Uncertainty Reduction for 3D Point Cloud Self-Supervised Traversability Estimation

Jihwan Bae¹, Junwon Seo¹, Taekyung Kim¹, Hae-gon Jeon², Kiho Kwak¹ and Inwook Shim³

Abstract—Traversability estimation in off-road environments requires a robust perception system. Recently, approaches to learning a traversability estimation from past vehicle experiences in a self-supervised manner are arising as they can greatly reduce human labeling costs and labeling errors. Nonetheless, the learning setting from self-supervised traversability estimation suffers from congenital uncertainties that appear according to the scarcity of negative information. Negative data are rarely harvested as the system can be severely damaged while logging the data. To mitigate the uncertainty, we introduce a method to incorporate unlabeled data in order to leverage the uncertainty. First, we design a learning architecture that inputs query and support data. Second, unlabeled data are assigned based on the proximity in the metric space. Third, a new metric for uncertainty measures is introduced. We evaluated our approach on our own dataset, ‘Dtrail’, which is composed of a wide variety of negative data.

I. INTRODUCTION

Estimating traversability for vehicles is an important task for autonomous driving and machine perception. However, the majority of the relevant works focus on constrained road environments like paved roads which are all possibly observed in public datasets [1], [2], [3]. In urban scenes, road detection with semantic segmentation is enough [4], [5], but in unconstrained environments like off-road areas, the semantic segmentation is insufficient as the environment can be highly complex and rough [6] (Fig. 1a). Several works from the robotics field have proposed a method to estimate the traversability cost in the unconstrained environments [7], [8], [9], [10], [11], and to infer probabilistic traversability map with visual information such as image [12] and 3D LiDAR [6]. Nonetheless, these works defined the traversability without considering the distinct properties of each vehicle. Traversability by its nature cannot be solely well-defined from visual information in unconstrained environments because it changes by vehicle specifications.

We believe actual physical state changes that a vehicle undergoes can give accurate and precise information on where it can traverse and how difficult it would be. Accordingly, self-supervised traversability estimation can secure more robust autonomous driving. Fig. 1b shows an example of the self-supervised traversability data. Previously, haptic inspection [13], [12] has been examined as traversability

¹Jihwan Bae, Junwon Seo, Taekyung Kim, and Kiho Kwak are with the Agency for Defense Development, Daejeon 34186, Republic of Korea (e-mail: mirojihan1008@gmail.com, junwonvision@gmail.com, tkkim.robot@gmail.com, kkwak.add@gmail.com).
²Hae-gon Jeon is with AI Grad School, GIST, Republic of Korea (e-mail: haejongi@gist.ac.kr)
³Inwook Shim is with Department of Smart Mobility Engineering, Inha University, Republic of Korea (e-mail: iwhsim@inha.ac.kr)
• We introduce a self-supervised traversability estimation in unconstrained environments using 3D point clouds.
• We adopt a deep metric learning-based method that jointly learns the semantic segmentation and the traversability estimation.
• We propose the unsupervised loss to utilize the unlabeled data in the current vehicle-supervised settings.
• We devise a new metric to evaluate the suggested framework properly.
• We present a new dataset for off-road vehicle driving in unconstrained environments.

II. RELATED WORK

A. Traversability Estimation

Traversability estimation is a crucial component in autonomous driving, navigating the vehicle about where it should go. In the case of paved road conditions, the traversability estimation task can be regarded as a subset of road detection [15], [4], [16] and semantic segmentation [17]. This task can be sufficiently handled with a human-supervision approach, and various public datasets (KITTI [1], NuScene [2], and Waymo [3]) are well-organized. However, the human-supervised method is clearly limited in estimating traversability for unconstrained environments like off-road areas. According to the diversity of road conditions, it is impossible to determine the traversability of a vehicle in advance by man-made predefined rules.

Self-supervised approaches [18], [19], [20], [13] are suggested in the robotics literature to estimate the traversability using proprioceptive sensors such as inertial measurement and force-torque sensors. Since these tasks only measured traversability in the proprioceptive-sensor domain, they do not affect the vehicle’s future driving direction. To solve this problem, a study to predict terrain properties by combining image information with vehicle supervision has been proposed [12]. They identify the terrain properties from haptic interaction and associate them with the image to facilitate self-supervised learning. This work demonstrates promising outputs for traversability estimation, but it does not take epistemic uncertainty into account that necessarily exists in the vehicle-supervised data.

To overcome such limitations, we propose self-supervised traversability estimation on 3D point cloud data in unconstrained environments. We carefully alleviate the epistemic uncertainty by splitting explicitly the ‘traversable region’ and ‘non-traversable region’ via learning semantic segmentation along with the traversability estimation.

B. Deep Metric Learning

One of the biggest challenges in learning with few labeled data is epistemic uncertainty. To handle this problem, researchers proposed deep metric learning (DML) [21], which learns embedding spaces and classifies an unseen sample in the learned space. Several works adopt the sampled mini-batches called episodes during training, which mimics the task with few labeled data to facilitate DML [22], [23], [24], [25]. These methods with episodic training strategies epitomize labeled data of each class as a single vector, referred to as a prototype [26], [27], [28], [29], [17]. The prototypes generated by these works require non-parametric procedures and insufficiently represent unlabeled data.

Other works [30], [31], [32], [33], [34], [35], [36] develop loss functions to learn an embedding space where similar examples are attracted, and dissimilar examples are repelled. Recently, proxy-based loss [37] is proposed. Proxies are representative vectors of the training data in the learned embedding spaces, which are obtained in a parametric way [38], [39]. Using proxies leads to better convergence as they reflect the entire distribution of the training data [37]. A majority of the works [38], [39] provide a single proxy for each class, whereas SoftTriple loss [40] adopts multiple proxies for each class. We adopt the SoftTriple loss, as traversable and non-traversable regions are represented as multiple clusters rather than a single one in the unstructured driving scenes according to their complexity and roughness.

III. METHODS

A. Overview

Our vehicle-supervised framework aims to learn a mapping between point clouds to traversability. We call input data containing the traversability as ‘query’. The traversable regions are referred to as the ‘positive’ class, and the non-traversable regions are referred to as the ‘negative’ class in this work. In query data, only positive data are labeled along with their traversability. The rest remains as unlabeled regions. Non-black points in Fig. 2a indicate the positive regions and the black points indicate the unlabeled regions.

However, there exists a limitation in that the query data is devoid of any supervision about negative regions. With query data only, results would be unreliable, as negative regions can be regressed as a good traversable region due to the epistemic uncertainty (Fig. 1f). Consequently, our task aims to learn semantic segmentation along with traversability regression to mask out the negative regions, thereby mitigating the epistemic uncertainty. Accordingly, we utilize a very small number of hand-labeled point cloud scenes and call it ‘support’ data. In support data, traversable regions and non-traversable regions are manually annotated as positive

Fig. 2: Examples of our task data settings: query and support data. (a) is an example of query data. Unlabeled points are colored black, and non-black points indicate the traversable region. Traversability is mapped only on the positive points. (b) is an example of support data. Traversable, non-traversable are manually labeled as red, blue, respectively. Note that only evident regions are labeled and used for the training in the support data.
and negative, respectively. Manually labeling entire scenes can be biased with human intuitions. Therefore, only evident regions are labeled and used for training. Fig. 2B shows the example of labeled support data. The overall schema of our task is illustrated in Fig. 3. When the input point cloud data is given, a segmentation mask is applied to the initial version of the traversability regression map, producing a masked traversability map as a final output.

For training, we form an episode composed of query and randomly sampled support data. We can optimize our network over both query and relatively small support data with the episodic strategy. Also, to properly evaluate the proposed framework, we introduce a new metric that comprehensively measures the segmentation and the regression, while highlighting the nature of the traversability estimation task with the epistemic uncertainty.

B. Fully-supervised learning method

Let query data, consisting of positive and unlabeled data, as \( Q = \{Q_P, Q_U\} \), and support data, consisting of positive and negative data, as \( S = \{S_P, S_N\} \). Let \( P_i \in \mathbb{R}^3 \) denotes the 3D point, \( a_i \in \mathbb{R} \) denotes the traversability, and \( y_i \in \{0, 1\} \) denotes the class of each point. Accordingly, data from \( Q_P, Q_U, S_P, \) and \( S_N \) are in forms of \( \{P_i, a_i, y_i\} \), \( \{P_i\} \), \( \{P_i, y_i\} \), and \( \{P_i, y_i\} \), respectively. Let \( f_\theta \) denote a feature encoding backbone where \( \theta \) indicates a network parameter, \( x_i \in \mathbb{R}^d \) as encoded features extracted from \( P_i \), and \( h_\theta \) as the multi-layer perceptron (MLP) head for the traversability regression. \( g_\theta \) denotes the MLP head for the segmentation that distinguishes the traversable and non-traversable regions. The encoded feature domain for each data is notated as \( Q_P, Q_U, S_P, \) and \( S_N \).

A fully-supervised solution learns the network with labeled data only. \( Q_P \) is used for the traversability regression and \( Q_P \) and \( S \) are both used for the segmentation. We obtain the traversability map \( t_i = h_\theta(x_i), t_i \in \mathbb{R} \), and segmentation map \( s_i = g_\theta(x_i), s_i \in \{0, 1\} \). The final masked traversability map \( T_i \) is represented as \( T_i = t_i \odot s_i \). The regression loss \( L_{reg} \) is computed with \( Q_P \) and based on a mean squared error loss as Eq. (1), where \( x_i \) is the \( i \)-th element of \( Q_P \).

\[
L_{reg}(x_i) = (h(x_i) - a_i)^2. \tag{1}
\]

For the segmentation loss \( L_{seg} \), binary cross-entropy loss is used in the supervised setting as Eq. (2), where \( x_i \) refers to the \( i \)-th element of either \( Q_P \) and \( S \). Both the positive query and the support data can be used for the segmentation loss as follows:

\[
L_{seg}(x_i) = - \left( y_i \log(g(x_i)) + (1 - y_i) \log(1 - g(x_i)) \right). \tag{2}
\]

Combining the regression and the segmentation, the traversability estimation loss in the supervised setting is defined as follows:

\[
L_{supervised}(Q_P, S) = \frac{1}{|Q_P|} \sum_{x_i \in Q_P} \left( L_{reg}(x_i) + L_{seg}(x_i) \right) + \frac{1}{|S|} \sum_{x_i \in S} L_{seg}(x_i). \tag{3}
\]

Nonetheless, it does not fully take advantage of data captured under various driving scenes. Since the learning is limited to the labeled data, it can not capture the whole characteristics of the training data. This drawback limits the capability of the traversability estimation trained in a supervised manner.
Fig. 4: Illustration of the effect of adopting the unlabeled data. Red and blue nodes are embedding vectors of positive and negative data. Gray nodes with a question mark indicate the unlabeled data, and the black ones indicate proxies. The background color and lines indicate decision boundaries in the embedding space. The embedded vectors (non-black nodes) assigned to the proxies are connected to each other with solid lines. (a) Without unlabeled data, proxies and decision boundaries are optimized only with labeled data. (b) With unlabeled data, the optimization exploits the broader context of the training data, resulting in a more precise and discriminative decision boundary.

C. Metric learning method

We adopt metric learning to overcome the limitation of the fully-supervised solution. The objectives of metric learning are to learn embedding space and find the representations that epitomize the training data in the learned embedding space. To jointly optimize the embedding network and the representations, we adopt a proxy-based loss. The embedding network is updated based on the positions of the proxies, and the proxies are adjusted by the updated embedding network iteratively. The proxies can be regarded as representations that abstract the training data. We refer this set of proxies as ‘proxy bank’, denoted as $\mathcal{B} = \{\mathcal{B}_p, \mathcal{B}_N\}$, where $\mathcal{B}_p$ and $\mathcal{B}_N$ indicate the set of proxies for each class. The segmentation map is inferred based on the similarity between feature vectors and the proxies of each class, as $s_i = g(\mathcal{B}, x_i)$.

The representations of traversable and non-traversable regions exhibit large intra-class variations, where numerous sub-classes exist in each class; flat ground or gravel road for positive, and rocks, trees, or bushes for negative. For the segmentation, we use SoftTriple loss [40] that utilizes multiple proxies for each class. The similarity between $x_i$ and class $c$, denoted as $S_{i,c}$, is defined by a weighted sum of cosine similarity between $x_i$ and $\mathcal{B}_c = \{p_{c,1}, ..., p_{c,K}\}$, where $c$ denotes positive or negative, $K$ is the number of proxies per class, and $p_{c,k}$ is $k$-th proxy in the proxy bank. The weight given to each cosine similarity is proportionate to its value. $S_{i,c}$ is defined as follows:

$$S_{i,c} = \sum_k \frac{\exp(\frac{1}{T} x_i^T p_{c,k})}{\sum_i \exp(\frac{1}{T} x_i^T p_{c,k})} x_i^T p_{c,k},$$

where $T$ is a temperature parameter to control the softness of assignments. Soft assignments reduce sensitivity between multiple centers. Note that the $l_2$ norm has been applied to embedding vectors to sustain divergence of magnitude. Then the SoftTriple loss is defined as follows:

$$L_{SoftTriple}(x_i) = -\log \frac{\exp(\lambda(S_{i,y_i} - \delta))}{\exp(\lambda(S_{i,y_i} - \delta)) + \sum_{y \neq y_i} \exp(\lambda S_{i,y})},$$

where $\lambda$ is a hyperparameter for smoothing effect and $\delta$ is a margin. The segmentation loss using the proxy bank can be reformulated using the SoftTriple loss as Eq. (6) and the traversability estimation loss using the proxy bank is defined as Eq. (7).

$$L_{seg}(x_i, \mathcal{B}) = -\log \frac{\exp(\lambda(S_{i,y_i} - \delta))}{\exp(\lambda(S_{i,y_i} - \delta)) + \exp(\lambda S_{i,1-y_i})},$$

$$L_{proxy}(Q_p, \mathcal{S}, \mathcal{B}) = \frac{1}{|Q_p|} \sum_{x_i \in Q_p} \left( L_{seg}(x_i) + L_{seg}(x_i, \mathcal{B}) \right) + \frac{1}{|\mathcal{S}|} \sum_{x_j \in \mathcal{S}} L_{seg}(x_j, \mathcal{B}).$$

Unlabeled data, which is abundantly included in vehicle-supervised traversability data, has not been considered in previous works. To enhance the supervision we can extract from the data, we utilize the unlabeled data in the query data in the learning process. The problem is that the segmentation loss cannot be applied to the $\mathcal{Q}_U$ because no class label $y_i$ exists for them. We assign an auxiliary target for each unlabeled data as clustering [41]. Pseudo class of $i$-th sample $\hat{y}_i$ is assigned based on the class of the nearest proxy in the embedding space as $\hat{y}_i = \arg\max_{c \in \{\mathcal{P}_B, \mathcal{P}_N\}} S_{i,c}$.

The unsupervised loss for the segmentation, denoted as $L_U$, is defined as Eq. (8) using the pseudo-class, where $x_i$ is an embedding of $i$-th sample in $Q_U$.

$$L_U(x_i, \mathcal{B}) = -\log \frac{\exp(\lambda(S_{i,\hat{y}_i} - \delta))}{\exp(\lambda(S_{i,\hat{y}_i} - \delta)) + \exp(\lambda S_{i,1-\hat{y}_i})},$$

Fig. 4 illustrates the benefit of incorporating unlabeled loss. The embedding network can learn to capture more broad distribution of data, and learned proxies would represent training data better. When unlabeled data features are assigned to the proxies (Fig. 4b), the embedding space and proxies are updated as Fig. 4b, exhibiting more precise decision boundaries.

Combining the aforementioned objectives altogether, we define our final objective as ‘Traverse Loss’, and is defined as Eq. (9). The overall high-level schema of the learning procedure is depicted in Fig. 5.

$$L_{traverse}(Q, \mathcal{S}, \mathcal{B}) = L_{proxy}(Q_p, \mathcal{S}, \mathcal{B}) + \frac{1}{|Q_U|} \sum_{x_i \in Q_p} L_U(x_i, \mathcal{B})$$

D. Re-initialization to avoid trivial solutions

Our metric learning method can suffer from sub-optimal solutions, which are induced by empty proxies. Empty proxies indicate the proxies to which none of the data are assigned. Such empty proxies should be re-deployed to be a
good representation of training data. Otherwise, the model might lose the discriminative power and the bank might include semantically poor representations.

Our intuitive idea to circumvent an empty proxy is to re-initialize the empty proxy with support data features. By updating the empty proxies with support data, the proxy bank can reflect training data that was not effectively captured beforehand. In order to obtain representative feature vectors without noises, $M$ number of prototype feature vectors, denoted as $\mu^+ = \{\mu_m^+, m = 1, ..., M\}$ and $\mu^- = \{\mu_m^-, m = 1, ..., M\}$, are estimated using an Expectation-Maximization algorithm [42]. The prototype vectors are cluster centers of support features. We randomly choose the prototype vectors with small perturbations and use them as re-initialized proxies. Algorithm 1 summarizes the overall training procedure of our method.

### E. Traversability Precision Error

We devise a new metric for the proposed framework, ‘Traversability Precision Error’ (TPE). The new metric should be able to comprehensively evaluate the segmentation and the regression while taking the critical aspect of the traversability estimation into account. One of the most important aspects of traversability estimation is to avoid the false-positive of the traversable region, the region that is impossible to drive but inferred as traversable. If such a region is estimated as traversable, a vehicle will likely go over that region, resulting in undesirable movements. The impact of the false-positive decreases if they are estimated as less traversable. TPE computes the degree of false-positive of the traversable region, extenuating its impact with the traversability, $t_i$. The TPE is defined as Eq. (10) where $TN$, $FP$, and $FN$ denote the number of true negative, false positive, and false negative points of the traversable region, respectively.

\[
\text{Traversability Precision Error (TPE)} = \frac{TN}{TN + FP(1 - t_i) + FN} \tag{10}
\]

---

**Algorithm 1:** Single epoch of traversability estimation with metric learning

**Input:** Query data $Q = \{Q_P, Q_U\}$ and Support Data $S = \{S_P, S_N\}$, where $|Q| \gg |S|$

**Output:** Network $f$ with parameters $\theta$, proxy bank $B = \{B_P, B_N\}$

**for each query data do**

- **Random Sample** support data from $S$
- **Feed** query and support data to $f_0$, and **Get** embedding features $x_i$
- **Calculate** similarity between $x_i$ and $B$
- **Estimate** Pseudo-class $\hat{y}_i$ for $x_i \in Q_U$
- **Calculate** $L^{\text{traverse}}$
- **Update** $\theta$ and $B$

**end**

**Calculate** the membership of each proxy

**if an empty proxy exists then**

- **Feed** $S$ to $f_0$, and **Get** embedding features
- **Estimate** $M$ cluster centers for each class, $\mu = \{\mu^+, \mu^-\}$ by the EM algorithm
- **Re-initialize** empty proxy to $\mu$ with small random perturbation

**end**

### IV. Experiments

In this section, our method is evaluated with Dtrail dataset for traversability estimation on off-road environments along with SemanticKITTI [14] dataset. Our method is compared to other metric learning methods based on episodic training strategies. Furthermore, we conduct various ablation studies to show the benefits of our method.

**A. Datasets**

**Dtrail:** Off-road driving dataset: In order to thoroughly examine the validity of our method, we build the Dtrail
### Table I: Comparison results on Dtrail and SemanticKITTI dataset.

Our methods with different objectives are annotated as follows.

- **Ours (supervised):** Eq. (3) that is trained in supervised manner.
- **Ours (w.o. unlabeled):** Eq. (7) that does not leverage unlabeled data.
- **Ours (w.o. re-init):** Eq. (9) excluding the re-initialization step.
- **Ours:** Eq. (9).

|          | Dtrail |          |  | SemanticKITTI |          |  |
|----------|--------|----------|---|---------------|----------|---|
|          | mIoU   | TPE      | mIoU |               |          |  |
| [\(S]\)/[Q] | 4%      | 1%      | 2%  | 1%      | 4%      | 1%  | 2%  | 1%  | 5%  | 1%  | 0.5% | 0.1% |
| ProtoNet [23] | 0.8033  | 0.7515  | 0.5049 | 0.7129  | 0.5624  | 0.3249 | 0.8809 | 0.8040 | 0.7993 | 0.7798 |
| MPTI [17]     | 0.6992  | 0.6936  | 0.6390 | 0.6202  | 0.5466  | 0.4995 | 0.8586 | 0.8108 | 0.7531 | 0.7663 |
| Ours (supervised) | 0.9238  | 0.8857  | 0.7779 | 0.8896  | 0.8447  | 0.7345 | 0.8405 | 0.8376 | 0.8338 | 0.8201 |
| Ours (w.o. unlabeled) | 0.8864  | 0.8529  | 0.8461 | 0.8434  | 0.8121  | 0.8164 | 0.8124 | 0.7896 | 0.8049 | 0.7994 |
| Ours (w.o. re-init) | 0.8970  | 0.8771  | 0.7935 | 0.8649  | 0.8163  | 0.7517 | 0.8058 | 0.7895 | 0.8058 | 0.7895 |
| Ours          | 0.9338  | 0.9151  | 0.9005 | 0.9067  | 0.8776  | 0.8636 | 0.8652 | 0.8402 | 0.8473 | 0.8973 |

**Fig. 6:** Dtrail dataset. (a) Our vehicle with one 32-layer and two 16 layers of LiDAR sensors. (b) Images of the mountain trail scenes where we construct the dataset.

**Fig. 7:** Qualitative Results for Dtrail dataset. (a) Camera image of each scene. (b)-(d) Ground truth and inference results of segmentation. A red point indicates a traversable region, a blue one indicates a non-traversable region, and a black point is an unlabeled region. (e) The final output of our traversability estimation. The traversability map of non-traversable regions is masked out using the segmentation result.

**SemanticKITTI:** We evaluate our method on the SemanticKITTI [14] dataset, which is an urban outdoor-scene dataset for point cloud segmentation. Since it does not provide any type of attributes for traversability, we conducted experiments on segmentation only. It contains 11 sequences, 00 to 10 as the training set, with 23,210 point clouds and 28 classes. We split 5 sequences (00, 02, 05, 08, 09) with 17,625 point clouds for training and the rest, with 5,576 point clouds, for evaluation. We define ‘road’, ‘parking’, ‘sidewalk’, ‘other-ground’, and ‘terrain’ classes as positive and the rest classes as negative. For query data, only ‘road’ class is labeled as positive and left other positive classes as unlabeled. We expect the model to learn the other positive regions using unlabeled data without direct supervision.

**B. Evaluation metric**

We evaluate the performance of our method with TPE (Section III-E), the new criteria designed for the traversability estimation task, which evaluate segmentation and regression quality simultaneously. Additionally, we evaluate the segmentation quality with mean Intersection over Union [43] (mIoU). For each class, the IoU is calculated by $\text{IoU} = \frac{TP}{TP+FP+FN}$, where $TP$, $FP$, and $FN$ respectively denote the number of true negative, false positive, and false negative points of each class.
C. Implementation Details

1) Embedding network: RandLA-Net [5] is fixed as a backbone embedding network for every method for a fair comparison. Specifically, we use 2 down-sampling layers in the backbone and excluded global \((x, y, z)\) positions in the local spatial encoding layer, which aids the network to embed local geometric patterns explicitly. The embedding vectors are normalized with \(l_2\) norm and are handled with cosine similarity.

2) Training: We train the model and proxies with Adam optimizer with the exponential learning rate decay for 50 epochs. The initial learning rate is set as \(1e^{-4}\). For query and support data, K-nearest neighbors (KNN) of a randomly picked point is sampled in training steps. We ensure that positive and negative points exist evenly in sampled points of the support data.

3) Hyperparameter setting: For learning stability, proxies are updated exclusively for the initial 5 epochs. The number of proxies \(K\) is set to 128 for each class and the proxies are initialized with normal distribution. We set small margin \(\delta\) as 0.01, \(\lambda\) as 20, and temperature parameter \(T\) as 0.05 for handling multiple proxies.

D. Results

Comparison: We compare the performance to ProtoNet [23] that uses a single prototype and MPTI [17] that adopts multiple prototypes for few shot 3D segmentation. Also, we compare the performance with our supervised manner method in Section III-B, denoted as ‘Ours(supervised)’. Table I summarizes the result of experiments. Our method shows a significant margin in terms of IoU and TPE compared to the ProtoNet and MPTI. It demonstrates that generating prototypes in a non-parametric approach does not represent the whole data effectively. Moreover, it is notable that we show the performance of our metric learning method is better than the supervised setting designed for our task. It verifies that ours can reduce epistemic uncertainty by incorporating unlabeled data by unsupervised loss. For SemanticKITTI, the observation is similar to that of the Dtrail dataset. Even though the SemanticKITTI dataset is based on the urban scenes, our method shows better performance than other few-shot learning methods by 6% and the supervised manner by 2%.

1) Ablation studies: We repeat experiments with varying support-to-query ratio \((|S|/|Q|)\) to evaluate robustness regarding the amount of support data. Table I shows that our metric learning method is much more robust from performance degradation than the others when the support-to-query ratio decreases. When the ratio decreases from 4% to 1% in the Dtrail dataset, TPE of our metric learning method only decreases about 4% while TPE of others dropped significantly: 39% for ProtoNet, 13% for MPTI, and 16% for Ours(supervised). It verifies that our method can robustly reduce epistemic uncertainty with small labeled data.

Moreover, we observe that performance increases by 6% in average on TPE when adopting the re-initialization step. It confirms the re-initialization step can help avoid trivial solutions. Also, it is shown that adopting the unsupervised loss can boost the performance up to 6% in average. It verifies that the unlabeled loss can give affluent supervision without explicit labels. Moreover, as shown in Table I, an increasing number of proxies boost the performance until it converges when the number exceeds 32, which demonstrates the advantages of multiple proxies.

2) Qualitative Results: Fig. 7 shows the traversability estimation results of our supervised-based and metric learning-based method on the Dtrail dataset. We can examine that our metric learning-based method perform better than the supervised-based method. Especially, our method yields better results on regions that are not labeled on training data. We compare the example of segmentation results with the SemanticKITTI dataset in Fig. 8. The first column indicates the ground truth and other columns indicate segmentation results of the supervised learning-based method and our method. Evidently, our method shows better results on unlabeled regions, which confirms that our metric learning-based method reduces epistemic uncertainty.

Fig. 9 shows the visualization of the proxies assigned to the point cloud scenes. For better visualization, proxies are clustered to three representations. We observe that the learned proxies successfully represent the various semantic features. Leaves, grounds, and tree trunks are mostly colored as green, black, and blue, respectively.
TABLE II: Ablation study on Dtrail according to the number of proxies $K$.

| $K$  | 1   | 2   | 4   | 8   | 16  | 32  | 64  | 128 | 256 | 512 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| mIoU(%) | 0.890 | 0.894 | 0.880 | 0.906 | 0.911 | 0.883 | **0.920** | **0.934** | **0.931** | **0.924** |
| mTE(%)  | 0.847 | 0.868 | 0.840 | 0.862 | 0.881 | 0.845 | **0.888** | **0.906** | **0.898** | **0.895** |

V. Conclusion

We propose a self-supervised traversability estimation framework on 3D point cloud data in terms of mitigating epistemic uncertainty. Self-supervised traversability estimation suffers from the uncertainty that arises from the limited supervision given to the data. We tackle the epistemic uncertainty by concurrently learning semantic segmentation along with traversability estimation, eventually masking out the non-traversable regions. We start from the fully-supervised setting and finally developed the deep metric learning method with unsupervised loss that harnessed the unlabeled data. To properly evaluate the framework, we also devise the new metric according to the task’s settings and underline the important criteria of the traversability estimation. We build our own off-road vehicle driving dataset in unconstrained environments for realistic testing. Various experimental results show that our framework is promising.

References

[1] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The kitti dataset,” in IJRR, 2013.
[2] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, “nuscenes: A multimodal dataset for autonomous driving,” in CVPR, 2020.
[3] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patain, P. Tuoi, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, “Scalability in perception for autonomous driving: Waymo open dataset,” in CVPR, 2020.
[4] Z. Chen, J. Zhang, and D. Tao, “Progressive lidar adaptation for road detection,” in IEEE/CAA Journal of Automatica Sinica, 2019.
[5] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, “Randla-net: Efficient semantic segmentation of large-scale point clouds,” in CVPR, 2020.
[6] J. Sock, J. Kim, J. Min, and K. Kwak, “Probabilistic traversability map generation using 3d-lidar and camera,” in ICRA, 2016.
[7] H. Roncancio, M. Becker, A. Broggi, and S. Cattani, “Traversability analysis using terrain mapping and online-trained terrain type classifier,” in IEEE Intelligent Vehicles Symposium Proceedings, 2014.
[8] J. Ahtianen, T. Stoyanov, and J. Saarinen, “Normal distributions transform traversability maps: Lidar-only approach for traversability mapping in outdoor environments,” in IFRJ, 2017.
[9] P. Arena, C. F. Blanco, A. Li Noce, S. Taffara, and L. Patanè, “Learning traversability map of different robotic platforms for unstructured terrains path planning,” in IJCNN, 2020.
[10] S. Matsuzaki, J. Miura, and H. Masuzawa, “Semantic-aware plant traversability estimation in plant-rich environments for agricultural mobile robots,” 2021.
[11] T. Guan, Z. He, D. Manocha, and L. Zhang, “Tmn: Terrain traversability mapping for autonomous evacuator navigation in unstructured environments,” 2021.
[12] L. Wellhausen, A. Dosovitskiy, R. Ranftl, K. Walas, C. Cadena, and M. Hutter, “Where should i walk? predicting terrain properties from images via self-supervised learning,” inRAL, 2019.
[13] C. Kolvenbach, Hendrik a nd Bärschi, L. Wellhausen, R. Grandia, and M. Hutter, “Haptic inspection of planetary soils with legged robots,” in RAL, 2019.
[14] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, “SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences,” in ICCV, 2019.
[15] H. Wang, R. Fan, P. Cai, and M. Liu, “SNE-RoadSeg+: Rethinking depth-normal translation and deep supervision for freespace detection,” in IROS, 2021.
[16] A. Valada, J. Vertens, A. Dhall, and W. Burgard, “Adapnet: Adaptive semantic segmentation in adverse environmental conditions,” in ICRA, 2017.
[17] N. Zhao, T-S. Chua, and G. H. Lee, “Few-shot 3d point cloud semantic segmentation,” in CVPR, 2021.
[18] P. Dallaire, K. Walas, P. Giguere, and B. Chaib-draa, “Learning terrain types with the pitman-yor process mixtures of gaussians for a legged robot,” in IROS, 2015.
[19] D. Deng, H. Gao, Z. Deng, J. Song, Y. Liu, G. Liu, and K. Iagnemma, “Foot-terrain interaction mechanics for legged robots: Modeling and experimental validation,” in IJRR, 2015.
[20] W. Bosworth, J. Whitney, S. Kim, and N. Hogan, “Robot locomotion on hard and soft ground: Measuring stability and ground properties in-situ,” in ICRA, 2016.
[21] G. R. Koch, “Siamese neural networks for one-shot image recognition,” 2015.
[22] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, “Learning to compare: Relation network for few-shot learning,” in CVPR, 2018.
[23] J. Snell, K. Swersky, and R. Zemel, “Prototypical networks for few-shot learning,” in NeurIPS, 2017.
[24] V. G. Satorras and J. B. Estrach, “Few-shot learning with graph neural networks,” in ICLR, 2018.
[25] O. Vinyals, C. Blundell, T. Lillicrap, k. kavukcuoglu, and D. Wierstra, “Matching networks for one shot learning,” in NeurIPS, 2016.
[26] C. Chen, O. Li, A. Barnett, J. Su, and C. Rudin, “This looks like that: deep learning for interpretable image recognition,” in NeurIPS, 2019.
[27] J. Deuschel, D. Firmbach, C. Geppert, M. Eckstein, A. Hartmann, V. Bruns, P. Kurtyczyn, J. Desi, D. Hartmann, D. Perrin, T. Wittenberg, and M. Benz, “Multi-prototype few-shot learning in histopathology,” in ICCVW, 2021.
[28] K. Allen, E. Shellerhain, H. Shin, and J. Tenenbaum, “Infinite mixture prototypes for few-shot learning,” in ICML, 2019.
[29] B. Yang, C. Liu, B. L. J. Jiao, and Y. Qixiang, “Prototype mixture models for few-shot semantic segmentation,” in ECCV, 2020.
[30] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in CVPR, 2015.
[31] M. Wang and W. Deng, “Deep face recognition: A survey,” in Neurocomputing, 2021.
[32] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in CVPR, 2005.
[33] R. Hadsell, S. Chopra, and Y. LeCun, “Dimensionality reduction by learning an invariant mapping,” in CVPR, 2006.
[34] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in CVPR, 2015.
[35] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, “Learning fine-grained image similarity with deep ranking,” in CVPR, 2014.
[36] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese, “Deep metric learning via lifted structured feature embedding,” in CVPR, 2016.
[37] Y. Movshovitz-Attias, A. Toshew, T. Leung, S. Ioffe, and S. Singh, “No fuss distance metric learning using proxies,” in ICCV, 2017.
[38] E. W. Teh, T. Devries, and G. W. Taylor, “Proxynca: Revisiting and revitalizing proxy neighborhood component analysis,” in ECCV, 2020.
[39] S. Kim, D. Kim, M. Cho, and S. Kwak, “Proxy anchor loss for deep metric learning,” in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
[40] Q. Qian, L. Shang, B. Sun, J. Hu, H. Li, and R. Jin, “Softtriple loss: Deep metric learning without triplet sampling,” in ICCV, 2019.
[41] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, “Deep clustering for unsupervised learning of visual features,” in ECCV, 2018.
[42] T. Moon, “The expectation-maximization algorithm,” in IEEE Signal Processing Magazine, 1996.
[43] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” in IJCV, 2010.