work is only to detect vehicles and vulnerable road users while accounting for unknown objects. Multi-class classification is thus not needed in this context.

A. *The evidential framework*

Let $\Theta = \{\theta_1, ..., \theta_n\}$ be a finite set of all the possible answers to a question. An evidential mass function m is a mapping $m : 2^\Theta \to [0, 1]$ such that $m(\emptyset) = 0$ and

$$\sum_{A \subseteq \Theta} m(A) = 1 \tag{1}$$

In the binary case, $n = 2$, and $2^\Theta = \{\emptyset, \theta_1, \theta_2, \Theta\}$. Then, $m(\theta_1)$ represent the amount of evidence towards the fact that the answer is θ_1, and $m(\Theta)$ is the evidence towards the fact that nothing can be said about the answer (i.e. it is unknown). An evidential mass function m is *simple* if $\exists \theta_i \subset \Theta, m(\theta_i) = s, m(\Theta) = 1 - s$.

Let $w = -ln(1 - s)$ be the *weight of evidence* associated to simple mass function m; m can be represented as $\{\theta_i\}^w$. Let $\oplus$ be the classical Dempster-Shafer operator used to fuse evidential mass functions [23]. Then $\{\theta_i\}^{w_1} \oplus \{\theta_i\}^{w_2} = \{\theta_i\}^{w1+w2}$. Evidential mass functions can be converted into probabilistic mass functions via the so-called *plausibility transformation* [24]. Let the quantity noted $Bel(A) = \sum_{B|B \subseteq A} m(B)$ be the belief on A. Let the quantity noted $Pl(A) = \sum_{B \cap A \neq \emptyset} m(B)$ be the plausibility on A. Then, for $\theta_i \subset \Theta$, a probabilistic mass value towards θ_i, and noted $p_m(\theta_i)$, can be obtained as follows:

$$p_m(\theta_i) = \frac{Pl(\theta_i)}{\sum\limits_{\theta_j \in \Theta} Pl(\theta_j)} \tag{2}$$

B. *Binary generalized logistic classifiers*

Let a binary classification problem with $X = (x_1, ..., x_d)$, a d-dimensional input vector, and $Y \in \Theta$ a class variable. Let $p_1(x)$ be the probability that $Y = \theta_1$ according to the fact that $X = x$. Then $1 - p_1(x) = p_2(x)$ is the corresponding probability that $Y = \theta_2$. Let w be the output of a binary logistic regression classifier, trained to solve the aforementioned classification problem. A generalized binary logistic regression classifier corresponds to the case where there exists a C-dimensional vector x_c and such that $x = (\phi_1(x_c), ..., \phi_d(x_c))$. Then, $p_1(x)$ is such that:

$$p_1(x) = S(w) = S(\sum_{i=1}^{d} \beta_i \phi_i(x_c) + \beta_0) \tag{3}$$

with S being the sigmoid function, and the β values being usually learnt alongside those of the potentially non-linear ϕ_i mappings. In Eq. 3, w exactly corresponds to the output of a multi-layer perceptron trained as a binary GLR classifier.

C. *Binary GLR classifiers as a fusion of simple mass functions*

The sigmoid function is strictly increasing. Then, in Eq. 3, the larger w is, the larger $p_1(x)$ is and the smaller $p_2(x)$ is. Moreover, w can be rewritten as follows:

$$w = \sum_{j=1}^{d} w_j = \sum_{j=1}^{d} (\beta_j \phi_j(x_c) + \alpha_j) \tag{4}$$

with

$$\sum_{j=1}^{d} \alpha_j = \beta_0 \tag{5}$$

Each w_j can then be seen as piece of evidence towards θ_1 or θ_2, depending on its sign. Let us assume that the w_j values are weights of evidence of simple mass functions, denoted by m_j. Let $w_j^+ = max(0, w_j)$ be the positive part of w_j, and let $w_j^- = max(0, -w_j)$ be its negative part. Whatever the sign of w_j, the corresponding m_j can be written as

$$m_j = \{\theta_1\}^{w_j^+} \oplus \{\theta_2\}^{w_j^-} \tag{6}$$

Under the assumption that all the m_i mass functions are independent, the Dempster-Shafer operator can be used to fuse them together. The resulting mass function obtained from the output of the binary logistic regression classifier, noted m_{LR} is as follows:

$$m_{LR} = \oplus_{j=1}^{d} (\{\theta_1\}^{w_j^+} \oplus \{\theta_2\}^{w_j^-}) = \{\theta_1\}^{w^+} \oplus \{\theta_2\}^{w^-} \tag{7}$$

with $w^+ = \sum_{j=1}^{d} w_j^+$ and $w^- = \sum_{j=1}^{d} w_j^-$. From Eq. 7, m_{LR} can be expressed as follows:

$$m_{LR}(\theta_1) = \frac{[1 - \exp(-w^+)] (\exp(-w^-))}{1 - K} \tag{8a}$$

$$m_{LR}(\theta_2) = \frac{[1 - \exp(-w^-)] (\exp(-w^+))}{1 - K} \tag{8b}$$

$$m_{LR}(\Theta) = \frac{\exp(-w^+ - w^-)}{1 - K} \tag{8c}$$

$$\text{with} \tag{8d}$$

$$K = \left[1 - \exp(-w^+)\right] \left[1 - exp(-w^-)\right] \tag{8e}$$

By applying the plausibility transformation in Eqs. 2 to 8, the following probability can be obtained:

$$p_{m_{LR}}(\theta_1) = S(w) \tag{9}$$

which exactly corresponds to the output the GLR classifier, depicted in Eq. 3. This means that any binary GLR classifier can be seen as a fusion of simple mass functions, that can be derived from its parameters. In the case of a multi-layer perceptron, its output can be converted into a mass function via Eqs. 8, only using the output from its penultimate layer and the parameters of its final layer. However, the α_i values in Eq. 5 have to be estimated.

IV. End-to-end evidential interpretation of a binary GLR classifier and online statistical filtering

T. Denoeux proposed to explicitely compute the α_i values after the training, so that the resulting simple mass functions are the most uncertain ones. This means that the weights of evidence of those mass functions should be as small as possible, which leads to the following minimization problem [3]:

$$minf(\alpha) = \sum_{i=1}^{n} \sum_{j=1}^{d} (\beta_j \phi_j(x_i) + \alpha_j)^2 \tag{10}$$

with $\{(x_i, y_i)\}_{i=1}^{n}$ being the training dataset, and $\alpha = (\alpha_i, ...\alpha_d)$.

However, this minimization problem can be instead solved during the training, under the assumption that the last layer of the MLP performs Batch Normalization [25] over each value of its input vector. Let $\upsilon(x_c) = (\upsilon_1(x_c), ..., \upsilon_d(x_c))$ be the mapping modelled by all the consecutive layers of the MLP but the last one ; let $\overline{\upsilon_j}$ be the mean value of the υ_j function on the training set, and $\sigma(\upsilon_j)^2$ its corresponding variance. The output of the MLP depicted in Eq. 3 then becomes:

$$p_1(x) = S(w) = S(\sum_{j=1}^{d}(\beta_j \frac{\upsilon_j(x_c) - \overline{\upsilon_j}}{\sqrt{\sigma(\upsilon_j)^2 + \epsilon}}) + \sum_{j=1}^{d} \alpha_j) \quad (11)$$

If ϵ can be neglected with regards to the $\sigma(\upsilon_j)^2$ values, the minimization problem in Eq. 10 becomes after development:

$$minf(\alpha) = n \sum_{j=1}^{d} \beta_j^2 + n \sum_{j=1}^{d} \alpha_j^2 \quad (12)$$

The minimization problem can then be trivially solved in an online fashion, by simply applying weight decay to the parameters of the final Batch Normalizations. Applying Batch Nomalization to the final layer of the MLP also has another interest: it can be the basis for an online statistical filtering scheme.

Let $z(\upsilon_j(x_i)) = \frac{\upsilon_j(x_i) - \overline{\upsilon_j}}{\sigma(\upsilon_j)}$ be the Z-score of $\upsilon_j(x_i)$. Under the assumption that the υ_j function can be modelled as a random variable following a normal distribution, a simple thresholding can be used to define confidence levels: the larger the Z-score is, the more unlikely to happen $\upsilon_j(x_i)$ is. Moreover, the Central Limit Theorem states that a sum of independent random variables can be modelled as a normal distribution [26]. If the MLP mainly implements linear functions, the $\upsilon_j(x_i)$ values can be approximately considered as sums of random variables. Statistically abnormal $\upsilon_j(x_i)$ values can then be rejected by a simple thresholding on their Z-score.

Again under the assumption that ϵ can be neglected with regards to the $\sigma(\upsilon_j)^2$ values, the w_j values in Eq. 4 can be seen as:

$$w_j \approx \beta_j * Zscore(\upsilon_j(x_c)) + \alpha_j \quad (13)$$

When trying to classify inputs without any guarantee that only pertinent objects will be passed to the classifier, the Z-Score can be used to detect objects that are extremely different from the training set, and should then be classified as unknown. Abnormal objects with regards to the application domain of the classifier can then be easily accounted for, by introducing an additional hyperparameter. Let $ZMax$ be a threshold value. During inference, each w_j can then be computed as follows:

$$w_j = \begin{cases} 0, \ if \ |\frac{\upsilon_j(x_i) - \overline{\upsilon_j}}{\sqrt{\sigma(\upsilon_j)^2 + \epsilon}}| > Zmax \\ \\ \beta_j * \frac{\upsilon_j(x_i) - \overline{\upsilon_j}}{\sqrt{\sigma(\upsilon_j)^2 + \epsilon}} + \alpha_j, \ otherwise \end{cases} \quad (14)$$

According to Eq. 6, the m_j mass function corresponding to the case where $|\frac{\upsilon_j(x_i) - \overline{\upsilon_j}}{\sqrt{\sigma(\upsilon_j)^2 + \epsilon}}| > Zmax$ becomes:

$$m_j(\theta_1) = 0, \ m_j(\theta_2) = 0, \ m_j(\Theta) = 1 \quad (15)$$

which indicates complete ignorance. By this online statistical filtering scheme, the final mass function in Eq. 8 is then affected, and the value for $m_{LR}(\Theta)$ is increased. From this formalism, an evidential multi-task multi-layer perceptron was designed, and trained to classify LIDAR objects as either vehicles, vulnerable road users, or unknown objects.

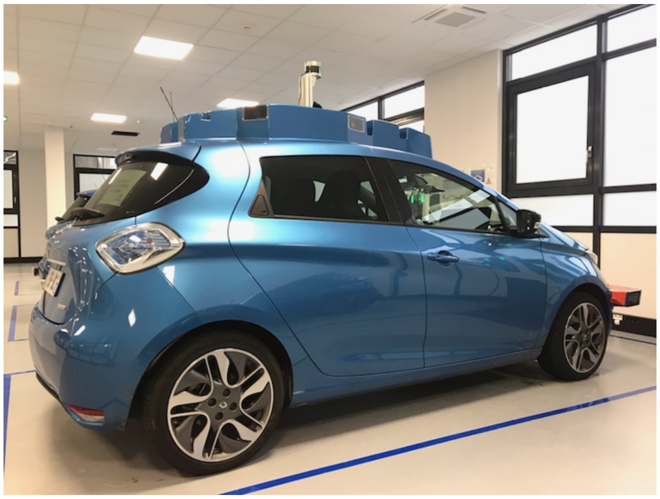

Fig. 2: The data acquisition platform ; the lidar stands on top of the vehicle

V. EVIDENTIAL CLASSIFICATION OF LIDAR OBJECTS

A. *Training dataset*

Although publicly available datasets of labelled LIDAR objects exist, such as the KITTI dataset [27], they do not include any explicitely unknown objects. Moreover, the classification of LIDAR objects is supposed to be performed after a detection step. Then, a coupling with a pre-existing detection system, and a dataset with labelled unknown objects, were needed to test the evidential framework that was previously defined, when applying it to LIDAR object classification. Raw point clouds were thus acquired with an autonomous and perception platform, via a Velodyne VLP-32C sensor. The final dataset is the result of three independent recordings: two that happened on different dates in Guyancourt, France, and one in Toulouse, France. In total, this represents approximately ninety minutes of raw data. The data acquisition platform is depicted in Fig. 2. Then a real-time clustering algorithm was used to detect objects within those LIDAR scans [28]. Detected objects that comprised less than ten points and were further than 45 meters from the vehicle were rejected. Object tracks were then created by associating and tracking the remaining objects over time via a simple Extended Kalman Filter. For each track, a single tracklet was then manually labelled as either ”unknown”, ”car”, ”truck”, ”bike” or ”pedestrian”, and the label was propagated to all the other objects of the track. Tab. I depicts the number of samples for each class in the dataset.

Although the main goal of this work was only to classify vehicles and vulnerable road users while rejecting unknown

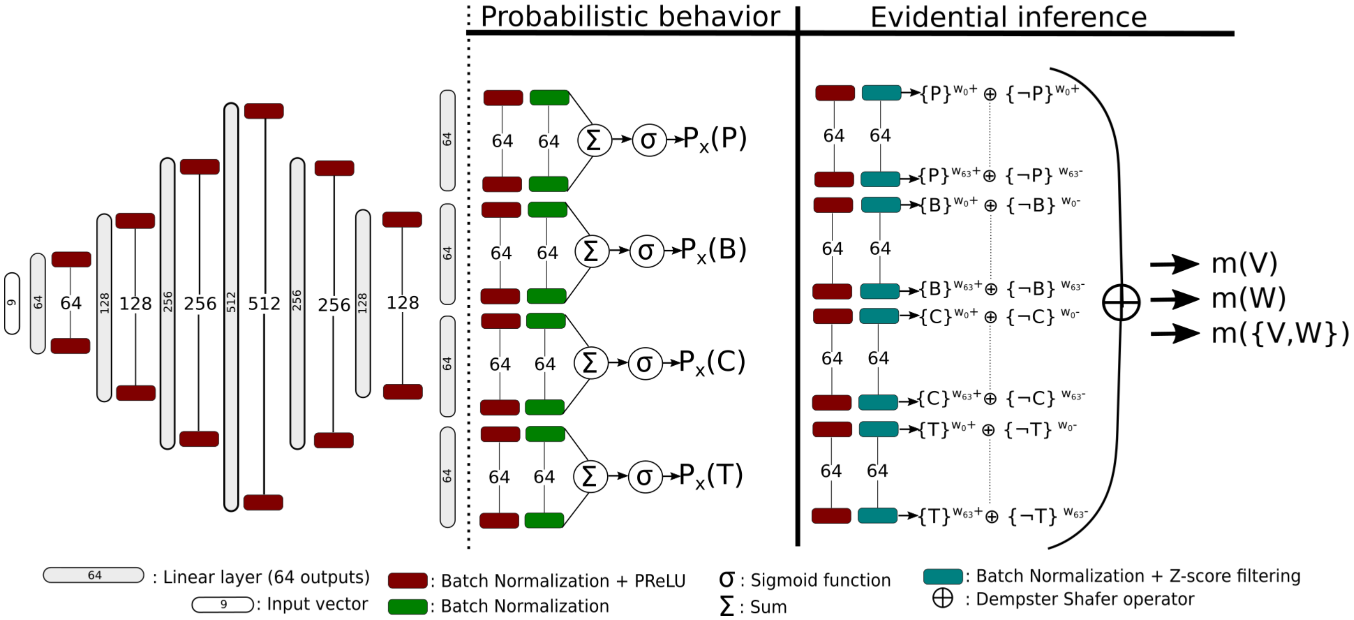
Probabilistic behavior
Evidential inference
9
64
64
128
128
256
256
512
512
256
256
128
128
64
64
64
Σ
σ
$P_x(P)$
$P_x(B)$
$P_x(C)$
$P_x(T)$
$\{P\}^{w_0+} \oplus \{\neg P\}^{w_0+}$
$\{P\}^{w_{63}+} \oplus \{\neg P\}^{w_{63}-}$
$\{B\}^{w_0+} \oplus \{\neg B\}^{w_0-}$
$\{B\}^{w_{63}+} \oplus \{\neg B\}^{w_{63}-}$
$\{C\}^{w_0+} \oplus \{\neg C\}^{w_0-}$
$\{C\}^{w_{63}+} \oplus \{\neg C\}^{w_{63}-}$
$\{T\}^{w_0+} \oplus \{\neg T\}^{w_0-}$
$\{T\}^{w_{63}+} \oplus \{\neg T\}^{w_{63}-}$
$\oplus$
m(V)
m(W)
m({V,W})
64 : Linear layer (64 outputs)
9 : Input vector
: Batch Normalization + PReLU
: Batch Normalization
σ : Sigmoid function
Σ : Sum
: Batch Normalization + Z-score filtering
⊕ : Dempster Shafer operator

Fig. 3: The proposed multi-task architecture

label	number of samples
Car	91297
Truck	9713
Pedestrian	3461
Bike	946
Unknown	10492

TABLE I: Number of LIDAR objects per class in the dataset

objects, extra labels were needed during training. Indeed, trucks, which are way larger than cars, could for instance easily be considered as outliers in a dataset of vehicles. This could be problematic with the statistical filtering scheme presented in the previous section.

A bounding box was fitted to each object by using the Variance Minimisation algorithm in [29], and each object was converted into a vector of nine features:

- Distance between the centroid of the box and the sensor;
- Length, width and height of the fitted bounding box;
- Mean distance between the object points and the centroid of the fitted bounding box, and the corresponding standard deviation;
- The three eigenvalues, computed from a principal component analysis on the Euclidean coordinates of the object points;

B. Model

A multi-task multi-layer perceptron, depicted in Fig. 3, was trained on the dataset of LIDAR objects. The neural networks includes linear layers, PReLU activation layers and batch normalization layers. The PReLU activation was used since it always applies a linear function to its input, though its behavior depends on the sign of the input. The multi-task behavior is needed as the model defined in section IV only corresponds to binary GLR classifiers, while four different classes are present, at least during the training. Then, the MLP has four outputs, corresponding to four binary GLR classifiers. For each object x, the multi-task MLP can then predict four probabilities:

- $P_x(P)$: probability of the object being a pedestrian
- $P_x(B)$: probability of the object being a bike
- $P_x(C)$: probability of the object being a car
- $P_x(T)$: probability of the object being a truck

Let $\neg$ represent logical negation. From Eq. 8, those probabilities can be converted into evidential mass functions:

$$m(P),\ m(\neg P),\ m(\{P,\neg P\}) \tag{16a}$$
$$m(B),\ m(\neg B),\ m(\{B,\neg B\}) \tag{16b}$$
$$m(C),\ m(\neg C),\ m(\{C,\neg C\}) \tag{16c}$$
$$m(T),\ m(\neg T),\ m(\{T,\neg T\}) \tag{16d}$$

Let $\Omega = \{V, W\}$ a frame of discernment, V representing the fact that an object is a vehicle, W representing the fact that an object is a vulnerable road user. This new frame of discernment is justified by the fact that the original goal of the work is only to classify vehicles and vulnerable road users. It is assumed that the fact of not being a car or a truck (resp. a pedestrian or a bike) is not considered as an evidence towards the fact of being a vulnerable road user (resp. a vehicle). The mass functions in Eq. 16 can then be projected into this new frame of discernement as follows:

$$m_p(V) = 0,\ m_p(W) = m(P),\ m_p(\{V,W\}) = 1 - m(P) \tag{17a}$$
$$m_b(V) = 0,\ m_b(W) = m(B),\ m_b(\{V,W\}) = 1 - m(B) \tag{17b}$$
$$m_c(V) = m(C),\ m_c(W) = 0,\ m_c(\{V,W\}) = 1 - m(C) \tag{17c}$$
$$m_t(V) = m(T),\ m_t(W) = 0,\ m_t(\{V,W\}) = 1 - m(T) \tag{17d}$$

Those four mass functions can then be fused via the Dempster-Shafer operator, to get the final masss value m generated from the MLP:

$$m = m_p \oplus m_b \oplus m_c \oplus m_t \tag{18}$$

Algorithm 1 I-D decision rule on $\{V, W\}$

```
if $1 - Bel(V) \leq 1 - Pl(W)$ then
    The object is classified as a vehicle
else if $1 - Bel(W) \leq 1 - Pl(V)$ then
    The object is classified as a vulnerable road user
else
    The object is classified as unknown
end if
```

C. Model training

The multi-task MLP was implemented in PyTorch. The evidential formulation is not used during inference. The training is only done on the object of the "pedestrian", "bike", "car" and "truck" classes, which compose a dataset of pertinent objects noted D_p. The "unknown" objects are only used to create a one class D_u dataset, which will only be used to evaluate the evidential output of the MLP. The parameters of the Batch Normalization layers are estimated during the training. Thus, no Dropout was used, as the statistics in the Batch Normalization layers have to be as accurate as possible to justify the behavior proposed in Eq. 14. Moreover, the training iterations were done on a single batch composed of all the pertinent objects. A training set D_{pt} and a validation set D_{pv} were created from D_p by a 70/30 split. As seen in Tab. I, D_p is very unbalanced. D_{pt} was then refined by randomly sampling objects of the "car" class, and by using the SMOTE algorithm [30] on the "pedestrian" and "bike" classes, to realign the number of samples for each class on the number of "trucks" in D_{pt}. The resulting refined training dataset is noted D'_{pt}.The ADAM optimizer was used with its default parameters, and a learning rate of 0.001. Moreover, following the results in Eq. 12, a weight decay of 1e-5 was used on the linear parameters of the final Batch Normalization layers. The training was done during 400 epochs. Let y_{pi}, y_{bi}, y_{ci}, y_{ti} be binary indicators respectively indicating whether x_i belongs to the class "pedestrian", "bike", "car" or "truck". The loss function is a sum of cross-entropies:

$$\begin{aligned}
-\Bigg[& \sum_{x_i \in D'_{pt}} (y_{pi} log P_{x_i}(P) + (1 - y_{pi}) log(1 - P_{x_i}(P))) \\
+ & \sum_{x_i \in D'_{pt}} (y_{bi} log P_{x_i}(B) + (1 - y_{bi}) log(1 - P_{x_i}(B))) \\
+ & \sum_{x_i \in D'_{pt}} (y_{ci} log P_{x_i}(C) + (1 - y_{ci}) log(1 - P_{x_i}(C)) \\
+ & \sum_{x_i \in D'_{pt}} (y_{ti} log P_{x_i}(T) + (1 - y_{ti}) log(1 - P_{x_i}(T)))) \Bigg]
\end{aligned} \tag{19}$$

Pedestrian or not		Bike or not	
Accuracy	F1-score	Accuracy	F1-score
0.993	0.939	0.996	0.833

Car or not		Truck or not	
Accuracy	F1-score	Accuracy	F1-score
0.983	0.990	0.989	0.943

TABLE II: Probabilistic classification results on D_{pv}

D. Evaluation

1) Probabilistic evaluation: First of all, the proposed multi-task MLP can be evaluated after training on the validation set D_{pv}, only using its initial probabilistic outputs. No Z-score filtering is used in this case, as this would not be meaningful with regards to the *Sigmoid* function S. An object is considered as classified into a class when the corresponding probabilistic output is higher than 0.5 (for e.g., if $P_x(C) > 0.5$, then x is classified as "car", otherwise it is classified as "not car"). In Tab. II, the results for each classification task are given as accurcy scores and F1-scores. The results are satisfactory, as all these indicators are above 0.9 except the F1-score for the bike classification. This can be explained by the significantly lower number of "bikes" compared to the other classes, which justified the use of the SMOTE algorithm. The results for the car and pedestrian classes are still satisfactory, although undersampling and oversampling were used on these classes.

2) Evidential evaluation: The evaluation of the evidential outputs generated from the MLP is done with regards to the $\Omega = \{V, W\}$ frame of discernment, with the mass functions generated from Eqs. 8, 16, 17 and 18. The *interval dominance* (ID) preference relation in [31], and depicted in Algorithm 1, is used to classify objects based on the mass values on V and W. The $ZMax$ value in Eq. 14 is still to be defined. When working with gaussian random variables, the three common thresholds to work with Z-scores are 2.58, 1.96, and 1.65, recpectively corresponding 99%, 95% and 90% confidence levels [26]. The MLP is thus tested with those three possible $Zmax$ values.

The decisions based on the evidential mass functions generated from the MLP are compared with decisions based on its probabilistic outputs, and classification results obtained from a set of one-class SVMs [20]. As said in section II, one-class SVMs are commonly used when trying to detect unknown objects. Moreover, such SVMs can be trained and tested directly on the dataset of LIDAR objects that was created in the context of this work. For a fair comparison with the proposed multi-task MLP, four one-class SVMs are trained on the D_{pt} dataset. Each one of those four SVMs is trained on one class of D_{pt}: "car", "truck", "pedestrian" or "bike". The following classification rule is used to classify objects as vehicles or vulnerable road users from either the probabilistic outputs of the MLP, or the set of four one-class SVMs:

- If an object is classified as a pedestrian or a bike, or as both, and neither as a car nor as a truck, then it is classified as a vulnerable road user (W);

Method	**IoU**	**Accuracy**	**F1-score on V**	**F1-score on W**	**F1-score on Ω**
Ours, probabilistic output with no Z-score filtering	0.312	0.729	0.890	0.408	0.377
Ours, evidential output with no Z-score filtering	0.320	0.733	0.890	0.412	0.388
Ours, evidential output with $ZMax$ = 2.58	0.558	0.825	0.938	0.458	0.675
Ours, evidential output with $ZMax$ = 1.96	0.682	0.872	**0.945**	0.570	0.786
Ours, evidential output with $ZMax$ = 1.65	**0.725**	**0.897**	0.929	**0.661**	**0.825**
One-class SVMs [20]	0.507	0.661	0.672	0.556	0.660

TABLE III: Classification results on D_{rc} ; V stands for ”vehicle”, W stands for ”vulnerable road user”, Ω stands for ”unknown” object

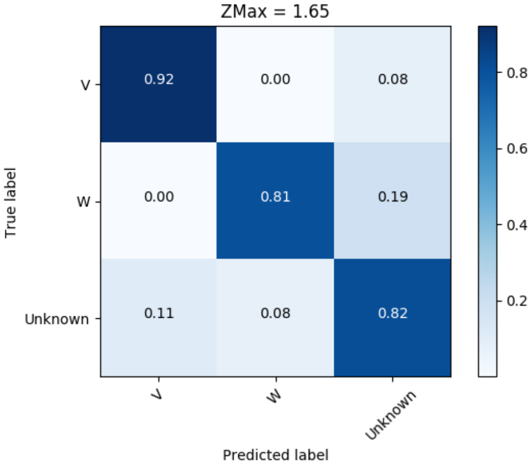

Fig. 4: Normalized confusion matrix for evidential classification with $ZMax$=1.65 on D_{rc}

- Else, if an object is classified as a car or as a truck, or as both, and neither as a pedestrian nor as a bike, then it is classified as a vehicle (V);
- Otherwise, the object is classified as unknown (Ω);

To simulate a test on real-life conditions, the set of SVMs and the MLP with the corresponding classification rules are tested on $D_{rc} = D_u \cup D_{pv}$, the union of the validation dataset and the dataset of unknown objects. The results are presented in Tab. III.

Based on the Intersection Over Union (IoU) scores, the best performing approach is the evidential classification with $ZMax$ equal to 1.65. This version is also the best on practically all the indicators, except the F1-score on V. The interest of Z-score filtering with an evidential formulation of a neural network is visible. Indeed, the worst performing approach is the purely probabilistic one, and the IoU scores increase with the $ZMax$ values. The Z-Score filetering scheme proved to be efficient, as the F1-score for Ω is equal to 0.825 when ZMax is equal to 0.165, although the system was never trained on those unknown objects. Vulnerable road users are still challenging to correctly classify though, as the best F1-score for W is only 0.661. This can again be explained by the fact that the original dataset was highly unbalanced. As seen on Fig. 4, using evidential classification with $ZMax = 1.65$ leads to the desirable feature that, on D_{rc}, all the wrongly classified vehicles and vulnerable road users were classified as unknown objects. Moreover, it is also to be noted that 81% of the vulnerable road users are correctly classified, and that the low F1-score for this class is explained by the 19% that are classified as unknown objects and the 8% of unknown objects that are classified as vulnerable road users. This can also be seen in Tab. IV, which indicates the accuracy and F1-scores only computed on D_{pv}. The IoU score and F1-score for Ω are not reported as these values are not meaningful anymore. In this case, the F1-score for W is equal to 0.890, which is more satisfactory. What’s more, given that the dataset was created in a semi-automatic fashion, it can be assumed that a certain amount of vulnerable road users were wrongly labelled, making it challenging to classify them.

Method	**Accuracy**	**F1-score (V)**	**F1-score (W)**
Ours, $ZMax$ = 1.65	0.914	0.959	0.890

TABLE IV: Results of the evidential classification on D_{pv}

VI. CONCLUSIONS

An evidential classification algorithm of LIDAR objects, represented as feature vectors, was proposed. The algorithm, which was trained as a probabilistic classifier and converted as an evidential classifier afterwards, effectively classifies unknown objects without having been trained on them. Evidential classification is compatible with real-time constraints: on a TitanX Pascal GPU, all the LIDAR objects in D_{rc} (which is composed of 30190 objects) can be classified at once in 0.4s with the current Pytorch implementation. Thus, several refinements are possible, and will be explored in future works. First of all, the input vector representing a LIDAR object could be replaced by an input vector encoded by a PointNet architecture [12], as nothing guarantees that the chosen features are the most appropriate ones to classify unknown LIDAR objects. Yet, this would require to define strategies to cope with the limitations and requirements of PointNet regarding its inputs. Secondly, the proposed Z-score filtering scheme relies on a Gaussian assumption that is probably not completely exact: thus, more refined selection strategies for the $ZMax$ threshold would potentially improve the results. Introducing carefully selected unknown objects in the training set could also help the system to classify pertinent objects (especially vulnerable road users) more effectively. Finally, the definition of strategies to use this system within an autonomous vehicle in a fusion framework and with heuristics (for e.g. ”a moving object is more likely to be pertinent”) is also a direction that has to be explored.

ACKNOWLEDGMENT

This work was realized within the SIVALab joint laboratory between Renault S.A.S, the CNRS and HeuDiaSyc. We thank Thierry Denoeux for his fruitful remarks, and Clement Le Bihan for his help in labelling and processing the dataset that was used in the presented work. We also thank NVidia for the donation of a TitanX Pascal GPU.

REFERENCES

[1] J. S. Albus, "4D/RCS: A reference model architecture for intelligent unmanned ground vehicles," in *Unmanned Ground Vehicle Technology IV*, International Society for Optics and Photonics, vol. 4715, 2002, pp. 303–311.

[2] P. Smets *et al.*, "What is dempster-shafer's model," *Advances in the Dempster-Shafer theory of evidence*, pp. 5–34, 1994.

[3] T. Denoeux, "Logistic regression, neural networks and dempster-shafer theory: A new perspective," *ArXiv preprint arXiv:1807.01846*, 2018.

[4] J. Moras, V. Cherfaoui, and P. Bonnifait, "Moving objects detection by conflict analysis in evidential grids," in *Intelligent Vehicles Symposium (IV)*, IEEE, 2011, pp. 1122–1127.

[5] C. Yu, V. Cherfaoui, and P. Bonnifait, "An evidential sensor model for velodyne scan grids," in *Control Automation Robotics & Vision (ICARCV), 2014 13th International Conference on*, IEEE, 2014, pp. 583–588.

[6] E. Capellier, F. Davoine, V. Frémont, J. Ibañez-Guzman, and Y. Li, "Evidential grid mapping, from asynchronous LIDAR scans and RGB images, for autonomous driving," in *21st International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, pp. 2595–2602.

[7] S. Wirges, C. Stiller, and F. Hartenbach, "Evidential occupancy grid map augmentation using deep learning," in *IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2018, pp. 668–673.

[8] R. O. Chavez-Garcia and O. Aycard, "Multiple sensor fusion and classification for moving object detection and tracking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 525–534, 2016.

[9] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3D lidar point cloud," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 1887–1893.

[10] Y. Wang, T. Shi, P. Yun, L. Tai, and M. Liu, "Pointseg: Real-time semantic segmentation based on 3D lidar point cloud," *ArXiv preprint arXiv:1807.06288*, 2018.

[11] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud," *ArXiv preprint arXiv:1809.08495*, 2018.

[12] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3D classification and segmentation," *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, vol. 1, no. 2, p. 4, 2017.

[13] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3D object detection from rgb-d data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 918–927.

[14] F. Engelmann, T. Kontogianni, A. Hermans, and B. Leibe, "Exploring spatial context for 3D semantic segmentation of point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 716–724.

[15] Y. Yan, Y. Mao, and B. Li, "SECOND: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.

[16] M. Simon, S. Milz, K. Amende, and H.-M. Gross, "Complex-yolo: Real-time 3D object detection on point clouds," *ArXiv preprint arXiv:1803.06199*, 2018.

[17] D. J. MacKay, "A practical bayesian framework for backpropagation networks," *Neural computation*, vol. 4, no. 3, pp. 448–472, 1992.

[18] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *International conference on machine learning*, 2016, pp. 1050–1059.

[19] D. Feng, L. Rosenbaum, and K. Dietmayer, "Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3D vehicle detection," *ArXiv preprint arXiv:1804.05132*, 2018.

[20] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[21] P. Perera and V. M. Patel, "Learning deep features for one-class classification," *ArXiv preprint arXiv:1801.05365*, 2018.

[22] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," in *Advances in Neural Information Processing Systems*, 2018, pp. 3183–3193.

[23] G. Shafer, *A mathematical theory of evidence*. Princeton university press, 1976, vol. 42.

[24] B. R. Cobb and P. P. Shenoy, "On the plausibility transformation method for translating belief function models to probability models," *International Journal of Approximate Reasoning*, vol. 41, no. 3, pp. 314–330, 2006.

[25] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[26] D. J. Rumsey, *U Can: Statistics for dummies*. John Wiley & Sons, 2015.

[27] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[28] Y. Li, C. Le Bihan, T. Ristorcelli, J. Ibanez-Guzman, and K. Bouziane, "Fast coarse-to-fine 3D point cloud segmentation in spherical coordinates for autonomous driving," *Submitted to the International Conference on Robotics and Automation (ICRA)*, 2019.

[29] X. Zhang, W. Xu, C. Dong, and J. M. Dolan, "Efficient l-shape fitting for vehicle detection using laser scanners," in *Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 54–59.

[30] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[31] M. C. Troffaes, "Decision making under uncertainty using imprecise probabilities.," *International journal of approximate reasoning.*, vol. 45, no. 1, pp. 17–29, 2007.