PointMixup: Augmentation for Point Cloud

Supplementary Material

Yunlu Chen*[1], Vincent Tao Hu*[1], Efstratios Gavves[1],

Thomas Mensink[2,1], Pascal Mettes[1], Pengwan Yang[1,3], and Cees Snoek[1]

[1] University of Amsterdam
[2] Google Research, Amsterdam
[3] Peking University

1 Proofs for the properties of PointMixup interpolation

We provide detailed proofs for the shortest path property, the assignment invariance property and the linearity, stated in Section 3.4.

Proof for the shortest path property We denote $x_i \in S_1$ and $y_j \in S_2$ are the points in S_1 and S_2, then the generated $S^{(\lambda)}_{\mathbf{OA}} = \{u_i\}_{i=1}^N$ and $u_i = (1-\lambda)\cdot x_i + \lambda \cdot y_{\phi^*(i)}$, where ϕ^* is the optimal assignment from S_1 to S_2.

Then we suppose an identical one-to-one mapping ϕ_I such that $\phi_I(i) = i$. Then by definition of the EMD as the minimum transportation distance, so

$$d_{\text{EMD}}(S_1, S^{(\lambda)}_{\mathbf{OA}}) \leq \frac{1}{N}\sum_i \|x_i - u_{\phi_I(i)}\|_2, \tag{1}$$

where the right term of (1) is the transportation distance under identical assignment ϕ_I. Since $\frac{1}{N}\sum_i \|x_i - u_{\phi_I(i)}\|_2 = \frac{1}{N}\sum_i \|x_i - ((1-\lambda)\cdot x_i + \lambda \cdot y_{\phi^*(i)})\|_2 = \lambda \frac{1}{N}\sum_i \|x_i - y_{\phi^*(i)}\|_2 = \lambda \cdot d_{\text{EMD}}(S_1, S_2)$. Thus,

$$d_{\text{EMD}}(S_1, S^{(\lambda)}_{\mathbf{OA}}) \leq \lambda \cdot d_{\text{EMD}}(S_1, S_2). \tag{2}$$

Similarly as in (1) and (2), the following inequality (3) can be derived by assigning the correspondence from $S^{(\lambda)}_{\mathbf{OA}}$ to S_2 with ϕ^*:

$$d_{\text{EMD}}(S^{(\lambda)}_{\mathbf{OA}}, S_2) \leq (1-\lambda) \cdot d_{\text{EMD}}(S_1, S_2). \tag{3}$$

With (2) and (3),

$$d_{\text{EMD}}(S_1, S^{(\lambda)}_{\mathbf{OA}}) + d_{\text{EMD}}(S_2, S^{(\lambda)}_{\mathbf{OA}}) \leq d_{\text{EMD}}(S_1, S_2). \tag{4}$$

However, as the triangle inequality holds for the EMD, *i.e.*

$$d_{\text{EMD}}(S_1, S^{(\lambda)}_{\mathbf{OA}}) + d_{\text{EMD}}(S_2, S^{(\lambda)}_{\mathbf{OA}}) \geq d_{\text{EMD}}(S_1, S_2), \tag{5}$$

* Equal contribution.

Then by summarizing (4) and (5), $d_{\text{EMD}}(S_1, S^{(\lambda)}_{\mathbf{OA}}) + d_{\text{EMD}}(S_2, S^{(\lambda)}_{\mathbf{OA}})$ $= d_{\text{EMD}}(S_1, S_2)$ is proved. □

Proof for the assignment invariance property We introduce two intermediate arguments. We begin with proving the first intermediate argument: ϕ_I is the optimal assignment from S_1 to $S^{(\lambda_1)}_{\mathbf{OA}}$. Similarly as in (2) ,(3) and (5) from the proof for Proposition 1, in order to allow all the three inequalities, the equal signs need to be taken for all of the three inequalities. Consider that the equal sign being taken for (2) is equivalent to the the equal sign being taken for (1), then,

$$d_{\text{EMD}}(S_1, S^{(\lambda_1)}_{\mathbf{OA}}) = \frac{1}{N}\sum_i \|x_i - u_{\phi_I(i)}\|_2, \tag{6}$$

which in turn means that ϕ_I is the optimal assignment from S_1 to $S^{(\lambda_1)}_{\mathbf{OA}}$ by the definition of the EMD. So the first intermediate argument is proved.

The second intermediate argument is that ϕ^* is the optimal assignment from $S^{(\lambda_1)}_{\mathbf{OA}}$ to S_2. This argument can be proved samely as the first one. Say the equal sign being taken for (3) is equivalent to that

$$d_{\text{EMD}}(S^{(\lambda_1)}_{\mathbf{OA}}, S_2) = \frac{1}{N}\sum_i \|u_i - y_{\phi^*(i)}\|_2. \tag{7}$$

Thus, ϕ^* is the optimal assignment from $S^{(\lambda_1)}_{\mathbf{OA}}$ to S_2 is proved.

Then, with the two intermediate arguments, we can reformalize the setup to regard that $S^{(\lambda_2)}_{\mathbf{OA}}$ is interpolated from source pairs $S^{(\lambda_1)}_{\mathbf{OA}}$ and S_2 with the mix ratio $\frac{\lambda_2-\lambda_1}{1-\lambda_1}$, because the optimal assignment from $S^{(\lambda_1)}_{\mathbf{OA}}$ to S_2 is the same as the optimal assignment from S_1 to S_2. This argument then becomes an isomorphic with respect to the first intermediate argument. Then we prove that ϕ_I is the optimal assignment from $S^{(\lambda_1)}_{\mathbf{OA}}$ to $S^{(\lambda_2)}_{\mathbf{OA}}$ similarly as the proof for the first intermediate argument. □

Proof for linearity We have shown that ϕ_I is optimal assignment between $S^{(\lambda_1)}_{\mathbf{OA}} = \{u_k\} = \{(1-\lambda_1)\cdot x_k + \lambda_1 \cdot y_{\phi^*(k)}\}$ and $S^{(\lambda_2)}_{\mathbf{OA}} = \{v_l\} = \{(1-\lambda_2)\cdot x_l + \lambda_2 \cdot y_{\phi^*(l)}\}$. Thus, $d_{\text{EMD}}(S^{(\lambda_1)}_{\mathbf{OA}}, S^{(\lambda_2)}_{\mathbf{OA}}) = \frac{1}{N}\sum_k \|((1-\lambda_1)\cdot x_k + \lambda_1 \cdot y_{\phi^*(k)}) - ((1-\lambda_2)\cdot x_{\phi_I(k)} + \lambda_2 \cdot y_{\phi^*(\phi_I(k))})\|_2 = \frac{1}{N}\sum_k \|(\lambda_2-\lambda_1)(x_k - y_{\phi^*(k)})\|_2 = (\lambda_2 - \lambda_1)\frac{1}{N}\sum_k \|(x_k - y_{\phi^*(k)})\|_2 = (\lambda_2-\lambda_1)\cdot d_{\text{EMD}}(S_1, S_2)$. □

2 Few-shot learning with PointMixUp

We test if our PointMixup helps point cloud few-shot classification task, where a classifier must generalize to new classes not seen in the training set, given only a small number of examples of each new class. We take ProtoNet [3] as the baseline method for few-shot learning, and PointNet++ [2] is the feature extractor h_θ.

Episodic learning setup ProtoNet takes the episodic training for few-shot learning, where an episode is designed to mimic the few-shot task by subsampling classes as well as data. A N_C-way N_S-shot setting is defined as that in each episode, data from N_C classes are sampled and N_S examples for each class is labelled. In the i^{th} episode of training, the dataset $\mathcal{D}_i$ consists of the training example and class pairs from N_C classes sampled from all training classes. Denote $\mathcal{D}_i^S \subset \mathcal{D}_i$ is the support set which consists of labelled data from N_C classes with N_S examples, and $\mathcal{D}_i^Q = \mathcal{D}_i \backslash \mathcal{D}_i^S$ is the query set which consists of unlabelled examples to be predicted.

Baseline method for few-shot classification: ProtoNet [3] In each episode $\mathcal{D}_i$, ProtoNet computes a prototype as the mean of embedded support examples $\bar{z}_c$ for each class c, from all examples from the support set $\mathcal{D}_i^S$. The latent embedding is from the network h_θ (for which we use PointNet++ [2] without the last fully-connected layer). Then each example S from the query set $\mathcal{D}_i^Q$ is classified into a label distribution by a softmax over (negative) distance to the class prototypes:

$$p(\hat{y} = c|S) = \frac{\exp(-d(S, \bar{z}_c))}{\sum_{c'} \exp(-d(S, \bar{z}_{c'}))},$$

Algorithm 1 Episodic training of ProtoNet with PointMixUp. From line 3 to line 8 is where PointMixUp takes a role in addition to the ProtoNet baseline. Testing stage is similar as training stage, but without line 13 and line 14 which learn new weight from query examples.

Require: Set of sampled episodes $\{\mathcal{D}_i\}$, where $\mathcal{D}_i = \mathcal{D}_i^S \cup \mathcal{D}_i^Q$ denoting the support and query sets
Require: h_θ: feature extractor network: input $\rightarrow$ latent embedding
1: randomly initialize θ
2: **for** episode i **do**
3:     **for** class c **do**
4:         calculate prototype $\bar{z}_c$ from $\mathcal{D}_i^S$, with h_θ.
5:     **end for**
6:     Construct Mixup samples $\mathcal{D}_i^{\text{mix}}$ from support set $\mathcal{D}_i^S$.
7:     Predict the label distributions for mixed examples in $\mathcal{D}_i^{\text{mix}}$, with distance to $\bar{z}_c$.
8:     Update θ with prediction from mixed examples, as episode-specific weights θ_i.
9:     **for** class c **do**
10:         calculate new prototype $\bar{z}_c^{(\theta_i)}$ from $\mathcal{D}_i^S$, with h_{θ_i}
11:     **end for**
12:     Predict the label distributions for query examples in $\mathcal{D}_i^Q$, with distance to $\bar{z}_c^{(\theta_i)}$.
13:     Update θ_i with prediction from query examples.
14:     $\theta \leftarrow \theta_i$
15: **end for**
16: **return** θ

where $d(\cdot,\cdot)$ is the Eudlidean distance in the embedding space. In training stage, the weights θ for the feature extractor h_θ is updated by the cross-entropy loss for the predicted query label distribution and the ground truth.

Few-shot point cloud classification with PointMixup We use PointMixup to learn a better embedding space for each episode. Instead of using the h_θ directly to predict examples from query set, we learn a episode-specific weight θ_i from the mixed data, and the query examples are predicted by h_{θ_i}. We use PointMixup to construct a mixed set $\mathcal{D}_i^{\text{mix}}$ from the labelled support set $\mathcal{D}_i^S$, which consists of examples from $\binom{N_c}{2}$ class pairs and for each class pairs N_s mixed examples are constructed from randomly sampling support examples. Then the weight θ is updated as θ_i from backprop the loss from the prediction of mixed examples from $\mathcal{D}_i^{\text{mix}}$. After that, the label of query examples from $\mathcal{D}_i^Q$ is then predicted with the updated feature extractor h_{θ_i}. See Algorithm 1 for an illustration of the learning scheme.

3 Further Discussion on Interpolation Variants

The proposed PointMixUp adopts *Optimal Assignment (OA) interpolation* for point cloud because of its advantages in theory and in practice. To compare Optimal Assignment interpolation with the two alternative strategies, *Random Assignment (RA) interpolation* and *Point Sampling (PS) interpolation*, the proposed PointMixUp with OA interpolation is the best performing strategy, followed by PS interpolation. RA interpolation, which has a non-shortest path definition of interpolation, does not perform well.

Here we extend the discussion on the two alternative interpolation strategies, through which we analyze the possible advantages and limitations under certain conditions, which in turn validates our choice of applying Optimal Assignment interpolation for PointMixup.

Random Assignment interpolation From our shortest path interpolation hypothesis for Mixup, the inferiority of RA interpolation comes from that it does not obey the shortest path interpolation rule, so that the mixed point clouds from different source examples can easily entangle with each other. From Fig. 3 in the main paper, the Random assignment interpolation produces chaotic mixed examples which can hardly been recognized with the feature from the source class point clouds. Thus, RA interpolation fails especially under heavy Mixup (the value of λ is large).

Point Sampling interpolation: yet another shortest path interpolation Point Sampling interpolation performs relatively well in PointNet++ and sometimes comparable with the Optimal Assignment interpolation. From Fig. 3 in the main paper, the PS interpolation produces mixed examples which can be recognized which classes of source data it comes from.

Reviewing the shortest path interpolation hypothesis, We argue that when the number of points N is large enough, or say $N \to \infty$, Point Sampling interpolation also (approximately) defines a shortest path on the metric space $(\mathcal{S}, d_{\mathrm{EMD}})$ (Note that given the initial and the final points, the shortest path in $(\mathcal{S}, d_{\mathrm{EMD}})$ is not unique). This is a bit counter-intuitive, but reasonable.

We show the *shortest path property*. Recall that point sampling interpolation randomly draws without replacement of points from each set are made according to the sampling frequency λ: $S_{\mathbf{PS}}^{(\lambda)} = S_1^{(1-\lambda)} \cup S_2^{(\lambda)}$, where $S_2^{(\lambda)}$ denotes a randomly sampled subset of S_2, with $\lfloor \lambda N \rfloor$ elements. ($\lfloor \cdot \rfloor$ is the floor function.) And similar for $S_1^{(1-\lambda)}$ with $N - \lfloor \lambda N \rfloor$ elements, such that $S_{\mathbf{PS}}^{(\lambda)}$ contains exactly N points. Imagine that a subset $S_1^{(1-\lambda)}$ with a number of $N - \lfloor \lambda N \rfloor$ points in $S_{\mathbf{PS}}^{(\lambda)}$ are identical with that in S_1. For $d_{\mathrm{EMD}}(S_{\mathbf{PS}}^{(\lambda)}, S_1)$, the optimal assignment will return these identical points as matched pairs, thus they contribute zero to the overall EMD distance. Thus,

$$
\begin{aligned}
d_{\mathrm{EMD}}(S_{\mathbf{PS}}^{(\lambda)}, S_1) &= \frac{N - \lfloor \lambda N \rfloor}{N} d_{\mathrm{EMD}}(S_{\mathbf{PS}}^{(\lambda)} \setminus S_1^{(1-\lambda)}, S_1 \setminus S_1^{(1-\lambda)}) \\
&= \frac{N - \lfloor \lambda N \rfloor}{N} d_{\mathrm{EMD}}(S_2^{(\lambda)}, S_1 \setminus S_1^{(1-\lambda)}) \\
&\approx \frac{N - \lfloor \lambda N \rfloor}{N} d_{\mathrm{EMD}}(S_2, S_1) \\
&\approx (1 - \lambda) \cdot d_{\mathrm{EMD}}(S_1, S_2),
\end{aligned}
$$

from which $d_{\mathrm{EMD}}(S_2^{(\lambda)}, S_1 \setminus S_1^{(1-\lambda)}) \approx d_{\mathrm{EMD}}(S_2, S_1)$ is because that S_1 and $S_1 \setminus S_1^{(1-\lambda)}$ are the point clouds representing the same shape but with different density, and the same with S_2 and $S_2^{(\lambda)}$.

Similarly, $d_{\mathrm{EMD}}(S_{\mathbf{PS}}^{(\lambda)}, S_2) \approx \lambda \cdot d_{\mathrm{EMD}}(S_1, S_2)$, and thus $d_{\mathrm{EMD}}(S_{\mathbf{PS}}^{(\lambda)}, S_1) + d_{\mathrm{EMD}}(S_{\mathbf{PS}}^{(\lambda)}, S_2) = d_{\mathrm{EMD}}(S_1, S_2)$, which in turn proves the shortest path property.

We note that the *linearity* of PS interpolation w.r.t. d_{EMD} also holds and the proof can be derived similarly. Thus, although strictly not an ideally continuous interpolation path, PS interpolation is (appoximately) a shortest path linear interpolation in $(\mathcal{S}, d_{\mathrm{EMD}})$, which explains its good performance.

Point Sampling interpolation: limitations The limitation of PS interpolation is from that the mix ratio λ controls change of local density distribution, but the underlying shape does not vary with λ. So, as shown in Table **??**, PS interpolation fails with PointNet [1], which is ideally invariant to the point density, because a max pooling operation aggregates the information from all the points.

A question which may come with PS interpolation is that how it performs relatively well with PointNet++, which is also designed to be density-invariant. This is due to the sampling and grouping stage. PointNet++ takes same operation as PointNet in learning features, but in order to be hierarchical, the

Table 1: **Different interpolation strategies on PointNet [1]** Following the original paper [1] we test on unaligned setting. PS interpolation fails with PointNet as a density-invariant model. The numbers are accuracy in percentage.

Baseline	PointMixup (OA)	RA	PS
89.2	**89.9**	88.2	88.7

sampling and grouping stage, especially the farthest point sampling (fps) operation is not invariant to local density changes such that it samples different groups of farthest points, resulting in different latent point cloud feature representations. Thus, PointNet++ is invariant to global density but not invariant to local density differences, which makes PS interpolation as a working strategy for PointNet++. However, we may still expect that the performance of Mixup based on PS interpolation is limited, because it does not work well with PointNet as a basic component in PointNet++.

By contrast, the proposed PointMixup with OA interpolation strategy is not limited by the point density invariance. As a well established interpolation, OA interpolation smoothly morphes the underlying shape. So we claim that OA interpolation is a more generalizable strategy.

References

1. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: CVPR (2017)
2. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: NeurIPS (2017)
3. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: NeurIPS (2017)