Exploring Cross-Point Embeddings for 3D Dense Uncertainty Estimation

Kaiwen Cai∗, Chris Xiaoxuan Lu†, Xiaowei Huang∗
∗University of Liverpool, †University of Edinburgh

Abstract— Dense prediction tasks are common for 3D point clouds, but the inherent uncertainties in massive points and their embeddings have long been ignored. In this work, we present CUE, a novel uncertainty estimation method for dense prediction tasks of 3D point clouds. Inspired by metric learning, the key idea of CUE is to explore cross-point embeddings upon a conventional dense prediction pipeline. Specifically, CUE involves building a probabilistic embedding model and then enforcing metric alignments of massive points in the embedding space. We demonstrate that CUE is a generic and effective tool for dense uncertainty estimation of 3D point clouds in two different tasks: (1) in 3D geometric feature learning we for the first time obtain well-calibrated dense uncertainty, and (2) in semantic segmentation we reduce uncertainty’s Expected Calibration Error of the state-of-the-arts by 43.8%. All uncertainties are estimated without compromising predictive performance.

I. INTRODUCTION

Dense prediction refers to the process of predicting the label for each point in a point cloud. It is widely known that dense prediction plays a pivotal role in 3D robotic perception and autonomy, enabling an array of tasks such as semantic segmentation, depth completion, and scene flow estimation.

UNet [1] based network has been the de-facto choice of today’s point neural network architecture design for cloud dense prediction [2]–[4]. In a UNet-like network, one can observe that the input and the output of two correspondingly linked layers have the same number of points, e.g., if the input point cloud is denoted by a $N \times 3$ tensor, then the output of its correspondingly linked layer is a $N \times D$ tensor. In this regard, the output can also be viewed as an embedding map, and a dense prediction network can then be decomposed as an embedding learning network and a task-specific regressor (or classifier). Therefore, the heart of the dense prediction task is embedding learning.

Embedding learning aims to learn a discriminative embedding model that pulls samples of the same class closer and pushes those of different classes away from each other in the embedding space. Successful embedding learning empowers many downstream tasks, including image retrieval [5], face recognition [6] and zero-shot learning [7]. In addition to improving the embedding model’s discriminative capability, quantifying its uncertainty is also attracting much attention.

For dense prediction tasks of point clouds, it is desirable that an uncertainty level could be provided in conjunction with the point-wise labels to make its downstream decision-making more information-aware. Consider a scenario where an autonomous vehicle is predicting semantic labels of each point on the road, a prediction with an estimated uncertainty level would be helpful for the computer to decide when to trust its prediction and moreover, utilize the uncertainty to optimize the vehicle’s planning and control. Such promising benefits have stimulated the development of various uncertainty estimation methods for different dense prediction tasks. In 3D semantic segmentation tasks, for example, the popular approaches include (1) using the output of the logit layer to calculate softmax entropy [8], (2) building a two-head network to predict the mean and variance of an embedding separately [9], and (3) resorting to a BNN model and approximating posterior weights with MCD [10].

However, two major issues remain in existing uncertainty estimation methods for dense prediction of 3D point clouds. First, points can only interact in the limited receptive field of
convolution kernels, and they need a shared MLP to realize an implicit interaction among logits (see Fig. 2.a). Such under-treatment of cross-point dependencies, unfortunately, often results in sub-optimal uncertainty estimation as evidenced by [11]. Second, a notable trait of the predominant dense prediction networks is that they are sequential compositions of embedding learning networks and task-specific regressors (or classifiers). While prior arts have shown that enforcing embedding learning in regression or classification tasks can yield better predictive performance [12], [13], it is largely under-explored if utilizing embedding learning can also give rise to better-calibrated uncertainty.

In this paper, we propose a novel and generic uncertainty estimation pipeline, called CUE in the paper for Cross-point embedding Uncertainty Estimation, to bridge the gap between the dense prediction of point clouds and its dense uncertainty quantification. CUE involves building a probabilistic embedding model and enforcing metric alignments of massive points in the embedding space. In view of the aforementioned issues, CUE identifies the importance of embedding learning, and exploits this embedding space via a diagonal multivariate Gaussian model amenable to cross-point interactions. Moreover, we propose CUE+ that further utilizes cross-point dependencies by a low-rank multivariate Gaussian model. Low-rank covariance matrix in CUE+ explicitly expresses off-diagonal elements' dependencies while maintaining computational efficiency. Specifically, our contributions are stated as follows:

- For the first time we propose a generic dense uncertainty estimation framework for dense prediction tasks of 3D point clouds.
- We propose a novel method that fully explores cross-point information for dense uncertainty estimation.
- We validate our proposed method on two representative dense prediction tasks, with the experimental results consistently showing that our method produces better-calibrated uncertainty than state-of-the-arts without losing any predictive performance.
- Source code of both CUE and CUE+ is available at: https://github.com/ramdrop/cue.

II. RELATED WORK

A. Dense Prediction of 3D Point Cloud

With the dense nature of the 3D point cloud, we focus on its dense prediction tasks, e.g., 3D geometric feature learning and 3D semantic segmentation.

3D Geometric Features Learning: To find the correspondences in the absence of relative transformation information, a series of methods is to convert point clouds from the 3D Euclidean space to a feature space, where the correspondences are the nearest neighbors. Early work focus on hand-crafted features, such as SHOT [14] and FPFH [15], we kindly refer readers to [16] for more details about hand-crafted features.

Recently deep learned geometric features are becoming popular, which are generally based on volumetric and point-wise operations on point clouds: (1) Volumetric: 3DMatch [17] learns patch descriptors by applying a 3D convolutional neural network on volumetric input. FCGF [2] directly applies 3D CNN to volumetric point clouds with the hardest contrastive loss, generating dense point features. (2) Point-wise: PointNet [18] uses multiple parallel shared MLP to learn global or dense features. DGCNN [19] combines point-wise MLP with dynamic graph neural networks, obtaining flexible and effective feature extractors for unordered point clouds. SpinNet [3] proposes a reference axis with a spherical voxelization to learn viewpoint-invariant point descriptors. Nevertheless, the above methods focus on improving predictive performance while ignoring the inherent uncertainty in massive points.

3D Semantic Segmentation: PointNet [18] is the very first work for 3D point cloud learning, and its shared-MLP architecture shows strong representation capability. However, the perturbation invariance of point clouds is obtained at the cost of ignorance of the local context. Following works propose different solutions to make for this limitation: PointNet++ [20] adopts hierarchical sampling strategies, KPConv [4] proposes a kernel-based MLP operation mimicking convolution, MinkowskiNet [21] extends 2D convolution to 3D voxel and specifically design sparse operation python library for point clouds, and recently PointTransformer [22] shows the power of Transformer mechanisms in point cloud processing.

B. Dense Uncertainty Estimation

Embedding Learning Uncertainty: Kendall [9] categorizes uncertainties in deep learning as two types: aleatoric uncertainty and epistemic uncertainty. Aleatoric uncertainty stems from data noises, while epistemic uncertainty refers to model uncertainty, which can be reduced with sufficient training data. Embedding learning is usually applied to image recognition tasks, where most methods focus on estimating aleatoric uncertainty: PFE [23] models face embeddings as Gaussian distributions and uses the proposed Mutual Likelihood Score to measure the likelihood of two embeddings belonging to the same class. DUL [24] proposes to learn aleatoric uncertainty for both regression and classification face recognition tasks. BTL [25] proposes a Bayesian loss to learn aleatoric uncertainty in place recognition. RUL [26] uses relative uncertainty measurements to learn aleatoric uncertainty.

In the above image recognition tasks, a single feature is learned for a whole image. But in the dense prediction task of the point cloud, a single point cloud will involve learning thousands of features (i.e., equals to the number of points in the point cloud). Furthermore, image recognition is applied to regular-size images, while point clouds are totally unordered and varied-size. The massive features within a batch and irregular input size render it rather challenging to estimate dense uncertainty for a 3D point cloud.

Semantic Segmentation Uncertainty: Popular uncertainty estimation methods for semantic segmentation include soft-max entropy [8], Bayesian Neural Network (BNN) [27], learned aleatoric uncertainty [9], auxiliary network [28] and variance propagation based on Assumed Density Function
(ADF) [29]. Please refer to [30] for a thorough overview. However, these dominant approaches for semantic segmentation usually treat pixels or points as independent of each other (see Fig. 2.a). Such ignorance of cross-pixel or cross-point dependencies tends to result in noisy uncertainty estimation [11].

Embedding learning has been explored in image segmentation: [13] and [31] show contrastive learning optimizes embedding space and improve prediction performance in a semantic segmentation task, [12] proves that optimized embeddings contribute to predictive performance. However, all the above methods exploit embedding learning for improving predictive performance, rather than estimating dense uncertainty. SSN [11] has used a low-rank multivariate Gaussian model to account for cross-pixel dependencies. But it is developed for logits, which does not involve embedding optimization.

Our CUE is based on a probabilistic embedding model and enforces metric alignments in the embedding space by using bayesian triplet loss. Bayesian triplet loss has been used in [25] in image recognition. The major differences are: (1) the image recognition [25] requires a single embedding for an image (i.e., whole pixels), while massive point-wise embeddings are desired in CUE. Thus, we design additional sophisticated sampling strategies and efficient networks for unordered point clouds; and (2) the probabilistic embedding model of [25] ignores the cross-point dependencies. Thus, we propose CUE+ to alleviate this issue by a low-rank multivariate Gaussian model.

III. METHOD

A. Preliminary

A dense prediction network maps a batch of points to a set of scalars. The process can be decomposed into a metric learning phase and a task-oriented regression or classification phase. Formally, given a point cloud \( P \in \mathbb{R}^{N \times 3} \), the network \( f_0 \) first maps it to a set of embeddings \( \mathcal{X} \in \mathbb{R}^{N \times D} \), where \( N \) is the number of points, and \( D \) is the embedding dimension:

\[
\text{metric learning} : \mathcal{X} = f_0(P)
\]  (1)

which is followed by a task-oriented regressor (or classifier) \( f_r \) that generates predictions \( Y \in \mathbb{R}^{N \times 1} \) (or \( Y \in \mathbb{R}^{N \times C} \) where \( C \) is the number of class ) for the set of embedding:

\[
\text{regression or classification} : Y = f_r(\mathcal{X})
\]  (2)

In the above formulation, predictions are regarded as deterministic, while the inherent noise from data is ignored. A probabilistic prediction model (e.g., probabilistic semantic segmentation [9]) casts the prediction as a Gaussian distribution, which provides uncertainty level along with prediction (See Fig. 2.a). But embeddings are still deterministic and equally weighted, which means each embedding will contribute equally to the regressor (or classifier).

Inspired by probabilistic contrastive learning in face recognition [23], [24], we adopt a probabilistic embedding model for a point cloud, where embeddings are represented by a diagonal multivariate Gaussian distribution:

\[
\mathcal{X} \sim \mathcal{N}(\mu, \Lambda^2)
\]  (3)

where \( \mu = f_\mu(P) \in \mathbb{R}^{N \times D} \) and \( \Lambda^2 = f_\sigma(P) \in \mathbb{R}^{(N \times D) \times (N \times D)} \) is a diagonal matrix. \( f_\mu \) and \( f_\sigma \) represents the mean branch and variance branch of the network \( f_0 \). We will later propose a full-covariance multivariate Gaussian model and show its superiority in Sec. III-B.2.

B. Exploring Cross-point Embeddings

After building the probabilistic embedding model, we now discuss how to optimize the embedding space and derive the uncertainty.

An overview of a traditional probabilistic prediction pipeline, the proposed CUE and CUE+ is presented in Fig. 2. A traditional probabilistic prediction only allows logits to interact implicitly through a shared MLP, while CUE explores cross-point embeddings by building a probabilistic embedding model and enforcing metric alignments, and CUE+ goes further by replacing the diagonal covariance matrix with a low-rank covariance matrix. In what follows we will first describe CUE that is based on the diagonal multivariate Gaussian model. Then an improved version
CUE+ will be introduced, which is based on the low-rank multivariate Gaussian model.

1) CUE: Given a triplet \( \{P_a, P_p, P_n\} \in \mathbb{R}^{1 \times 3}, i = a, p, n \), their embeddings are obtained as \( \{X_a, X_p, X_n\} | \mathcal{X} \sim \mathcal{N}(\mu, \Sigma_d^2), \mu_i \in \mathbb{R}^{1 \times D}, \Sigma^2_i \in \mathbb{R}^{1 \times D}, i = a, p, n \), where the subscripts \( a, p, n \) denote an anchor, positive and negative sample, respectively. In the probabilistic setting, we are interested in the probability of the positive embedding being closer than the negative to the anchor:

\[
P(\|X_a - X_p\| - \|X_a - X_n\| + m < 0) \tag{4}
\]

Rewrite it as:

\[
P(\tau < -m) = \Phi_{\mathcal{N}(0,1)}(-m - \frac{\mu \tau}{\sigma_\tau}) \tag{6}
\]

where \( \Phi \) is the Conditional Density Function (CDF). Now the task is converted to finding an analytical solution of \( \mu_\tau \) and \( \sigma_\tau \). The mean \( \mathbb{E}[\tau] \) and variance \( \mathbb{D}[\tau] \) of a single dimension is given as follows (the superscript \( d \) at right-hand side is omitted for brevity):

\[
\mathbb{E}[T^d] = \mu_p^2 + \sigma_p^2 - \mu_n^2 - \sigma_n^2 - 2\mu_a (\mu_p - \mu_n)
\]

\[
\mathbb{D}[T^d] = 2(\sigma_p^4 + 2\mu_p^2\sigma_p^2 + 2(\sigma_a^2 + \mu_a^2)(\sigma_p^2 + \mu_p^2) - 2\mu_a^2\mu_p^2 - 4\mu_a\mu_p\sigma_p^2 + 2[\sigma_n^4 + 2\mu_n^2\sigma_n^2 + 2(\sigma_a^2 + \mu_a^2)(\sigma_n^2 + \mu_n^2)] - 2\mu_a^2\mu_n^2 - 4\mu_a\mu_n\sigma_n^2) + 4\mu_a\mu_n\sigma_a^2
\]

Since the embedding model is assumed to be isotropic, we arrive at:

\[
\mu_\tau = \sum_d \mathbb{E}[T^d], \quad \sigma_\tau^2 = \sum_d \mathbb{D}[T^d] \tag{8}
\]

In summary, after the network generates a set of embeddings for a point cloud, we calculate the probability of the positive embedding being closer to the negative than to the anchor, and the goal of training is to minimize the metric loss derived from Eq. 4:

\[
L_M = -\frac{1}{T} \sum_{t=1}^{T} P(\|X_{t,a} - X_{t,p}\| - \|X_{t,a} - X_{t,n}\| + m < 0) \tag{9}
\]

where \( T \) is the number of total triplets in a mini-batch.

2) CUE+: Points usually show spatial correlation with their neighbors. For example, points at the boundaries of an object usually exhibit high uncertainty since the points around the boundary have varied semantic labels. But CUE fails to model point-wise dependencies because the diagonal covariance matrix of CUE (see Eq. 3) is based on the assumption that points are independent of each other. To solve this issue, we propose further capturing the point-wise dependencies by a full-covariance multivariate Gaussian model. Specifically, the diagonal covariance matrix in Eq. 3 is replaced with a full covariance matrix \( \Sigma^2 \in \mathbb{R}^{(N \times D) \times (N \times D)} \):

\[
\mathcal{X} \sim \mathcal{N}(\mu, \Sigma^2) \tag{10}
\]

where \( \mu \in \mathbb{R}^{N \times D} \). However, the computational complexity of the full covariance matrix \( \Sigma^2 \) scales with the square of \( N \), and a point cloud usually consists of tens of thousands of points, i.e., \( N > 10^4 \). This makes training networks difficult. To alleviate this issue, we resort to a low-rank parameterization of the covariance matrix [32]:

\[
\Sigma^2 = PP^T + \Lambda^2 \tag{11}
\]

where the scale factor \( P \in \mathbb{R}^{(N \times D) \times K} \) and \( K \) is the rank of the parameterization, \( \Lambda^2 \in \mathbb{R}^{(N \times D) \times (N \times D)} \) and \( \Lambda^2 \) is a diagonal matrix. Therefore, we refer to the pipeline based on a low-rank covariance matrix as CUE+. Compared with CUE, CUE+ learns parameters of additional elements other than diagonal elements of the covariance matrix. This makes the point-wise dependencies explicitly described by the learned variances.

For ease of application, we choose \( K = 1 \). Then the equivalent of the embedding \( \mathcal{X} \) is obtained as

\[
\mathcal{X} = \mu + (P + \Lambda) \cdot \mathcal{N}(0, I) \tag{12}
\]

\( L_M \) is then used to train the network. By experimental results we show that CUE+ generates better-calibrated uncertainty than CUE (see Sec. IV).

The network architectures of the proposed CUE and CUE+ are shown in Fig. 3, where \( \mathcal{P} \) means a 3D point cloud. The backbone encoder and decoder can be chosen from any UNet-like network. We add three branches to predict the mean \( \mu \), diagonal covariance matrix \( \Lambda^2 \) and the scale factor.
$P, \mu$ branch ends with an L2-Normalization layer, while $\Lambda^2$ and $P$ branches with softplus layers.

IV. EXPERIMENTAL RESULTS

A. 3D Geometric Feature Learning

While CUE and CUE+ are generic to dense prediction tasks, sampling strategies for triplets should be adapted according to different downstream tasks. Here, we present practical sampling strategies for two different tasks: 3D geometric feature learning and 3D semantic segmentation.

3D geometric feature learning aims to learn a discriminative mapping function represented by a deep neural network, such that raw points in the Euclidean space are mapped to the feature space. Ideally, points with similar geometric characteristics should be close to each other in the feature space. [2] studies different sampling strategies, including hardest-triplet sampling and random triplet sampling, where triplet loss is adopted. We follow their sampling methods but adapt their conventional triplet loss to our metric loss $L_M$. Specifically, given point clouds $P_i$ and $P_j$ and the relative transformation $T$, we first sample anchor embeddings $X_{i,a}$ and $X_{j,a}$. Then, we randomly choose its positives $X_{i,p}, X_{j,p}$ and negatives $X_{i,n}, X_{j,n}$. Finally, we calculate the metric loss $L_M$ of the triplets $\{X_{i,a}, X_{i,p}, X_{i,n}\}$ and $\{X_{j,a}, X_{j,p}, X_{j,n}\}$ for training.

Datasets. We use the 3D Match dataset [17], following the official training and evaluation splits.

Model Architectures. FCGF is the first 3D convolutional network to integrate metric learning in a fully-convolutional setting. We choose FCGF [2] as our backbone because it holds state-of-the-art predictive performance with fast training and inferencing. To empower the deterministic FCGF to estimate the uncertainty of each point, we integrate it with our proposed CUE and CUE+ as is shown in Fig. 3.

Training Details. We train FCGF following the original paper [2], i.e., Hardest-contrastive loss, 100 epochs with SGD optimizer and batch size 4, learning rate starts from $0.1$ with exponential decay rate $0.99$, data augmentation includes random scaling $\in [0.8, 1.2]$ and random rotation $\in [0^\circ, 360^\circ]$.

Competing Methods.

- RG: After training the FCGF, we randomly form ten bins of points and then calculate ECE.
- MCD: We insert dropout layers with dropout rate $p = 0.1$ after every convolutional layer. We take $N = 40$ samples from the weights’ posterior distribution at test time.
- CUE: To assure original predictive performance, we freeze the $\mu$ branch and train $\Lambda^2$ branches with the

### Table I

| Method      | FMR@0.05 ↑ | ECE ↓ |
|-------------|------------|-------|
| FPFH∗ [15]  | 36.4       |       |
| PerfectMatch∗ [33] | 94.9       |       |
| FCGF∗ [2]   | 95.3       |       |
| SpinNet [3] | 97.5       |       |
| FCGF [2]    | 97.5       |       |
| FCGF+RG     | 97.5       | 0.251 |
| FCGF+MCD    | 94.1       | 0.344 |
| FCGF+CUE    | 97.5       | 0.142 |
| FCGF+CUE+   | 97.6       | 0.135 |

* denotes predicting correspondences without a symmetric test [34].
metric loss $L_M$.

- **CUE+**: We freeze the $\mu$ branch, and train $A^2$ and $P$ branches with the metric loss $L_M$.

Note that MCD produces epistemic uncertainty, while our methods generate aleatoric uncertainty. We include it here for a comprehensive comparison.

**Evaluation Metrics.** To evaluate the predictive performance, we use Feature Matching Recall with 0.1 inlier distance threshold and 0.05 inlier recall threshold (FMR@0.05) [2]. We adopt the widely used Expected Calibrated Error (ECE) [25] and the reliability diagram [25] to evaluate uncertainty quality, where we calculate the Hit Ratio [2] of points in the same bin.

**Results.** We evaluate the above methods on the 3D Match Benchmark [17]. We establish correspondences by the nearest neighbor search in the embedding space, where each correspondence has an estimated uncertainty.\(^1\) Table. IV-A shows the predictive performance and uncertainty quality of different methods on the 3DMatch dataset. MCD shows degraded predictive performance due to the dropout layers significantly harming the network’s representation ability. Since the $\mu$ branch is inherited from the backbone network, CUE and CUE+ do not sacrifice any predictive accuracy. Compared with MCD, CUE reduces ECE by 0.202. Besides, CUE+ outperforms CUE with ECE 0.135.

Fig. 4 illustrates the reliability diagram on the 3D Match Benchmark. The ideal line means points with higher uncertainty levels should have lower hit ratios. RG produces a horizontal line, while MCD fails to produce a sensible estimation. CUE and CUE+ present closer lines to the ideal line. Fig. 5 shows the matching results and dense uncertainty map estimated by CUE+ of a point cloud. We can observe that incorrect correspondences (1 and 2 areas) tend to have high uncertainties.

In summary, the proposed CUE and CUE+ provide well-calibrated uncertainty that can be used as an effective tool to filter incorrect correspondence.

**B. 3D Semantic Segmentation**

3D semantic segmentation aims to learn a classification network that predicts class labels for each point in a point cloud. To optimize the embedding space by enforcing metric alignments, we first randomly sample anchors from a point cloud. To optimize the embedding space by enforcing metric network that predicts class labels for each point in a point cloud estimated by CUE+ of a point cloud. We can observe that incorrect correspondences (1 and 2 areas) tend to have high uncertainties.

In summary, the proposed CUE and CUE+ provide well-calibrated uncertainty that can be used as an effective tool to filter incorrect correspondence.

Finally, we calculate the metric loss $L_M$ of the triplet $(X_a, X_p, X_n)$ for training.

**Datasets.** Following [35], we use the ScanNet dataset [36] and evaluate models on the ScanNet validation split.

**Model Architectures.** Considering inference latency and accuracy, we choose MinkowskiNet42 (Mink) [21], [35] as our 3D semantic segmentation backbone. The semantic segmentation network is the same as that in Fig. 3, except that we add a convolution layer as the segmentation classifier before the L2-Normalization layer of the $\mu$ branch.

**Training Details.** We train the model for $10^5$ steps with an SGD optimizer, learning rate starting from 0.1 with a cosine annealing schedule and a linear warmup. We use a batch size of 8. For more training details, we kindly refer readers to [35].

**Competing Methods.** We compare CUE and CUE+ with the following popular uncertainty estimation methods from image segmentation:

- **Softmax Entropy [30] (SE):**

  \[
  H = - \sum_c p_c \log(p_c)/\log(C) \in [0, 1] \tag{13}
  \]

  where $C$ is the number of classes, $p_c$ is a probability by the softmax layer.

- **Aleatoric Uncertainty [9], [30] (AU):** Logs are modeled as Gaussian distribution, whose mean and variance are predicted by two heads of the network. We use
MC sampling ($n = 10$) to draw samples from the logits distribution and optimize the network with Cross Entropy Loss.

- MCD [30]: MCD estimates epistemic uncertainty because dropout at test time approximates random sampling of the model’s weights. Test time inference is obtained by $p_c = \frac{1}{N} \sum_{n}^N p_{n,c}$, where $p_c$ denotes the output of the Softmax layer. We set the number of MC samples $N = 40$ as suggested by [39]. We evaluate MCD with two dropout probability settings: $p = 0.2$ and $p = 0.05$. Aleatoric uncertainty and MCD uncertainty generates high-dimensional variance vectors, which are converted to uncertainty levels by $y(1 - 0.5q) + (1 - y)(0.5q)$, where $q \in [0, 1]$ is the normalized variance [30].

- CUE / CUE+: We train the CUE / CUE+ network from scratch with a weighted sum of cross entropy loss and the metric loss $L = L_{CH} + \lambda L_M$, where we set $\lambda = 1$ for all experiments.

**Evaluation Metrics.** We use the mean IOU (mIOU) to evaluate the predictive performance. mIOU refers to the ratio of the intersection of ground-truth labels and predicted labels to their union, and a higher mIOU indicates better performance. The reliability diagram [25] and ECE [25] are adopted to evaluate uncertainty quality, where we calculate the precision of points in each bin.

**Results.** Table IV-B presents the predictive performance and uncertainty quality on the ScanNet validation split. In terms of predictive performance, we observe that AU, CUE produce comparable results to Mink, while MCD shows degraded performance due to dropout layers decreasing representative power. CUE+ promotes the Mink’s predictive power with a 0.13 boost in mIOU. From the perspective of uncertainty quality, SE achieves a 0.251 ECE, outperformed by MCD ($p=0.05$) with a 0.170 ECE, while CUE and CUE+ provide significantly improved uncertainty with ECE 0.142 and 0.141, i.e., CUE+ reduces ECE of SE by 43.8%.

Fig. 6 shows the reliability diagram on the ScanNet validation split. It can be seen that CUE is close to the ideal calibrated line, while CUE+ improves CUE in the low-uncertainty region. Fig. 1 presents the qualitative results of CUE+, where we can observe a significant correlation between segmentation prediction error and estimated uncertainty, i.e., Incorrect predictions tend to have high uncertainties.

Fig. 7 presents segmentation errors and dense uncertainty maps by different methods on the ScanNet validation split. For incorrect predictions (black points in the magnified area), we can observe that SE fails to detect them and shows high confidence, while CUE and CUE+ are uncertain about those incorrect predictions. AU is under-confident in most areas, while MCD ($p=0.05$) cannot produce sensible results. For correct predictions (Rectangular area 1), CUE is under-confident while CUE+ is more confident than CUE.

The above results indicate that CUE and CUE+ provide better-calibrated uncertainty than existing methods without compromising any predictive performance. CUE+ outperforms CUE in both predictive performance and uncertainty quality. This shows that embedding learning contributes to uncertainty estimation of dense prediction tasks and low-rank multivariate Gaussian model is more effective than a diagonal one.

**V. Conclusion**

Observing the fact that dense prediction networks are sequential compositions of embedding learning networks and task-specific regressors (or classifiers), we propose CUE that estimates dense uncertainty by building a probabilistic embedding model and enforcing metric alignments with a diagonal multivariate Gaussian model. Besides, we propose CUE+ that further enhances cross-point interactions with a low-rank multivariate Gaussian model, which explicitly expresses off-diagonal elements’ dependencies while maintaining computational efficiency. Experimental results on the 3D Match Benchmark and the ScanNet dataset prove that CUE and CUE+ are generic and effective tools for 3D dense uncertainty estimation.
References

[1] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in International Conference on Medical image computing and computer-assisted intervention. Springer, 2015, pp. 234–241.

[2] C. R. Qi, Y. Li, and V. Koltun, “Fully convolutional geometric features,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 8958–8966.

[3] S. Ao, Q. Hu, B. Yang, A. Markham, and Y. Guo, “Spinnet: Learning a general surface descriptor for 3d point cloud registration,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 11 753–11 762.

[4] H. Thomas, C. R. Qi, J. E. Deschaud, V. Marcotegui, F. Goulette, and L. J. Guibas, “Kpconv: Flexible and deformable convolution for point clouds,” in Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 6411–6420.

[5] K. Musgrave, S. Belongie, and S.-N. Lim, “A metric learning reality check,” in European Conference on Computer Vision. Springer, 2020, pp. 681–699.

[6] Q. Meng, S. Zhao, Z. Huang, and F. Zhou, “Magface: A universal representation for face recognition and quality assessment,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 14 225–14 234.

[7] M. Bucher, S. Herbin, and F. Jurie, “Improving semantic embedding consistency by metric learning for zero-shot classification,” in European Conference on Computer Vision. Springer, 2016, pp. 730–746.

[8] S. Crooke, K. Arnavaz, O. Krause, and A. Feragen, “Is segmentation uncertainty useful?” in International Conference on Information Processing in Medical Imaging. Springer, 2021, pp. 715–726.

[9] A. Kendall and Y. Gal, “What uncertainties do we need in bayesian deep learning for computer vision?” Advances in neural information processing systems, vol. 30, 2017.

[10] C. Qi, J. Yin, H. Liu, and J. Liu, “Neighborhood spatial aggregation based efficient uncertainty estimation for point cloud semantic segmentation,” in 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021, pp. 14 025–14 031.

[11] M. Monteiro, L. Le Folgoc, D. Coelho de Castro, N. Pawlowski, B. Marques, K. Kamitats, M. van der Wilk, and B. Glocker, “Stochastic segmentation networks: Modelling spatially correlated aleatoric uncertainty,” Advances in Neural Information Processing Systems, vol. 33, pp. 12 756–12 767, 2020.

[12] W. Li, X. Huang, J. Lu, J. Feng, and J. Zhou, “Learning probabilistic ordinal embeddings for uncertainty-aware regression,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 13 896–13 905.

[13] W. Wang, T. Zhou, F. Yu, J. Dai, E. Konukoglu, and L. Van Gool, “Exploring cross-image pixel contrast for semantic segmentation,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 7303–7313.

[14] S. Salti, F. Tombari, and L. Di Stefano, “Shot: Unique signatures of histograms for surface and texture description,” Computer Vision and Image Understanding, vol. 125, pp. 251–264, 2014.

[15] R. B. Rusu, N. Blodow, and M. Beetz, “Fast point feature histograms (fpfh) for 3d registration,” in 2009 IEEE international conference on robotics and automation, IEEE, 2009, pp. 3212–3217.

[16] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, J. Wan, and N. M. Kwok, “A comprehensive performance evaluation of 3d local feature descriptors,” International Journal of Computer Vision, vol. 116, no. 1, pp. 66–89, 2016.

[17] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, “3dmatch: Learning local geometric descriptors from rgbd reconstructions,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1802–1811.

[18] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 652–660.

[19] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, “Dynamic graph cnn for learning on point clouds,” Acm Transactions On Graphics (tog), vol. 38, no. 5, pp. 1–12, 2019.

[20] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” Advances in neural information processing systems, vol. 30, 2017.

[21] C. Choy, J. Gwak, and S. Savarese, “4d spatio-temporal convnets: Minkowski convolutional neural networks,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 3075–3084.

[22] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, “Point transformer,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 16 259–16 268.

[23] Y. Shi and A. K. Jain, “Probabilistic face embeddings,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6902–6911.

[24] J. Chang, Z. Lan, C. Cheng, and Y. Wei, “Data uncertainty learning in face recognition,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 5710–5719.

[25] F. Warburg, M. Jørgensen, C. Jivera, and S. Hauberg, “Bayesian triplet loss: Uncertainty quantification in image retrieval,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 12 158–12 168.

[26] Y. Zhang, C. Wang, and W. Deng, “Relative uncertainty learning for facial expression recognition,” Advances in Neural Information Processing Systems, vol. 34, 2021.

[27] A. Kendall, V. Badrinarayanan, and R. Cipolla, “Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding,” arXiv preprint arXiv:1511.02680, 2015.

[28] Z. Zheng and Y. Yang, “Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation,” International Journal of Computer Vision, vol. 129, no. 4, pp. 1106–1120, 2021.

[29] T. Cortinhal, G. Tzelepis, and E. Erdal Aksoy, “Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds,” in International Symposium on Visual Computing. Springer, 2020, pp. 207–222.

[30] A. Jungo and M. Reyes, “Assessing reliability and challenges of uncertainty estimations for medical image segmentation,” in International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, 2019, pp. 48–56.

[31] L. Tang, Y. Zhan, Z. Chen, B. Yu, and D. Tao, “Contrastive boundary learning for point cloud segmentation,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 8489–8499.

[32] M. Magdon-Ismail and J. T. Purnell, “Approximating the covariance matrix of gmms with low-rank perturbations,” in International Conference on Intelligent Data Engineering and Automated Learning. Springer, 2010, pp. 300–307.

[33] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, “The perfect match: 3d point cloud matching with smoothed densities,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 5545–5554.

[34] L. Tang, Y. Zhan, Z. Chen, B. Yu, and D. Tao, “Contrastive boundary learning for point cloud segmentation,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 16 949–16 958.

[35] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Niethammer, “Scannet: Richly-annotated 3d reconstructions of indoor scenes,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 5828–5839.

[36] W. Wu, Z. Qi, and L. Fuxin, “Pointconv: Deep convolutional networks on 3d point clouds,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 9621–9630.

[37] B. Graham, M. Engelcke, and L. Van Der Maaten, “3d semantic segmentation with submanifold sparse convolutional networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 9224–9232.

[38] A. Kendall and R. Cipolla, “Modelling uncertainty in deep learning for camera relocalization,” in 2016 IEEE international conference on Robotics and Automation (ICRA). IEEE, 2016, pp. 4762–4769.