Hyperbolic Uncertainty Aware Semantic Segmentation

Bike Chen, Wei Peng, Xiaofeng Cao, Member, IEEE, and Juha Röning

Abstract—Semantic segmentation (SS) aims to classify each pixel into one of the pre-defined classes. This task plays an important role in self-driving cars and autonomous drones. In SS, many works have shown that most misclassified pixels are commonly near object boundaries with high uncertainties. However, existing SS loss functions are not tailored to handle these uncertain pixels during training, as these pixels are usually treated equally as confidently classified pixels and cannot be embedded with arbitrary low distortion in Euclidean space, thereby degenerating the performance of SS. To overcome this problem, this paper designs a Hyperbolic Uncertainty Loss (HyperUL), which dynamically highlights the misclassified and high-uncertainty pixels in Hyperbolic space during training via the hyperbolic distances. The proposed HyperUL is model agnostic and can be easily applied to various neural architectures. After employing HyperUL to three recent SS models, the experimental results on Cityscapes, UAVid, and ACDC datasets reveal that the segmentation performance of existing SS models can be consistently improved. Additionally, reliable measurement of model uncertainty plays a key role in real-world applications such as autonomous controls of vehicles and drones. To meet this requirement, we propose the Hyperbolic Uncertainty Estimation method, which is easily implemented by only post-processing the generated Hyperbolic embeddings. By this approach, we can calculate the uncertainty values almost for free. Quantitative and qualitative results on Cityscapes, UAVid, and ACDC datasets verify that our proposed uncertainty estimation method usually outputs more meaningful results compared with popular MC-dropout and ensembling methods.

Index Terms—Hyperbolic space, hyperbolic uncertainty estimation, semantic segmentation, self-driving cars, autonomous drones.

I. INTRODUCTION

Semantic segmentation (SS) separates an image into different meaningful and coherent parts to identify distinct objects. It can serve as a powerful and practical tool for the downstream image analysis [1], [2] such as scene categorization, and free space detection. Therefore SS plays a key role in the intelligent transportation systems [3], [4]. Apparently, high segmentation performance in autonomous vehicles or autonomous drones will help comprehensively understand the surrounding environment, hence greatly improving the safety [5].

Although most existing SS models have achieved impressive segmentation performance, they did not utilize uncertainty information during training. Uncertainty indicates whether a pixel in an image will be classified confidently or not. Improving uncertain predictions can improve segmentation performance, as uncertain predictions are strongly related to misclassifications of pixels. As depicted in Fig. 1, uncertain pixels contain a large number of misclassified pixels, i.e., green pixels. It means that improving uncertain predictions also corrects wrong predictions. There are some works exploring this direction. For example, the works [7], [8], and [9] used Monte Carlo Dropout and Deep Ensembles to estimate uncertainties and improve segmentation outputs. These models commonly make accurate uncertainty estimation, but they have large computational overhead. The works [10], [11], [12] incorporated explicit modules into single deterministic networks to predict uncertainty. However, the measurement of the predictive uncertainty is not sufficient for safety-critical decisions. Also all of the above approaches do not
consider utilizing uncertainty information during training. These works [13], [14] took uncertainty during the training stage. However, [13] only considered the uncertainty of the class by assigning different weights to different classes, not to each pixel. Reference [14] intended to reduce the impact of uncertainty by giving each uncertain pixel a fixed small weight. In this paper, we argue that weighting each uncertain pixel dynamically during training might improve segmentation performance.

Uncertain pixels are commonly near object boundaries [14], [15]. However, existing SS models have not the natural ability to distinguish these pixels only based on their Euclidean embeddings, as hierarchical structures [6], [16] cannot be embedded with arbitrary low distortion in Euclidean space even with an unbounded number of dimensions [17]. We argue that segmentation datasets exhibit latent hierarchical structures, and thus better segmentation performance could benefit from Hyperbolic embeddings in Hyperbolic space. The recent work [15] has verified the existence of hierarchical structures in segmentation data. Additionally, we here provide an observation and comprehensive quantification analysis to further validate this. To achieve better segmentation performance, containing rich contextual information for each pixel embedding is extremely important. The contextual information is commonly aggregated by modeling the long-range dependencies among pixels. Therefore, a pixel embedding, especially for a pixel on the boundary, might be contributed by other nearby pixels from totally different classes. This means that a boundary pixel commonly contains representative features of multiple classes. By contrast, the interior pixel usually includes specific features of only one class. Therefore, embedding an image is similar to embedding a hierarchy (i.e., relationships between boundary pixels and interior pixels). In addition, we further verify that there exist hierarchical structures in segmentation data by $\delta$-hyperbolicity [6]. The $\delta$-hyperbolicity can be utilized to measure the data structure similarity between Euclidean space and Hyperbolic space. The $\delta$-hyperbolicity is in $[0, 1]$. When the calculated $\delta$-hyperbolicity on a data set is closer to 0, the degree of hyperbolicity of the data is quite high. This means that there exists strong hierarchy in the data. In contrast, the computed $\delta$-hyperbolicity closer to 1 indicates that there is no hierarchy in the dataset. Taking Cityscapes [1], UAVid [18], Mapillary Vistas [19], BDD [20], and ACDC [21] for examples, in Table I, we see that these datasets are with strong hierarchical structures, since their $\delta$-hyperbolicity values are closer to 0.

In addition, the ability of estimating epistemic uncertainty of segmentation models is also important for autonomous vehicles or drones [29], [30], [31]. For example, under the adverse weather conditions and environments, segmentation models commonly fail to segment challenging but vital regions. In this scenario, we expect that segmentation models have the ability to output their uncertainty estimation values used for the decisions of autonomous vehicles or drones. Uncertainty estimation hence helps raise the safety in these real-world applications. MC-dropout [7], [29], [31], [32] and ensembling [8], [33], [34] are popular approaches in the uncertainty estimation task, because they are easily tractable and applicable. However, these two kinds of methods are usually time-consuming and require more computing resources during the test phase.

A. Our Method

To cope with the segmentation problem, we first turn to Hyperbolic space [6], [35] to better model the underlying hierarchical structures, as exponentially expanding Hyperbolic space has the innate ability to capture the structures of segmentation data. Then we adopt the uncertainty information during training to improve the performance of SS. To achieve this goal, we introduce the Hyperbolic Uncertainty Loss (HyperUL), which dynamically emphasizes the misclassified and not confidently classified pixels in the training phase. Specifically, as shown in Fig. 2, we first transform the embeddings of the last layer of trained SS models from Euclidean space into Hyperbolic space. Then HyperUL is appended to the transformed embeddings during training, and dynamically assigns distinct weights to different pixels by the calculated hyperbolic distances. Uncertain pixels whose embeddings are closer to the origin will obtain larger weights and vice versa. Moreover, inspired by Online Hard Example Mining (OHEM) [36], we design a corresponding Online High Uncertainty Example Mining (OHUEM) in HyperUL, which enables the HyperUL to pick higher uncertain pixels for back-propagation during training.

Regarding estimating the epistemic uncertainty, we propose to only post-process the obtained Hyperbolic embeddings. Specifically, for each pixel, its Hyperbolic distance from its Hyperbolic embedding to the origin is mapped to the interval $[0, 1]$. Then the generated value is subtracted by 1 to output the final uncertainty value. This can be easily implemented. By this way, we can get the uncertainty values almost for free.

To validate the effectiveness of our proposed HyperUL and Hyperbolic uncertainty estimation method, we conduct comprehensive experiments based on three deep learning methods, i.e., SegFormer-B0 [22], BiSeNetV2 [24], and STDC2-Seg50 [23], and three datasets, i.e., Cityscapes, UAVid, and ACDC. The experimental results on three datasets demonstrate that by employing our methods, existing SS models can consistently obtain segmentation performance improvement and output meaningful uncertainty estimations.

B. Our Contributions

The contributions of this paper are as follows:

1. We propose HyperUL, which is designed to dynamically weight higher uncertain pixels and select these pixels to compute losses for back-propagation. By considering such hyperbolic uncertainty information during training, HyperUL can improve SS performance.

2. We propose the Hyperbolic Uncertainty Estimation method, and the uncertainty values could be obtained almost for free.

3. For the semantic segmentation task, we provide comprehensive quantitative and qualitative comparison results between the proposed HyperUL and baseline losses (i.e., cross-entropy loss, cross-entropy combined with OHEM, and focal
loss) on Cityscapes, UAvid, and ACDC datasets. With the proposed HyperUL, existing SS models achieve additional segmentation performance improvement.

(4) For the uncertainty estimation task, we also carry out a series of experiments. The quantitative and qualitative experimental results demonstrate that the proposed Hyperbolic uncertainty estimation method usually produces more meaningful uncertainty results compared to the popular MC-dropout and ensembling approaches.

The rest of this paper is organized as follows. In Section II, we review related works. After that, we provide basics of Poincaré Ball model in Section III. In Section IV, we detail our proposed HyperUL and Hyperbolic uncertainty estimation approach. In Section V, we provide the experimental results on Cityscapes, UAvid, and ACDC datasets. Finally, we conclude our paper in Section VI.

II. RELATED WORK

In this section, we first introduce recent semantic segmentation methods, then Hyperbolic space related works, and finally uncertainty relevant approaches.

A. Semantic Segmentation

Recent works [23], [24], [37], [38] focus on the design of real-time and high-performance semantic segmentation (SS) models. For instance, the paper [24] designed a two-branch neural network, which simultaneously captures low-level details and obtains high-level semantic context to achieve a perfect speed-accuracy tradeoff. The work [23] inherited the desirable properties of BiSeNetV2, but adopted a detail guidance module to enforce the proposed SS model to learn spatial features directly instead of adding an extra branch. Building a long-range dependency or capturing global context information plays a key role in SS models. These works [22], [27], [39], [40] adopted various Transformer techniques to encode global contextual features to achieve better segmentation performances. For example, SegFormer [22] incorporated Transformer modules and lightweight multi-layer perception decoders in the design of the SS model. HRFormer [40] introduced a local-window self-attention in HRNet [41] to improve segmentation efficiency and performance. Although these networks have achieved high performances, they are only considered in Euclidean space and do not take the uncertainties of segmentation results into consideration. Hence, this work proposes HyperUL, which is in Hyperbolic space and incorporates the uncertainties in training SS models. With the proposed HyperUL, most existing SS models could achieve better performance.

B. Hyperbolic Space

Due to the outstanding ability of modeling the underlying hierarchical structure of datasets, Hyperbolic space [6], [35], [42], [43] has recently attracted increasing attention in the fields of Natural Language Processing (NLP) and Computer Vision (CV). These two works [42], [43] tried to create corresponding hyperbolic neural network operations to build pure hyperbolic neural networks in NLP. These papers [6], [15], [16], [44] demonstrated that in some practical CV applications, hyperbolic embeddings would be a better alternative compared with Euclidean and spherical embeddings. In this paper, we will use the properties of Hyperbolic space in the supervised SS task and to estimate the per-pixel uncertainty.

C. Uncertainty

For deep learning models, providing uncertainty estimation plays an important role in safety-critical real-world applications, such as medical image analysis and autonomous vehicle control. Many methods [7], [8], [9], [10], [11], [12], [29], [45] such as Bayesian inference, deep ensembles, and single deterministic networks containing explicit uncertainty prediction modules, have been proposed for modeling data uncertainty (i.e., aleatoric uncertainty) and model uncertainty (i.e., epistemic uncertainty). However, Bayesian inference and deep ensembles usually have large computational overhead. Single deterministic networks are not sufficient for safe decision-making. Besides, all of the above approaches do not utilize uncertainty during training. Although these works [13], [14] took uncertainty during the training stage, [13] only considered the uncertainty of the class, not every single pixel, and [14] intended to reduce the impact of uncertainty, not utilizing the uncertainty. In contrast, we use the hyperbolic distance to efficiently estimate the uncertainty of each pixel, and then adopt the calculated uncertainty values as weights to operate on corresponding pixels during training.

---

**TABLE I**

Real 4-Hyperbolicity Values of Cityscapes, UAvid, Mapillary Vistas, BDD, and ACDC Datasets. We First Use Different Segmentation Models to Extract All Features on Each Dataset. Then We Randomly Choose a Subset of 450 Feature Embeddings From All Extracted Features on Corresponding Data to Calculate One Hyperbolicity Value Each Time. We Repeat This Process 1000 Times. Therefore, the Final Results in This Table Are Mean and Standard Deviation of 1000 Values.

| Model          | Cityscapes  | UAvid    | Mapillary Vistas | BDD         | ACDC         |
|----------------|-------------|----------|------------------|-------------|--------------|
| SegFormer-B0 [23] | 0.19 ± 0.01 | 0.21 ± 0.01 | -                | -           | 0.20 ± 0.01  |
| STDTC2-Seg50 [23] | 0.20 ± 0.01 | 0.21 ± 0.01 | -                | -           | 0.20 ± 0.01  |
| BiSeNetV2 [24]  | 0.20 ± 0.01 | 0.21 ± 0.01 | -                | -           | 0.20 ± 0.01  |
| HMSA-HRNet-OCR [25] | 0.17 ± 0.02 | -        | 0.18 ± 0.02      | -           | -            |
| HMSA-ResNet-50 [25] | -          | -        | 0.19 ± 0.01      | -           | -            |
| ConvNeXt-B [26]  | -           | -        | -                | 0.16 ± 0.01 | -            |
| Swin-S [27]      | -           | -        | -                | 0.17 ± 0.01 | -            |
| Trans4Trans-M [28] | 0.19 ± 0.01 | -        | -                | -           | 0.20 ± 0.01  |
III. BASICS OF POINCARE BALL MODEL

In this section, we first briefly introduce Hyperbolic space, and then introduce the definition of Poincaré Ball model, and finally introduce commonly used operations in the Poincaré Ball model.

A. Hyperbolic Space

A Hyperbolic space is defined by a Riemannian manifold with a constant negative curvature. There are five isometric models and the most basic model is an n-dimensional Hyperbolicoid model with a hypersurface $\mathbb{H}_n^c$, as depicted in Fig. 3a. The Poincaré Ball model $\mathbb{B}_n^c$ is a projection of the Hyperboloid model onto the n-dimensional space-like hyperplane.

B. Poincaré Ball Model

An n-dimensional Poincaré Ball model with a constant sectional curvature $-c$ is defined as $(\mathbb{B}_n^c, g^c)$, where $\mathbb{B}_n^c = \{ x \in \mathbb{R}^n | \| x \| < 1 \}$ is the Hyperbolic space and $g^c_x = (\lambda^c_x)^2 I_n$ is the corresponding metric tensor at the point $x$. $\lambda^c_x$ is the conformal factor defined as $\lambda^c_x = 2 (1 - c \| x \|^2)^{-1}$. $I_n$ is the Euclidean metric tensor.

The $c^{-\frac{1}{2}}$ is the radius of Poincaré Ball. Same as previous works [42], [43], we adopt the formalism of Möbius gyrovector space [46] to provide basic Euclidean-like operations.

C. Möbius Addition

For $\forall x, y \in \mathbb{B}_n^c$, the Möbius addition is defined as follows:

$$ x \oplus_c y = \frac{(1 + 2 c(x, y) + c \| y \|^2)x + (1 - c \| x \|^2)y}{1 + 2 c(x, y) + c \| x \|^2 \| y \|^2}. \quad (1) $$

D. Exponential Map

For $\forall x, y \in \mathbb{B}_n^c$, and $\forall \nu \in T_x \mathbb{B}_n^c$ (i.e., $T_x \mathbb{B}_n^c$ can be seen as Euclidean space), the exponential map $\exp^c_x(\nu) : T_x \mathbb{B}_n^c \rightarrow \mathbb{B}_n^c$ is described as follows:

$$ \exp^c_x(\nu) = x \oplus_c \frac{1}{\sqrt{c}} \tanh \left( \frac{\sqrt{c} \lambda^c_x \| \nu \|}{2} \right) \nu. \quad (2) $$

E. Distance

For $\forall x, y \in \mathbb{B}_n^c$, the distance $d_c : \mathbb{B}_n^c \times \mathbb{B}_n^c \rightarrow \mathbb{R}$ is given by the following:

$$ d_c(x, y) = \frac{2}{\sqrt{c}} \tanh^{-1} \left( \sqrt{c} \parallel x \oplus_c y \parallel \right). \quad (3) $$

The distance between any point and the origin in the Poincaré Ball model indicates uncertainty of the point [6]. Specifically, there is a center in the ball. The hyperbolic distance from the embedding of each pixel to the origin (i.e., the center of the ball) can serve as a measure of how confidently or certainly the pixel will be classified. When pixels are classified with high confidence, their embeddings should be far away from the center. By contrast, for pixels classified with high uncertainty, their embeddings are closer to the center of the ball. For example, in Fig. 3b, the hyperbolic distance of the pixel C is longer than that of the pixels A and B, indicating that the pixel C is classified more confidently.

F. $\delta$-Hyperbolicity

As described in the introduction, $\delta$-hyperbolicity is mainly used to calculate the data structure similarity between Euclidean space and Hyperbolic space. Additionally, it can be utilized to estimate the manifold curvature $c$ of a Poincaré Ball model on specific datasets. According to [6] and [47], we set $\mathbb{E}^n$ as an n-dimensional Euclidean space and $d_c$ is the
distance function. The *Gromov product* for points \( x, y, z \in \mathbb{E}^n \) is defined as follows:

\[
(y, z)_x = \frac{1}{2} (d_\mathcal{E}(x, y) + d_\mathcal{E}(x, z) - d_\mathcal{E}(y, z)).
\] (4)

With a set of points, the matrix \( M \) of pairwise Gromov products can be calculated using Eq. 4. Then the \( \delta \)-hyperbolicity can be computed by the following Eqs. 5 and 6,

\[
M \otimes M = \max_{k} \left\{ M_{i, k}, M_{k, j} \right\},
\] (5)

\[
\delta = \max_{i,j} \left\{ (M \otimes M) - M \right\},
\] (6)

where \( \otimes \) means the min-max matrix product and \( i, j, k \) are indices of the matrix \( M \).

In practice, after we use segmentation models to extract feature embeddings of pixels on semantic segmentation datasets, we normalize these embeddings with the softmax function before computing the matrix \( M \). After using Eqs. 4, 5, and 6 to obtain the theoretical \( \delta \)-hyperbolicity value, we further calculate the scale-invariant one defined as:

\[
\delta_{rel} = \frac{\delta}{\max_{i,j} (M)}.
\] (7)

This equation is similar to the work [6]. When \( \delta_{rel} \) of a segmentation data set is close to 0, the data is with a higher hyperbolicity. To compare the hyperbolicity of various segmentation datasets, we compute hyperbolicity values with different segmentation models on five datasets, i.e., Cityscapes, UA Vid, Mapillary Vistas, BDD, and ACDC. Among these models, we train SegFormer-B0, BiSeNetV2, and STDC2-Seg50 on Cityscapes, UA Vid, and ACDC datasets. The rest of segmentation models on corresponding datasets are from publicly released code. To measure hyperbolicity values of different datasets as accurately as possible, we choose 450 feature embeddings each time from the extracted features of the data set to calculate a hyperbolicity value, and we repeat this process 1000 times. The final hyperbolicity results are mean and standard deviation values of the 1000 values. The experimental results are shown in Table I. We unveil that all segmentation datasets in Table I are with bigger hyperbolicity as their \( \delta_{rel} \) values are closer to 0 than 1.

Additionally, the \( \delta_{rel} \) can be adopted to roughly estimate the manifold curvature \( c \) of a Poincaré Ball model on a specific data set. As presented in [6], the relationship between \( \delta_{rel} \) and \( c \) is as follows:

\[
c = \left( \frac{0.144}{\delta_{rel}} \right)^2.
\] (8)

This estimated value \( c \) usually works well, but we also choose other values for parameter analysis and the experimental results are provided in Tables VII and VIII.

IV. THE PROPOSED METHOD

In this section, we first describe our proposed loss function. Then we detail the way to estimate the uncertainty values with our proposed Hyperbolic uncertainty estimation method.

A. The Proposed Loss Function

In this subsection, we first describe hyperbolic uncertainty again, then detail how to compute the uncertainty weight, and finally provide the proposed Hyperbolic Uncertainty Loss (HyperUL) and its application.

1) Hyperbolic Uncertainty: As mentioned in the introduction, hyperbolic uncertainty can be used to measure the uncertainty of per-pixel segmentation result by computing the hyperbolic distance from the pixel embedding \( x \) to the origin \( 0 \). According to Eq. 3, the distance is given by the following:

\[
d_\mathcal{E}(x, 0) = \frac{2}{\sqrt{c}} \tanh^{-1} \left( \sqrt{c} \| x \| \right),
\] (9)

where \( c \) is a hyper-parameter and \( -c \) is the constant sectional curvature of a Poincaré Ball model. According to Eq. 9, we know that the longer the distance is, the lower the uncertainty is.

2) Uncertainty Weight: In order to incorporate the hyperbolic uncertainty in the loss function, some transformations are needed. We first scale the distance to the interval \([0, 1]\) by diving by the largest distance \( d_\mathcal{E}(u, 0) \) in an image, which is

\[
d = \frac{d_\mathcal{E}(x, 0)}{d_\mathcal{E}(u, 0)},
\] (10)

where \( x \) is the hyperbolic embedding of each pixel, and \( u \) is the hyperbolic embedding of the most confidently classified pixel. Then the uncertainty weight is defined as follows:

\[
uw_x = \frac{1}{\log (t + d)},
\] (11)

where \( t \) is a hyper-parameter used to control the level of uncertainty weights, and the default value is \( t \approx 2.718 \). Due to \( \log_1 1 = 0 \) and \( \log_0 e = 1 \) and avoiding the 0 denominator, we choose five discrete values evenly from the interval \([1.02, 2.718]\) for \( t \) parameter analysis, and the results are shown in Table XII and Fig. 10. The relationships between uncertainty weights and \( t \) values are depicted in Fig. 4. It shows that smaller \( t \) provides bigger weights and wider variation range.

3) Cross Entropy Loss: According to the work [48], Cross Entropy Loss is defined as follows:

\[
L_{ce} = -u_k \log \frac{\exp(x_k)}{\sum_{m=1}^{M} \exp(x_m)}.
\] (12)
where $k$ indicates the target class, $w_k$ is the weight for the class $k$, and $M$ is the total number of classes. In Cross Entropy Loss, $w_k$ is commonly used for class imbalance.

4) Online High Uncertainty Example Mining (OHUEM): Similar to Online Hard Example Mining (OHEM) [36], we can also compute the losses only on the high uncertain pixels. Hyperbolic uncertainty can be used to find not confidently classified pixels. Therefore, it can serve as a criterion to only select the highest-uncertainty pixels for back-propagation during training, which behaves like OHEM. In conventional OHEM, each pixel in an image is sorted by its loss, and only these pixels with higher loss values can be selected for back-propagation. The rest of pixels will be completely discarded, as these pixels are seen as easy examples. Similarly, OHUEM uses the criterion of higher uncertainties instead of higher losses to choose pixels during training. For example, OHUEM only considers the losses of these pixels with top 70% uncertainty, while the losses of other pixels are totally ignored. The new criterion is given by the following:

$$d^h_c = d_c(x, 0) + h_r × (d_c(u, 0) - d_c(x, 0)).$$ (13)

where $d_c(x, 0)$ is the hyperbolic distance of each pixel, and $d_c(u, 0)$ is the hyperbolic distance of the most confidently classified pixel, and $h_r$ is used to decide “how many” pixels will be considered in computing losses for back-propagation. The best $h_r$ should be in $[0.3, 1]$. $h_r = 1$ means that all pixel losses will be considered in the training stage. We choose $h_r = [0.3, 0.5, 0.7, 0.9, 1.0]$ for parameter analysis, and the results are depicted in Table X and Fig. 9.

5) Hyperbolic Uncertainty Loss: By combining Eqs. 11, 12 and 13, we provide the final Hyperbolic Uncertainty Loss (HyperUL) as follows:

$$\text{HyperUL} = \begin{cases} - (uw_x) w_k \log \frac{\exp(x_k)}{\sum_{m=1}^{M} \exp(x_m)}, & d_c(x, 0) \leq d^h_c; \\ 0, & \text{otherwise}, \end{cases}$$ (14)

where as described in the above mentioned content, $uw_x$ is the uncertainty weight and $d^h_c$ is the criterion of OHUEM. There are three hyper-parameters $c$, $t$, and $h_r$. In the following parameter analysis, we first provide parameter analysis results about $c$ under the condition of $t = 2.718, h_r = 1$. Then parameter analysis results about $h_r$ and $t$ are provided. All results are demonstrated in Tables VIII, X and XII, as well as Figs. 8, 9, and 10, respectively.

6) Application of HyperUL: As shown in Fig. 2, after passing through a segmentation model’s encoder and decoder parts, an input image is transformed into Euclidean embeddings $v$, which is the last layer of the segmentation model. Then the Euclidean embeddings are converted into corresponding Hyperbolic embeddings by the Eq. 2. Finally, for the semantic segmentation task, we calculate the loss by the proposed HyperUL during the training phase. Meanwhile we do post-processing to estimate the uncertainty of per pixel of the input image.

B. Uncertainty Estimation

In this subsection, we detail how to measure the uncertainty of each pixel in an image. We first describe commonly used MC-dropout and ensembling uncertainty estimation approaches. Then we illustrate our proposed Hyperbolic uncertainty measurement method.

1) MC-dropout & Ensembling Uncertainty Estimation: Bayesian inference is the principled method to calculate the model uncertainty (i.e., epistemic uncertainty). However, there are a large number of parameters in deep learning based segmentation models, and using conventional Bayesian inference to estimate the epistemic uncertainty is intractable. At present, MC-dropout [7], [29], [31] and ensembling [8], [33], [34] are popular approaches to estimate uncertainty as they can be easily applied for deep learning models. Therefore, we adopt MC-dropout and ensembling as the baselines in this paper.

Similar to the works [7], [32], for MC-dropout, we place the dropout layer in segmentation models and during the testing stage, we still use the dropout layer to approximately pick $M$ samples from network weights to calculate the uncertainty. Regarding ensembling, similar to the works [8], [32] we train the same segmentation model $M$ times with the same training procedure and random parameter initialization. During the testing phase, we use all $M$ models to estimate the uncertainty. Here we set $M = 8$. More detailed implementations are described in Section V-C.

2) Hyperbolic Uncertainty Estimation: As depicted in Fig. 3b, the uncertainty of a pixel is estimated by computing the hyperbolic distance from pixel’s hyperbolic embedding $x$ to the origin $a$ using the Eq. 9. This is easily implemented since our proposed segmentation model eventually outputs hyperbolic embeddings for pixels of an input image. We just need to do post-processing. To map all uncertainty values of an image to the interval $[0, 1]$, similar to the Eq. 10, the uncertainty of per pixel could be calculated by the following formula:

$$p_{\text{uncertainty}} = 1 - \frac{d_c(x, 0)}{d_c(u, 0)},$$ (15)

where $x$ is the hyperbolic embedding of each pixel, and $u$ is the hyperbolic embedding of the most confidently classified pixel. $p_{\text{uncertainty}} = 0$ means the lowest uncertainty while $p_{\text{uncertainty}} = 1$ corresponds to the highest uncertainty.

Note that calculating the hyperbolic uncertainty values by this way is almost for free. After we trained a segmentation model, we can directly get the uncertainty results by the Eq. 15. It is also a single pass approach. In contrast, MC-dropout and ensembling based methods need multiple passes to calculate uncertainty values. They commonly consume more time and computing resources especially when $M$ is larger. The comparisons among MC-dropout, ensembling, and the proposed Hyperbolic uncertainty estimation methods are provided in Tables XIV, XV, and XVI, as well as Figs. 11, 12, and 13.

V. Experiments

In this section, we first introduce the experimental settings, i.e., employed semantic segmentation models, adopted
datasets, and implementation details. Then we show our semantic segmentation results on three typical scene parsing datasets, e.g., Cityscapes [1], UA Vid [18], and ACDC [21]. The results include quantitative comparisons, qualitative comparisons, parameter analysis, and training time comparisons. Finally, we compare uncertainty estimation results with three uncertainty measurement approaches on three datasets by providing quantitative and qualitative results.

A. Experimental Settings

1) Models: As mentioned in Section IV-A.6, SegFormer-B0 [22], BiSeNetV2 [24], and STDC2-Seg50 [23] are recently proposed lightweight and high-performance semantic segmentation approaches. For fair comparisons, we use the cross-entropy loss, cross-entropy loss combined with OHEM, and focal loss as baselines to calculate losses when training SegFormer-B0, BiSeNetV2, and STDC2-Seg50. For our proposed HyperUL, we replace these baseline loss functions with HyperUL and term these new segmentation models as “SegFormer-B0+HyperUL”, “BiSeNetV2+HyperUL”, and “STDC2-Seg50+HyperUL”, respectively.

2) Datasets: We conduct experiments on three publicly released datasets, i.e., Cityscapes [1], UA Vid [18], and ACDC [21]. Cityscapes is a large semantic segmentation datasets, i.e., 5000 fine annotated images with commonly used 19 classes, especially for the field of autonomous driving. This dataset is divided into 2975 training images, 500 validation images, and 1525 test images, respectively. All images are with the high resolution of 2048 × 1024. UA Vid is also a high-resolution street scene semantic segmentation dataset. The images in it were collected by Unmanned Aerial Vehicles (UAV). There are two types of image resolutions, i.e., 3840 × 2160 and 4096 × 2160. UA Vid is comprised of 200 training images, 70 validation images, and 150 test images, respectively. Also, it has been densely labeled with 8 classes for the UAV-based semantic segmentation task. ACDC was introduced recently to promote the development of robust visual perception systems and hence all images in it were captured under four common adverse weather conditions, including 1000 foggy, 1006 nighttime, 1000 rainy, and 1000 snowy images with the size of 1920 × 1080. Same as Cityscapes, all images provide per-pixel annotations with 19 traffic-related classes. Besides, all images under each adverse condition are separated into 400 training images, 100 validation images (i.e., nighttime data set is with 106 validation images), and 500 test images.

3) Implementation Details: We train all models on a server with 4 Tesla A100 GPUs. We initialize the backbones of SegFormer-B0(+HyperUL), and STDC2-Seg50(+HyperUL) with corresponding pretrained models on the Imagenet-1K dataset, while same as previous work [24], BiSeNetV2(+HyperUL) is trained from scratch. When training these models, we adopt multiple data augmentation techniques [22], [23], such as Random Resize with the ratio of [0.5, 2.0], Random Horizontal Flipping, and Random Cropping to 1024 × 512 for Cityscapes, UA Vid, and ACDC. Besides, the batch size for SegFormer-B0(+HyperUL), BiSeNetV2(+HyperUL), and STDC2-Seg50(+HyperUL) is set to 24 on three datasets. We utilize AdamW optimizer [49] with initial learning rate of 0.00015 for the encoder module and 0.0015 for the decoder module for SegFormer-B0(+HyperUL), and we set 0.0015 for BiSeNetV2(+HyperUL) and STDC2-Seg50(+HyperUL). As commonly used configuration, Polynomial Learning Rate Decay Scheduler with the power of 0.9 is used. In addition, we train all models for 80K iterations on three datasets, and adopt the warmup trick for the first 1K iterations. Finally, we use IoU and mIoU to evaluate the segmentation performance.

B. Semantic Segmentation

In this subsection, we first provide comparison results between the proposed SegFormer-B0+HyperUL, BiSeNetV2+HyperUL, and STDC2-Seg50+HyperUL and their counterparts, SegFormer-B0, BiSeNetV2, and STDC2-Seg50 on Cityscapes, UA Vid, and ACDC test datasets. Then we compare the proposed loss with cross-entropy loss, cross-entropy combined with OHEM, and focal loss by providing quantitative results and qualitative results on all validation sets. Finally, we show analysis results of hyper-parameters $c$, $h_t$, and $t$ in the proposed loss.

1) Comparisons on Three Datasets: Similar to works [22], [23], [24], we trained segmentation models on both the training and validation datasets, and then submitted testing results to the benchmarks to obtain the final mIoU scores. Specifically, for the BiSeNetV2 and STDC2-Seg50 models on Cityscapes, we obtained the mIoU scores directly from the works [23] and [24]. Regarding the rest of segmentation models on Cityscapes, UA Vid, and ACDC, we trained them for 160k iterations and got their mIoU scores from corresponding benchmarks. To further compare the proposed HyperUL with cross-entropy loss, cross-entropy combined with OHEM, and focal loss, we trained all models working with these loss functions only on the training datasets and reported per-class IoU and mIoU scores on the validation datasets. Also we provided corresponding qualitative comparisons. All experimental results are shown in Tables II, III, IV, V, and VI, as well as in Figs. 5, 6, and 7.

a) Quantitative comparisons: As shown in Table II, we see that equipped with our proposed HyperUL, existing segmentation models can consistently achieve segmentation performance improvement. Note that during the training
phase, BiSeNetV2 and STDC2-Seg50 were trained using cross-entropy loss and OHEM. These performance gains validate the effectiveness of our proposed method.

Tables III, IV, and V further investigate the impact of our proposed loss on per-class performance on three datasets. Moreover, we compare the proposed loss with cross-entropy loss, cross-entropy joined with OHEM, and focal loss. We find that applied with the HyperUL, SegFormer-B0+HyperUL, BiSeNetV2+HyperUL, and STDC2-Seg50+HyperUL have consistently obtained the best segmentation scores on the Bus class. Importantly, the proposed HyperUL helps BiSeNetV2+HyperUL, STDC2-Seg50+HyperUL improve segmentation performance on Car, Truck, Bus, and Bicycle categories on Cityscapes, which are challenging but critical classes for autonomous driving systems. Similarly, on the ACDC dataset, with the help of HyperUL, SegFormer-B0+HyperUL obtains the best IoU scores on Road, Rider, Car, Truck, and Bicycle classes. BiSeNetV2+HyperUL also achieves the best IoU scores on Fence, Rider, Truck, and Bus classes. These classes are important for autonomous vehicles. Besides, we provide comparison results in four different weather conditions on ACDC, as shown in Table VI. We see that by employing HyperUL, SegFormer-B0+HyperUL
shows some improvements in fog, nighttime, and snow conditions. BiSeNetV2+HyperUL achieves improvements in fog and rain, as well as STDC2-Seg50+HyperUL obtains improvements in both nighttime and snow. On the UA Vid, we see that BiSeNetV2 and STDC2-Seg50 models trained with our proposed HyperUL consistently obtain the better segmentation results on half of all classes, e.g., Clutter, Road, Tree, and Vegetation. All results validate the effectiveness of our proposed HyperUL, especially on vital classes for autonomous vehicles.

b) Qualitative comparisons: Consistent with the IoU results in Table III on Cityscapes, SegFormer-B0+HyperUL, BiSeNetV2+HyperUL, and STDC2-Seg50+HyperUL can coherently and completely segment the regions of Rider, Bus, Sidewalk, Motorcycle, and Truck, as highlighted by white rectangles and circles in Fig. 5. By contrast, trained with cross-entropy loss, cross entropy loss and OHEM, and focal loss, the segmentation models fail to completely pick these challenging but important classes from images. On the ACDC dataset, we see the similar results. The regions of interesting classes, i.e., Motorcycle, Sidewalk, Car, Vegetation, Bus, and Road, segmented by three models with the proposed HyperUL are more coherent and complete than their counterparts. These qualitative results are emphasized by white rectangles and circles in Fig. 7. Similarly, on UA Vid, with the help of HyperUL, the segmentation models have the ability to further improve performance by refining small regions, as indicated in white rectangles in Fig. 6. All of qualitative comparisons verify the effectiveness of the proposed HyperUL.

2) Parameter Analysis: In this subsection, we show the impacts on segmentation performance of SegFormer-B0+HyperUL, BiSeNetV2+HyperUL, and
STDC2-Seg50+HyperUL with different values of hyper-parameters \(c\), \(h_r\), and \(t\) in the Eq. 14 on Cityscapes, UAVid, and ACDC validation datasets. We trained all models on the training datasets for 60k iterations and obtained the mIoU values on the validation datasets.

\(a)\) Hyper-parameter \(c\): As illustrated in Section III, the value of \(c\) is the manifold curvature of a Poincaré Ball model. When a \(c\) value is closer to 0, the corresponding Poincaré Ball is approximately flat as the Euclidean space. By contrast, larger \(c\) value makes a Poincaré Ball model steeper. The \(c\) can be roughly measured by the Eq. 8. The approximately measured \(c\) values by SegFormer-B0, BiSeNetV2, and STDC2-Seg50 models on Cityscapes, UAVid, and ACDC datasets are shown in Table VII. However, to comprehensively analyze the impacts of the parameter \(c\), in addition to the estimated ones shown in Table VII, we also choose \(c = \{0.1, 0.3, 0.5, 0.7, 0.9, 1.0\}\) to conduct segmentation experiments on three datasets. The impacts are measured by the mIoU scores (%) obtained using SegFormer-B0+HyperUL, BiSeNetV2+HyperUL, and STDC2-Seg50+HyperUL models on three datasets with different \(c\) values, as reported in Table VIII and Fig. 8.

We find that under the chosen \(c\) values, outputs from three segmentation models are generally stable. This means that the variations of \(c\) values do not significantly influence the segmentation results. Moreover, utilizing the roughly estimated

**Fig. 7.** Qualitative segmentation examples of SegFormer-B0, BiSeNetV2, and STDC2-Seg50 with various loss functions on the ACDC validation data set. Here CE corresponds to the cross-entropy loss. CE+OHEM means the cross-entropy loss combined with Online Hard Example Mining. HyperUL refers to the proposed Hyperbolic Uncertainty Loss. (Best viewed in color.)

**TABLE VI**
COMPARISONS (mIoU %) BETWEEN THE PROPOSED HYPERUL AND CROSS-ENTROPY LOSS (CE), CROSS-ENTROPY LOSS USED WITH OHEM (CE+OHEM), AND FOCAL LOSS IN FOUR DIFFERENT WEATHER CONDITIONS ON THE ACDC VALIDATION DATA

| Model         | Loss | Fog | Nighttime | Rain | Snow |
|---------------|------|-----|-----------|------|------|
| SegFormer-B0  | CE   | 73.32 | 47.77 | 65.58 | 70.88 |
|               | CE+OHEM | 73.65 | 47.79 | **68.73** | 70.13 |
|               | Focal Loss | 73.71 | 47.07 | 67.03 | 71.46 |
|               | HyperUL | 74.61 | 48.40 | 66.33 | **71.47** |
| BiSeNetV2     | CE   | 69.08 | 50.29 | 62.99 | 67.97 |
|               | CE+OHEM | 72.12 | 47.42 | 64.72 | 66.72 |
|               | Focal Loss | 69.09 | 48.66 | 62.24 | 67.31 |
|               | HyperUL | 72.16 | 48.01 | 65.55 | 67.34 |
| STDC2-Seg50   | CE   | 71.97 | 50.27 | 67.58 | 68.63 |
|               | CE+OHEM | 74.00 | 50.98 | 67.34 | 71.68 |
|               | Focal Loss | 73.73 | 52.07 | **67.66** | 69.84 |
|               | HyperUL | 73.61 | 52.45 | 66.87 | **72.69** |

**TABLE VII**
ESTIMATED \(c\) VALUES BY SEGFORMER-B0, BISENETV2, AND STDC2-SEG50 ON CITYSCAPES, UAVID, AND ACDC DATASETS

| Model         | Cityscapes  | UAVid | ACDC |
|---------------|-------------|-------|------|
| SegFormer-B0  | 0.553       | 0.473 | 0.522 |
| BiSeNetV2     | 0.535       | 0.474 | 0.502 |
| STDC2-Seg50+HyperUL | 0.520 | 0.474 | 0.502 |

**TABLE VIII**
The Impacts (i.e., mIoU (%)) of the Parameter \(c\) Used in SegFormer-B0+HyperUL (Se.H), BiSeNetV2+HyperUL (Bi.H), and STDC2-Seg50+HyperUL (ST.H) on the Cityscapes, UAVid, and ACDC Validation Sets With \(h_r\) Fixed to 1.0 and \(t\) Fixed to 2.718

| \(c\) | Cityscapes  | UAVid | ACDC |
|-------|-------------|-------|------|
| \(0.1\) | 73.8       | 74.6   | 76.0  |
| \(0.3\) | 73.2       | 74.0   | 76.0  |
| \(0.5\) | 73.1       | 74.0   | 76.0  |
| \(0.7\) | 73.9       | 75.1   | 76.0  |
| \(0.9\) | 73.4       | 72.6   | 75.9  |
| \(C\)  | 73.3       | 74.1   | 76.3  |
c values in these segmentation models usually works quite well, as shown in the last row in Table VIII. Therefore, the c value in our approach can be easily fine-tuned for practical use.

b) Hyper-parameter $h_r$: As described in Eq. 13, the hyper-parameter $h_r$ is used to control “how many” pixels will be taken into consideration when computing losses for back-propagation during the training phase. $h_r$ can be adopted to control which pixel will be considered. As $h_r$ becomes bigger, more uncertain pixels will be selected for back-propagation. Here we set $h_r = \{0.3, 0.5, 0.7, 0.9, 1.0\}$ for sensitivity analysis. Parameter analysis results are provided in Table X and Fig. 9.

We see that when $h_r = \{0.7, 0.9, 1.0\}$, the performance of three segmentation models is not affected significantly. However, using $h_r = 0.3$ for these models destroys segmentation performance (i.e., see the second row in Table X). It means that considering at least 70% top uncertain pixels during the training stage commonly leads to competitive performance.

c) Hyper-parameter $t$: The uncertainty weight (i.e., Eq. 11) is adopted to dynamically highlight top uncertain pixels. During the training phase, these top uncertain pixels are allocated larger weights to their losses. This drives models to pay more attention to unconfident classification results. $t$ is used to control the variations of uncertainty weights. As shown in Fig. 4, when $t$ becomes bigger, the uncertainty weights will be relatively smaller, and the dynamic range of weights for each pixel also becomes limited. For example, when $t = 1.02$, the wide range of weights is from 1.42 to 50.5. In contrast, when $t = 2.718$, the narrow range of weights is only from 0.76 to 1.0. We conduct comparison experiments with $t = \{1.02, 1.445, 1.869, 2.294, 2.718\}$ under the fixed values $c$ and $h_r$ shown in Table XI. Parameter analysis results are shown in Table XII and Fig. 10.

We see that the variations of $t$ values cannot strongly influence segmentation results. In addition, it is more likely that under $t = 1.869$, the best performance is easily achieved by segmentation models.

3) Comparison of Training Time: In this subsection, we compare the training time of SegFormer-B0, BiSeNetV2, and STDC2-Seg50 equipped with different loss functions, i.e., cross entropy loss, cross entropy with Online Hard Example Mining, focal loss, and the proposed HyperUL on Cityscapes, UAVid, and ACDC. To fairly compare these models, we trained all of them under the same conditions. All models are trained on four Nvidia A100 GPUs. The batch size and the number of iterations are set to 24 and 80K, respectively. More implementation details have been

| TABLE IX | SELECTED c VALUES USED FOR PARAMETER ANALYSIS OF $h_r$ |
|----------|---------------------------------|
| Cityscapes | UAVid | ACDC |
| SegFormer-B0 + HyperUL | BiSeNetV2 + HyperUL | STDC2-Seg50 + HyperUL |
| 0.7 | 0.7 | 0.520 |
| 0.1 | 0.1 | 0.9 |
| 0.522 | 0.3 | 0.7 |

| TABLE X | THE IMPACTS (i.e., mIoU (%)) OF THE PARAMETER $h_r$ USED IN SegFormer-B0 + HyperUL (Se.H), BiSeNetV2 + HyperUL (Bi.H), and STDC2-Seg50 + HyperUL (ST.H) ON THE CITYSCAPES, UAVID, AND ACDC VALIDATION SETS. THE FIXED c VALUES CORRESPONDING TO EACH MODEL AND DATASET ARE SHOWN IN TABLE IX, AND THE t VALUE IS FIXED TO 2.718 |
|----------|---------------------------------|
| $h_r$ | Cityscapes | UAVid | ACDC |
| 0.3 | 70.3 | 71.3 | 76.1 | 77.3 | 80.6 | 84.8 | 71.1 | 61.6 | 66.4 |
| 0.5 | 73.7 | 73.5 | 76.7 | 82.0 | 82.3 | 84.6 | 83.4 | 83.6 | 66.5 |
| 0.7 | 73.6 | 75.0 | 76.6 | 83.7 | 83.6 | 84.9 | 86.0 | 64.5 | 66.5 |
| 0.9 | 73.5 | 74.3 | 76.3 | 84.2 | 84.0 | 85.9 | 66.3 | 63.2 | 66.6 |
| 1.0 | 73.9 | 75.1 | 76.3 | 83.8 | 84.1 | 85.3 | 66.1 | 65.0 | 67.7 |

| TABLE XI | SELECTED c AND $h_r$ VALUES USED FOR t PARAMETER ANALYSIS |
|----------|---------------------------------|
| $c$ | $h_r$ | Cityscapes | UAVid | ACDC |
| SegFormer-B0 + HyperUL | BiSeNetV2 + HyperUL | STDC2-Seg50 + HyperUL |
| 0.7 | 0.7 | 0.520 | 0.5 |
| 0.1 | 0.9 | 0.9 | 0.9 | 0.9 |
| 0.522 | 0.3 | 0.7 | 1.0 |

| TABLE XII | THE IMPACTS (i.e., mIoU (%)) OF THE PARAMETER $t$ USED IN SegFormer-B0 + HyperUL (Se.H), BiSeNetV2 + HyperUL (Bi.H), and STDC2-Seg50 + HyperUL (ST.H) ON THE CITYSCAPES, UAVID, AND ACDC VALIDATION SETS. THE FIXED c VALUES AND $h_r$ VALUES CORRESPONDING TO EACH MODEL AND DATASET ARE SHOWN IN TABLE XI |
|----------|---------------------------------|
| $t$ | Cityscapes | UAVid | ACDC |
| 1.02 | 73.9 | 72.8 | 75.8 | 83.3 | 85.7 | 85.4 | 66.1 | 64.5 | 66.7 |
| 1.445 | 73.5 | 73.3 | 76.0 | 83.8 | 84.3 | 85.6 | 66.4 | 64.2 | 67.5 |
| 1.869 | 73.2 | 75.0 | 76.9 | 83.9 | 83.5 | 85.8 | 66.8 | 63.9 | 68.1 |
| 2.294 | 73.7 | 73.7 | 76.0 | 83.9 | 84.3 | 85.8 | 65.3 | 63.6 | 67.5 |
| 2.718 | 73.9 | 75.1 | 76.7 | 84.2 | 84.1 | 85.9 | 66.3 | 65.0 | 67.7 |
Fig. 9. Sensitivity analysis for the parameter $h_r$ used in SegFormer-B0+HyperUL, BiSeNetV2+HyperUL, and STDC2-Seg50+HyperUL models on Cityscapes, UA Vid, and ACDC datasets. The variations of $h_r$ values are {0.3, 0.5, 0.7, 0.9, 1.0}. mIoU scores (%) are used to measure the impacts. When conducting experiments, we use the fixed $c$ values depicted in Table IX and $t = 2.718$.

Fig. 10. Sensitivity analysis for the parameter $t$ used in SegFormer-B0+HyperUL, BiSeNetV2+HyperUL, and STDC2-Seg50+HyperUL models on Cityscapes, UA Vid, and ACDC datasets. $t$ values are {0.3, 0.5, 0.7, 0.9, 1.0}. mIoU scores (%) are used to assess the impacts. When conducting experiments, we fix $c$ and $h_r$ values illustrated in Table XI.

TABLE XIII

| Model           | Loss          | Cityscapes | UA Vid | ACDC |
|-----------------|---------------|------------|--------|------|
| SegFormer-B0    | CE            | 17.5       | 19.4   | 20.0 |
|                 | CE + OHEM     | 18.2       | 19.6   | 20.4 |
|                 | Focal Loss    | 18.2       | 19.6   | 20.3 |
|                 | HyperUL       | 20.3       | 20.0   | 24.0 |
| BiSeNetV2       | CE            | 17.6       | 18.9   | 19.4 |
|                 | CE + OHEM     | 17.9       | 19.0   | 20.0 |
|                 | Focal Loss    | 18.3       | 19.3   | 20.5 |
|                 | HyperUL       | 21.2       | 20.2   | 24.1 |
| STDC2-Seg50     | CE            | 17.8       | 19.2   | 20.1 |
|                 | CE + OHEM     | 18.4       | 19.5   | 21.0 |
|                 | Focal Loss    | 18.7       | 19.9   | 21.3 |
|                 | HyperUL       | 22.6       | 21.4   | 25.4 |

TABLE XIV

| Model           | Loss          | Cityscapes | UA Vid | ACDC |
|-----------------|---------------|------------|--------|------|
| SegFormer-B0    | CE            | 17.5       | 19.4   | 20.0 |
|                 | CE + OHEM     | 18.2       | 19.6   | 20.4 |
|                 | Focal Loss    | 18.2       | 19.6   | 20.3 |
|                 | HyperUL       | 20.3       | 20.0   | 24.0 |
| BiSeNetV2       | CE            | 17.6       | 18.9   | 19.4 |
|                 | CE + OHEM     | 17.9       | 19.0   | 20.0 |
|                 | Focal Loss    | 18.3       | 19.3   | 20.5 |
|                 | HyperUL       | 21.2       | 20.2   | 24.1 |
| STDC2-Seg50     | CE            | 17.8       | 19.2   | 20.1 |
|                 | CE + OHEM     | 18.4       | 19.5   | 21.0 |
|                 | Focal Loss    | 18.7       | 19.9   | 21.3 |
|                 | HyperUL       | 22.6       | 21.4   | 25.4 |

Provided in the section V-A. Comparison results are shown in Table XIII.

From the Table XIII, we see that the incorporation of the HyperUL in the segmentation models increases little extra training time compared with their counterparts applied with other loss functions. However, the experimental results in the subsection V-B reveal that HyperUL-based models are able to consistently improve the segmentation performance. Importantly, with the proposed method, we can easily estimate the uncertainty almost for free, as illustrated in the following subsection V-C.

C. Uncertainty Estimation

In this subsection, we first compare ensembling, MC-dropout, and the proposed Hyperbolic uncertainty estimation method in terms of the ECE [50], [51] and mIoU metrics on Cityscapes, UA Vid, and ACDC validation datasets. Then we provide corresponding qualitative comparisons on three datasets.

Specifically, for the ensembling uncertainty estimation approach, similar to the work [32], we train SegFormer-B0, BiSeNetV2, and STDC2-Seg50 models 8 times with the same training procedure on Cityscapes, UA Vid, and ACDC, respectively. During the testing phase, we use these 8 models together to calculate ECE and mIoU values on the validation datasets. Regarding the MC-dropout uncertainty measurement method, similar to the works [32], [52], we place a dropout layer with $p = 0.5$ after the four Transformer Blocks 1, 2, 3, and 4 in SegFormer-B0. We add a dropout layer with $p = 0.5$ after two $1 \times 1$ convolution operations at the end of Detail Branch and Semantic Branch in the Bilateral Guided Aggregation Layer in BiSeNetV2. In STDC2-Seg50, we place the dropout layer with the possibility of $p = 0.5$ between two Attention Refine Module and Feature Fusion Module, as well as after the Feature Fusion Module. In the testing stage, we switch all dropout layers to the training mode in the segmentation model and then run each model 8 times to compute ECE and mIoU values. For Hyperbolic uncertainty estimation method, we directly run the trained SegFormer-B0+HyperUL, BiSeNetV2+HyperUL, and STDC2-Seg50+HyperUL models and use the Eq. 15 to measure ECE values.

1) Quantitative Comparisons: Table XIV shows the comparison results of three uncertainty estimation methods with three segmentation models on Cityscapes, UA Vid, ACDC validation datasets. We see that SegFormer-B0 and BiSeNetV2 together with MC-dropout and Ensembling usually obtain
better performance in terms of the ECE and mIoU values on three datasets compared to that with Hyperbolic. However, STDC2-Seg50 working with Hyperbolic consistently achieves better results on three datasets, as shown in the last row in Table XIV.

Additionally, we observe that compared with CNN-based architectures (i.e., BiSeNetV2 and STDC2-Seg50), Transformer-based architecture (i.e., SegFormer-B0) appears more certain about its predictions in terms of smaller ECE values shown in the third, fifth, and seventh columns in Table XIV. We guess that the main reason is that Transformer-based models commonly have the innate ability to model long-range context and to extract global features and hence to obtain more confident results. Moreover, compared with results on Cityscapes and UAVid, results on the ACDC dataset show larger ECE values when using SegFormer-B0 with all three uncertainty estimation methods. This is depicted in the 3-5 rows and the third, fifth, and seventh columns in Table XIV. A reason is that ACDC is collected under more adverse weather conditions and hence ACDC data contains more irreducible data noise with higher aleatoric uncertainty.

More important, we find that working together with the proposed Hyperbolic uncertainty estimation method, STDC2-Seg50 obtains better ECE results on three datasets (i.e., see the last three rows in Table XIV). Similarly, BiSeNetV2 also achieves a better ECE value on the UAVid validation data set (i.e., see the eighth row and fifth column in Table XIV). This means that with the proposed Hyperbolic uncertainty measurement approach, existing segmentation models are more likely to output meaningful certain predictions. This is also verified by the following qualitative comparison results.

2) Qualitative Comparisons: Figs. 11, 12, and 13 illustrate qualitative comparisons of MC-dropout, ensembling, and the proposed Hyperbolic uncertainty measurement approach with three segmentation models and three datasets. We see that consistent with results in Table XIV, SegFormer-B0 with three uncertainty prediction methods outputs meaningful results. The generated uncertainty maps clearly show larger uncertain predictions at object boundaries due to the inherent ambiguity (i.e., see the second row in Figs. 11, 12, and 13). BiSeNetV2 combined with MC-dropout and ensembling generates relatively better performance on Cityscapes and ACDC datasets.

However, with our Hyperbolic uncertainty estimation method, BiSeNetV2 significantly outputs more meaningful results on UAVid, as shown in the fourth row and the last column in Fig. 12. More important, equipped with the proposed Hyperbolic-based approach, STDC2-Seg50 consistently generates better uncertainty maps on three datasets compared to the counterparts with MC-dropout and ensembling (i.e., see the last row and last column in Figs. 11, 12, and 13). All results...
verify that the proposed Hyperbolic uncertainty estimation method is effective.

3) Training and Test Time Comparisons: Comparison results are provided in Tables XV and XVI. According to the above experimental settings, all ensembling-based models are trained 8 times separately during the training phase, and a group of 8 models are inferred sequentially during the test phase. Therefore, we see that in Tables XV and XVI, ensembling-based approaches are time-consuming during both the training and test stages. Regarding MC-dropout, corresponding models are trained only once but inferred 8 times with all dropout layers switched to the training mode in the test stage. Hence, it can be seen that in Table XVI, MC-dropout related approaches need more inference time. By contrast, hyperbolic relevant uncertainty estimation approaches are more efficient than their counterparts because these hyperbolic related methods are inferred only once (i.e., see the last row in Table XVI).

VI. CONCLUSION

Hyperbolic Uncertainty Loss (HyperUL) and Hyperbolic Uncertainty Estimation were proposed in this work to handle underlying hierarchical structures of segmentation datasets and uncertainty prediction. For the semantic segmentation task, in the paper, we detailed our proposed HyperUL and applied them to recently representative segmentation models. By employing the proposed HyperUL, existing segmentation models consistently obtained additional performance improvement compared to their counterparts. Additionally, with the proposed Hyperbolic uncertainty estimation method, we could calculate the epistemic uncertainty almost for free. Meanwhile, the proposed uncertainty estimation method commonly produced more meaningful results.

However, there is a disadvantage in this paper. This work only considered utilizing the properties of Hyperbolic space on the segmentation datasets instead of taking more complex spaces such as a product manifold (i.e., with multiple model spaces, like Spherical, Hyperbolic, and Euclidean spaces). It is possible that a product manifold is more suitable for analyzing the segmentation datasets.

Therefore, there are at least three research directions. The first is that we could try the product manifold applied to the semantic segmentation task. The second is that we might try to find the similar property from Euclidean space to calculate the uncertainty. The last one is that our HyperUL and Hyperbolic uncertainty estimation methods could be applied to other tasks such as 3D point cloud segmentation and medical image segmentation, as the proposed methods are general in nature.

ACKNOWLEDGMENT

The authors wish to acknowledge the CSC–IT Center for Science, Finland, for the provision of computational resources.

REFERENCES

[1] M. Cordts et al., “The cityscapes dataset for semantic urban scene understanding,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 3213–3223.
[2] A. Milioto, P. Lottes, and C. Stachniss, “Real-time semantic segmentation of crop and weed for precision agriculture robots leveraging background knowledge in CNNs,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2018, pp. 2229–2235.
[3] Y. Qian, L. Deng, T. Li, C. Wang, and M. Yang, “Gated-residual block for semantic segmentation using RGB-D data,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 8, pp. 11836–11844, Aug. 2022.
[4] B. Xie et al., “Multi-scale fusion with matching attention model: A novel decoding network cooperated with NAS for real-time semantic segmentation,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 8, pp. 12622–12632, Aug. 2022.
[5] B. Chen, C. Gong, and J. Yang, “Importance-aware semantic segmentation for autonomous vehicles,” IEEE Trans. Intell. Transp. Syst., vol. 20, no. 1, pp. 137–148, Jan. 2019.
[6] V. Khrulkov, L. Mirvakhabova, E. Ustinova, I. Oseledets, and V. Lempitsky, “Hyperbolic image embeddings,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 6418–6428.
[7] Y. Gal and Z. Ghahramani, “Dropout as a Bayesian approximation: Representing model uncertainty in deep learning,” in Proc. Int. Conf. Mach. Learn., vol. 48, 2016, pp. 1050–1059.
[8] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” in Proc. Adv. Neural Inf. Process. Syst., vol. 30, 2017, pp. 1–12.
[9] M. Valdenebro-Toro, “Deep sub-ensembles for fast uncertainty estimation in image classification,” in Proc. 4th Workshop Bayesian Deep Learn., 2019, pp. 1–7.
[10] M. Sensory, L. Kaplan, and M. Kandemir, “Evidential deep learning to quantify classification uncertainty,” in Proc. Adv. Neural Inf. Process. Syst., vol. 31, 2018, pp. 1–11.
[11] Y. Fathullah and M. J. F. Gales, “Self-distribution distillation: Efficient uncertainty estimation,” in Proc. Conf. Uncertainty Artif. Intell., vol. 18, 2022, pp. 663–673.
[12] M. Raghun et al., “Direct uncertainty prediction for medical second opinions,” in Proc. Int. Conf. Mach. Learn., Jun. 2019, pp. 5281–5290.
[13] B. Bischke, F. Helber, D. Both, and A. Dengel, “Segmentation of imbalanced classes in satellite imagery using adaptive uncertainty weighting-class loss,” in Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS), Jul. 2018, pp. 6191–6194.
[14] P. O. Bressan et al., “Semantic segmentation with labeling uncertainty and class imbalance applied to vegetation mapping,” Int. J. Appl. Earth Observ. Geoinf., vol. 108, Apr. 2022, Art. no. 102690.
[15] M. G. Atigh, J. Schoep, E. Acar, N. Van Noord, and P. Mettes, “Hyperbolic image segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 4443–4452.
[16] J. Yan, L. Luo, C. Deng, and H. Huang, “Unsupervised hyperbolic metric learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 12465–12474.
[17] N. Linial, E. London, and Y. Rabinovich, “The geometry of graphs and its algorithmic applications,” Combinatorica, vol. 15, no. 2, pp. 197–212, Apr. 1995.
[18] Y. Lyu, G. Vosselman, G.-S. Xia, A. Yilmaz, and M. Y. Yang, “UAVisD: A semantic segmentation dataset for UAV imagery,” ISPRS J. Photogramm. Remote Sens., vol. 165, pp. 108–119, Jul. 2020.
[19] G. Neuhold, T. Ollmann, S. R. Bulo, and P. Kontschieder, “The mapillary vistas dataset for semantic understanding of street scenes,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 4990–4999.
[20] E. Andre et al., “BD100K: A diverse driving dataset for heterogeneous multitask learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 2636–2645.
[21] C. Sakaridis, D. Dai, and L. Van Gool, “ACDC: The adverse conditions dataset with correspondences for semantic driving scene understanding,” in Proc. Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 12465–12474.
[22] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, “SegFormer: Simple and efficient design for semantic segmentation with transformers,” in Proc. Adv. Neural Inf. Process. Syst., vol. 34, 2021, pp. 12077–12090.
[23] M. Fan et al., “Rethinking BiSeNet for real-time semantic segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 9716–9725.
[24] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, “BiSeNet V2: A hybrid network with guided aggregation for real-time semantic segmentation,” Int. J. Comput. Vis., vol. 130, pp. 3051–3068, Sep. 2021.
[25] A. Tao, K. Sapra, and B. Catanzaro, “Hierarchical multi-scale attention for semantic segmentation,” 2020, arXiv:2005.13081.
[26] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, “A ConvNet for the 21st century,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2022, pp. 11966–11976.
[27] Z. Liu et al., “Swin Transformer: Hierarchical vision transformer using shifted windows,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2021, pp. 10012–10022.
[28] J. Zhang, K. Yang, A. Constantinescu, K. Peng, K. Müller, and R. Stiefelhagen, “Trans4Trans: Efficient transformer for transparent object- and semantic-aware segmentation in real-world navigation assistance,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 10, pp. 19173–19186, Oct. 2022.
Bike Chen received the M.S. degree in pattern recognition and intelligent systems from the Nanjing University of Science and Technology, China, in 2018. He is currently pursuing the Ph.D. degree with the Biomimetics and Intelligent Systems Group (BISG), University of Oulu, Finland. He has published technical papers at prominent conferences and journals, such as International Joint Conference on Artificial Intelligence (IJCAI) and IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS. His current research interests include deep learning, image segmentation, 3D point cloud segmentation, simultaneous localization and mapping (SLAM), unmanned aerial vehicle (UAV), and robotics.

Wei Peng received the M.S. degree in computer science from Xiamen University, China, in 2016, and the Ph.D. degree from the Center for Machine Vision and Signal Analysis, University of Oulu, Finland, in 2022. He is currently a Post-Doctoral Researcher with the Department of Psychiatry and Behavioral Sciences of Stanford Medicine, Stanford University, USA. His paper have published in mainstream journals and conferences, such as IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON IMAGE PROCESSING, CVPR, AAAI, ICCV, and ACM Multimedia. His current research interests include machine learning, affective computing, medical imaging, and human action analysis.

Xiaofeng Cao (Member, IEEE) received the Ph.D. degree from the Australian Artificial Intelligence Institute, University of Technology Sydney, Australia. He is currently an Associate Professor with the School of Artificial Intelligence, Jilin University, China. He is also leading the Machine Perceptron Research Group with 20 Ph.D. and master’s students. He has published more than 20 technical papers in top tier journals and conferences, such as IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, ICML, CVPR, and IJCAI. His current research interests include PAC learning theory, agnostic learning algorithm, and generalization analysis.

Juha Röning was a Visiting Research Scientist with the Center for Robotic Research, University of Cincinnati via Asla/Fulbright Scholarship, from 1985 to 1986. From 1986 to 1989, he held a Young Researcher Position with the Finnish Academy. He is currently a Professor of embedded systems with the University of Oulu and a Visiting Professor of the Tianjin University of Technology, China. He is also the Principal Investigator of the Biomimetics and Intelligent Systems Group (BISG). He is serving as the Board of Director for euRobotics aisbl (Vice-President Research) and Adra. He was the Academic Coordinator of the DIMECC CyberTrust Program and the Project Coordinator for the H2020 HYFLIERS Project and the CS-AWARE Project. He is the Project Coordinator of CS-AWARE-NEXT Project (Horizon Europe). He has three patents and has published more than 400 articles in the areas of computer vision, robotics, intelligent signal analysis, and software security. In 2000, he was nominated as a fellow of SPIE.