FM-OCC: FEATURE-DIVERSE UPSAMPLING AND MEMORY-AWARE DECODING FOR 3D OCCUPANCY PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

3D semantic occupancy prediction estimates voxel-wise occupancy and semantic labels throughout a scene. Our experiments reveal that conventional sparse transposed-convolution upsampling yields poorly differentiated voxel responses and that successive decoder layers discard previously correct semantic predictions. To address these issues, we present Feature-Diverse Upsampling and Memory-Aware Decoding for 3D Occupancy Prediction (**FM-Occ**), comprising two components. First, 3D Sparse Subvoxel Shuffle (**3S**) mitigates feature homogenization by constructing candidate child features in the channel dimension, jointly transforming and normalizing them, and then rearranging them onto a finer sparse grid with explicit feature–coordinate correspondence. Second, Entropy-Regulated Layer-wise Memory (**ER-LM**) retains semantic information across decoder layers by maintaining a layer-wise memory for each query through gated recurrent updates and using prediction entropy to regulate the historical information propagated to the next layer. During training, a class-weighted entropy loss $\mathcal{L}_{\text{cwe}}$ provides label-guided regularization for matched semantic queries. On SemanticKITTI, 3S increases the spatial coefficient of variation of normalized voxel-wise root-mean-square (RMS) responses from 0.41% to 4.03%, while ER-LM reduces semantic degradation and improves error correction across decoder layers. The combined FM-Occ improves mIoU from 12.16% to 12.73% with negligible memory overhead. ER-LM further improves mIoU in the BEV-based DISC framework from 16.78% to 17.17%, supporting its applicability across 3D voxel-based and BEV-based architectures.

1 INTRODUCTION

3D occupancy prediction (Cao & De Charette, 2022; Wei et al., 2023; Zhang et al., 2023) aims to infer voxel-wise occupancy states and semantic categories in 3D space, providing a dense and structured scene representation for autonomous driving and robotic perception. Compared with 3D object detection (Yan et al., 2018; Lang et al., 2019; Li et al., 2024), which outputs sparse bounding boxes, occupancy prediction explicitly models scene geometry and spatial continuity. Following sparse occupancy approaches such as SparseOcc and OPUS (Tang et al., 2024; Wang et al., 2024a), we study an image-based encoder–decoder architecture that constructs sparse 3D features and progressively predicts semantic occupancy through feature upsampling and iterative Transformer decoding.

We further examine these two key stages and identify two overlooked issues: weak spatial differentiation of upsampled voxel features and progressive loss of semantic information during iterative decoding.

Observation 1: Limited differentiation of upsampled voxel features. Our analysis shows that sparse transposed convolution followed by group normalization produces voxel features with limited spatial variation in their response magnitudes. This observation suggests that the organization of feature construction and normalization deserves attention alongside the expansion of the voxel grid. We introduce 3D Sparse Subvoxel Shuffle (3S), which constructs candidate child representations in the channel dimension, jointly transforms and normalizes them, and then rearranges them onto the finer sparse grid. Across 50 validation frames, 3S increases the spatial CV of voxel-wise root-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

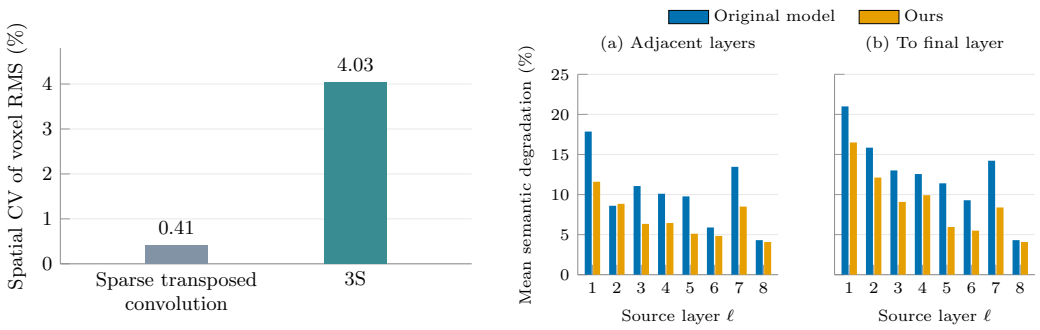

Figure 1: **Two challenges in occupancy prediction: feature homogenization and fragile semantic consistency. Left:** Spatial variation of voxel-wise feature responses after upsampling, measured by spatial CV (higher indicates greater variation). **Right:** Mean class-wise semantic degradation (mD; lower is better) without and with ER-LM + $\mathcal{L}_{\text{cwe}}$, comparing (a) adjacent decoder layers and (b) intermediate layers with the final layer.

mean-square (RMS) responses from 0.41% to 4.03% (Fig. 1, left). 3S also improves both mIoU and IoU.

Observation 2: Semantic degradation across decoder layers. Our analysis reveals substantial semantic degradation during iterative decoding, with a mean class-wise degradation rate of 21.00% from the first to the final decoder layer (Fig. 1, right). To address this issue, we introduce Entropy-Regulated Layer-wise Memory (ER-LM), which maintains a gated memory for each query across decoder layers. Shared recurrent parameters accumulate historical query information, while query-specific classification entropy regulates the memory state propagated to subsequent stages. During training, a complementary class-weighted entropy loss $\mathcal{L}_{\text{cwe}}$ encourages concentrated class predictions on matched positive queries, with greater relative emphasis on rare categories. In the separate decoding experiments, ER-LM trained with $\mathcal{L}_{\text{cwe}}$ improves final mIoU from 12.16% to 12.88% and reduces the mean class-wise semantic degradation rate from 21.00% to 16.50%.

Our contributions are summarized as follows:

- We propose FM-Occ, a 3D semantic occupancy framework that improves feature differentiation during sparse upsampling and semantic preservation across decoder layers.
- We propose 3S to preserve spatial variation in feature magnitudes during sparse upsampling by reorganizing child-feature construction, normalization, and spatial expansion.
- We introduce ER-LM to mitigate cross-layer semantic degradation through entropy-regulated memory, complemented by a class-weighted entropy loss during training. ER-LM generalizes across both 3D voxel-based and BEV-based frameworks.

2 RELATED WORK

2.1 Representative Approaches for 3D Occupancy Prediction

Vision-based semantic occupancy methods differ in how they represent and process 3D scenes. Dense voxel-based approaches construct features on regular 3D grids to predict scene geometry and semantics (Cao & De Charette, 2022; Wang et al., 2023; Wei et al., 2023; Zhang et al., 2023). BEV-based and tri-perspective representations reduce volumetric processing by projecting or decomposing 3D scene features onto compact planes (Hou et al., 2024; Xue et al., 2024; Liu et al., 2025; Huang et al., 2023; Zuo et al., 2023). Sparse approaches avoid maintaining features over the entire 3D grid by operating on a limited set of active elements. Tang et al. (2024) adopts a sparse voxel representation and enhances it through multi-scale feature interaction. Wang et al. (2024a) adopts a different sparse formulation, representing occupancy as a set of occupied locations and semantic classes predicted by learnable queries. These approaches illustrate different choices of scene

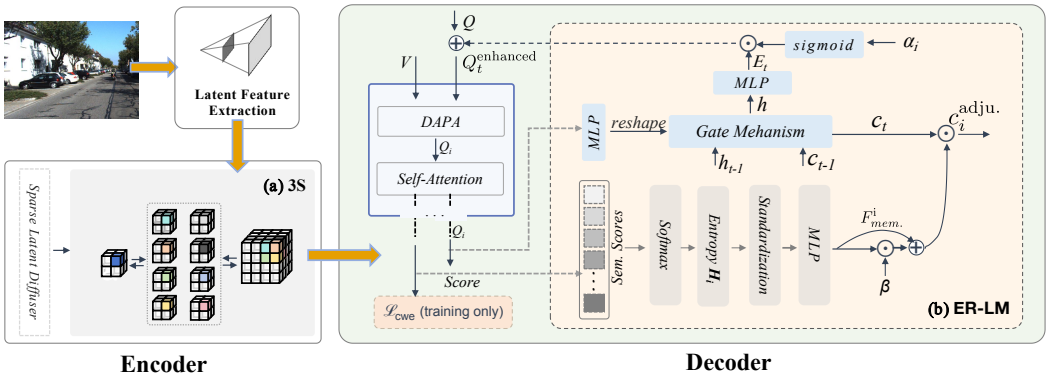

Figure 2: **Overall framework of FM-Occ.** 3S performs sparse 3D feature upsampling, while ER-LM propagates information across successive decoder stages under entropy-guided regulation. $\mathcal{L}_{\text{cwe}}$ is applied only during training.

representation; our work examines feature reconstruction and semantic information retention within sparse voxel and iterative decoding architectures.

2.2 Feature Upsampling

2D feature upsampling has been explored through learned kernels, content-aware reassembly, adaptive sampling, and channel-to-space rearrangement (Zeiler et al., 2010; Wang et al., 2019; Liu et al., 2023; Shi et al., 2016). These methods operate on dense feature maps with spatial positions implicitly defined by regular tensor grids. Unlike dense 2D feature maps, sparse 3D representations store only active voxel features together with explicit coordinates. Sparse 3D upsampling must therefore expand both features and coordinates while maintaining their correspondence. 3S addresses this requirement through channel-to-space feature rearrangement and matched coordinate expansion.

2.3 Information Retention in Occupancy Decoding

Query-based occupancy methods employ stacked decoder layers to progressively refine semantic predictions (Zhang et al., 2023; Wang et al., 2024b; Oh et al., 2025). In these decoders, successive layers repeatedly update the query representations, with information from earlier stages carried forward implicitly through the evolving queries. As a result, previously acquired semantic information may be altered or lost during subsequent refinement, motivating explicit mechanisms for cross-layer information retention.

3 METHOD

The overall framework of FM-Occ is illustrated in Fig. 2. Latent Feature Extraction first maps image features into a sparse 3D representation. The Sparse Latent Diffuser then performs scene completion, followed by a multi-scale sparse feature pyramid. 3S is used to recover higher-resolution sparse features during multi-scale feature fusion. Finally, the Transformer decoder predicts semantic occupancy, where ER-LM propagates information across decoder stages under entropy-guided regulation. The class-weighted entropy loss $\mathcal{L}_{\text{cwe}}$ is additionally applied during training.

3.1 3D Sparse Subvoxel Shuffle

Sparse feature pyramids rely on upsampling to recover fine-grained voxel representations from coarse features. Conventional sparse transposed convolution generates child features while expanding the spatial grid, coupling feature construction with spatial placement. We introduce 3S, which first generates r^3 candidate child representations in the channel dimension for each active parent

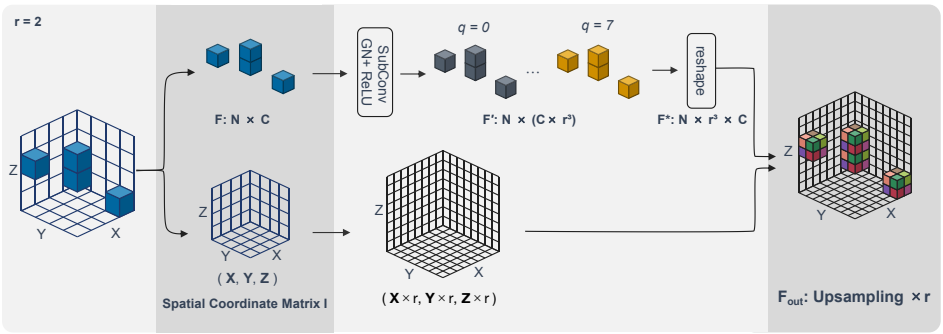

Figure 3: **3S.** The **upper branch** expands feature channels through a submanifold convolution, applies group normalization and ReLU, and reshapes the features into r^3 child representations per parent voxel. The **lower branch** generates the corresponding fine-grid coordinates. The two branches combine to form the upsampled sparse tensor, illustrated here for $r = 2$.

voxel. These representations are jointly normalized and activated at the coarse resolution before being deterministically rearranged to their corresponding locations on the finer sparse grid. The overall process is illustrated in Fig. 3.

3.1.1 Sparse tensor representation

Let the input sparse tensor be represented as $\mathcal{S} = (\mathbf{F}, \mathbf{I}, \mathbf{s})$, where $\mathbf{F} \in \mathbb{R}^{N \times C_{\text{in}}}$ contains the features of N active voxels, $\mathbf{I} \in \mathbb{Z}^{N \times 4}$ stores their batch and three-dimensional spatial indices, and $\mathbf{s} = (S_1, S_2, S_3)$ denotes the spatial shape. The coordinate of the n-th active voxel is written as $\mathbf{I}_n = (b_n, \mathbf{p}_n)$, where b_n is the batch index and $\mathbf{p}_n = (p_{n,1}, p_{n,2}, p_{n,3})$ is its three-dimensional spatial coordinate.

3.1.2 Child-feature construction and joint normalization

Given an upsampling factor r, a submanifold sparse convolution expands each input feature from C_{in} channels to $r^3 C_{\text{out}}$ channels, yielding $\widetilde{\mathbf{F}} = \text{Conv}_{1\times1\times1}(\mathbf{F})$, where $\widetilde{\mathbf{F}} \in \mathbb{R}^{N \times (r^3 C_{\text{out}})}$.

Before spatial rearrangement, we apply group normalization and a ReLU activation to the expanded representation, obtaining $\overline{\mathbf{F}} = \text{ReLU}(\text{GN}(\widetilde{\mathbf{F}}))$.

3.1.3 Channel-to-space rearrangement

The normalized representation is reshaped as

$$\overline{\mathbf{F}} \in \mathbb{R}^{N \times (r^3 C_{\text{out}})} \longrightarrow \mathbf{F}^{\text{sub}} \in \mathbb{R}^{N \times r^3 \times C_{\text{out}}}. \tag{1}$$

Let $q \in \{0, \ldots, r^3 - 1\}$ denote a child index. The exact feature mapping implemented by 3S is

$$\mathbf{F}^{\text{sub}}_{n,q,c} = \overline{\mathbf{F}}_{n,\, cr^3+q}, \tag{2}$$

where $c \in \{0, \ldots, C_{\text{out}} - 1\}$. Flattening the parent and child dimensions produces

$$\mathbf{F}_{\text{out}} \in \mathbb{R}^{(Nr^3) \times C_{\text{out}}}, \qquad \mathbf{F}_{\text{out}}[nr^3 + q, c] = \mathbf{F}^{\text{sub}}_{n,q,c}. \tag{3}$$

Thus, each parent voxel produces r^3 location-specific child-feature vectors.

3.1.4 Coordinate expansion and alignment

We enumerate the complete set of three-dimensional child offsets as $\mathcal{D}_r = \{(d_1, d_2, d_3) \mid d_1, d_2, d_3 \in \{0, \ldots, r-1\}\}$. Let $\mathbf{d}_q \in \mathcal{D}_r$ be the offset associated with child index q. The output coordinate paired with the feature in Eq. equation 3 is

$$\mathbf{I}_{\text{out}}[nr^3 + q] = (b_n,\, r\mathbf{p}_n + \mathbf{d}_q)\,. \tag{4}$$

The output spatial shape is correspondingly updated to $\mathbf{s}_{\text{out}} = (rS_1, rS_2, rS_3)$.

The resulting sparse tensor is $\mathcal{S}_{\text{out}} = (\mathbf{F}_{\text{out}}, \mathbf{I}_{\text{out}}, \mathbf{s}_{\text{out}})$. Within each sparse pyramid stage, $\mathcal{S}_{\text{out}}$ is subsequently merged with the existing feature level at the same resolution.

3.2 Entropy-Regulated Layer-wise Memory

Iterative query updates can both correct semantic errors and disrupt previously correct predictions. To support the reuse of semantic evidence across decoding stages, we introduce ER-LM. As illustrated in the right panel of Fig. 1, ER-LM maintains a gated memory alongside the evolving query representations and incorporates its output into subsequent decoder updates. The recurrent module is shared across decoding stages, while classification entropy provides a query-specific signal for modulating memory propagation.

3.2.1 Gated Memory across Decoding Stages

Let $\mathbf{Q}^{(\ell)} \in \mathbb{R}^{B\times N_q\times D}$ denote the query representations after decoder layer ℓ, where B, N_q, and D are the batch size, number of queries, and query feature dimension, respectively. The initial learnable queries are denoted by $\mathbf{Q}^{(0)}$. We use an LSTM with hidden dimension D_h to maintain a separate memory state for each query across the L decoding stages.

At the beginning of each forward pass, the hidden and cell states are initialized to zero, $\mathbf{h}^{(0)} = \mathbf{0}$ and $\widetilde{\mathbf{c}}^{(0)} = \mathbf{0}$, where both states have shape $1 \times (BN_q) \times D_h$.

Before decoder layer ℓ, the incoming queries are reshaped as

$$\mathbf{X}^{(\ell)} = \mathcal{R}\Big(\mathbf{Q}^{(\ell-1)}\Big) \in \mathbb{R}^{1\times(BN_q)\times D}$$

and passed to the shared LSTM:

$$\Big(\mathbf{h}^{(\ell)}, \mathbf{c}^{(\ell)}\Big) = \mathrm{LSTM}\left(\mathbf{X}^{(\ell)}; \mathbf{h}^{(\ell-1)}, \widetilde{\mathbf{c}}^{(\ell-1)}\right), \tag{5}$$

where $\mathcal{R}$ merges the batch and query dimensions, and $\mathcal{R}^{-1}$ restores their original layout. Each LSTM invocation processes one recurrent step, with the hidden and cell states propagated across decoding stages. The standard LSTM gates control the incorporation of current query information and the retention of previous memory. Projecting the hidden state to the query dimension and restoring the batch–query layout yields the memory feature $\mathbf{E}^{(\ell)} \in \mathbb{R}^{B\times N_q\times D}$, which is fused with the incoming queries before decoder layer ℓ:

$$\mathbf{E}^{(\ell)} = \mathcal{R}^{-1}\Big(\mathbf{h}^{(\ell)}\mathbf{W}_p^{\top} + \mathbf{b}_p\Big). \tag{6}$$

$$\widehat{\mathbf{Q}}^{(\ell-1)} = \mathbf{Q}^{(\ell-1)} + \alpha_\ell \mathbf{E}^{(\ell)}. \tag{7}$$

$$\mathbf{Q}^{(\ell)} = \mathcal{D}_\ell\Big(\widehat{\mathbf{Q}}^{(\ell-1)}, \mathbf{F}_{s(\ell)}\Big). \tag{8}$$

where $\mathbf{W}_p \in \mathbb{R}^{D\times D_h}$ and $\mathbf{b}_p \in \mathbb{R}^D$ are the shared projection matrix and bias, respectively, $\mathcal{D}_\ell$ denotes decoder layer ℓ, and $\mathbf{F}_{s(\ell)}$ is the feature level accessed at that stage. The fusion weight is $\alpha_\ell = \sigma(a_\ell)$; the scalar a_ℓ is learned separately for each decoder layer and initialized to 0.1.

3.2.2 Entropy-Regulated Memory Propagation

We use classification entropy to modulate the cell state transmitted between decoding stages. Let $\mathbf{S}^{(\ell-1)}$ denote the classification logits obtained from the preceding stage, and let K be the number of classification outputs, including the no-object class. For each query q, the class probabilities are $p_{q,k}^{(\ell-1)} = \mathrm{Softmax}(\mathbf{S}_q^{(\ell-1)})_k$. The entropy and its normalized form are

$$H_q^{(\ell-1)} = -\sum_{k=1}^{K} p_{q,k}^{(\ell-1)} \log\Big(p_{q,k}^{(\ell-1)} + \epsilon\Big). \tag{9}$$

$$\overline{H}_q^{(\ell-1)} = \frac{H_q^{(\ell-1)}}{\log K}. \tag{10}$$

where $\epsilon = 10^{-8}$ ensures numerical stability.

For stages $\ell \geq 2$, a shared multilayer perceptron maps the normalized entropy to a query-specific modulation coefficient:

$$r_q^{(\ell)} = \sigma\left(g\left(\overline{H}_q^{(\ell-1)}\right)\right). \tag{11}$$

where $g(\cdot)$ is a lightweight MLP that maps prediction entropy to a gating coefficient. At the first stage, we use the default coefficient $r_q^{(1)} = 0.5$.

After the LSTM update in Eq. equation 5, the cell state is modulated as

$$\widetilde{\mathbf{c}}^{(\ell)} = \mathbf{c}^{(\ell)} \odot \left(1 + \beta\, \mathcal{R}\Big(\mathbf{r}^{(\ell)}\Big)\right), \qquad \beta = 0.3. \tag{12}$$

The coefficient $\beta = 0.3$ bounds the modulation gain between 1 and 1.3. The adjusted cell state $\widetilde{\mathbf{c}}^{(\ell)}$ is passed to the next recurrent step, while the current hidden output $\mathbf{h}^{(\ell)}$ is used for query fusion.

3.2.3 Class-Weighted Entropy Regularization

Because the prediction entropy used by ER-LM is not directly supervised, we introduce $\mathcal{L}_{\text{cwe}}$ to explicitly regularize it on matched positive queries. Ground-truth assignments determine the positive queries and their class weights. This entropy supervision also reduces uncertainty in the predicted class distributions, complementing the classification loss.

For prediction stage ℓ, let $\mathcal{Q}_+^{(\ell)}$ denote the queries matched to ground-truth semantic masks, excluding those assigned to the no-object class; let $y_q^{(\ell)}$ denote their assigned labels, and $w_{y_q^{(\ell)}}$ the corresponding class weights used by the classification loss. The stage-wise loss is defined as

$$\mathcal{L}_{\text{cwe}}^{(\ell)} = \Big(\sum\nolimits_{q\in\mathcal{Q}_+^{(\ell)}} w_{y_q^{(\ell)}} H_q^{(\ell)}\Big) \Big/ \Big(\sum\nolimits_{q\in\mathcal{Q}_+^{(\ell)}} w_{y_q^{(\ell)}}\Big). \tag{13}$$

We apply this regularizer to the initial query predictions and the outputs of all decoder layers; the overall training objective is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{base}} + \lambda_{\text{cwe}}\mathcal{L}_{\text{cwe}}, \qquad \mathcal{L}_{\text{cwe}} = \sum_{\ell=0}^{L} \mathcal{L}_{\text{cwe}}^{(\ell)}. \tag{14}$$

where $\ell = 0$ denotes the initial prediction stage, L is the number of decoder layers, and $\mathcal{L}_{\text{base}}$ comprises the original classification, mask, Dice, and voxel-level losses. We set $\lambda_{\text{cwe}} = 1.0$ in our experiments.

4 Experimental Results

4.1 Experimental Setup

We evaluate FM-Occ on the SemanticKITTI (Behley et al., 2019) dataset. SemanticKITTI is a large-scale dataset for 3D semantic occupancy prediction in autonomous driving scenarios, based on the raw KITTI sequences. The dataset contains 21 classes, including 19 semantic categories, one free-space class, and one unknown class. We train FM-Occ for 30 epochs on four NVIDIA RTX 4090 GPUs using AdamW with an initial learning rate of 1×10^{-4} and a total batch size of 4. We adopt a step learning-rate schedule.

We adopt commonly used evaluation metrics for 3D occupancy prediction, including voxel-wise Intersection over Union (IoU) and mean Intersection over Union (mIoU). To provide a more systematic analysis of model performance across different semantic structures, we further propose category-grouped evaluation metrics, namely ComplexIoU, TailIoU, and RegularIoU. Detailed definitions and the category grouping of these metrics are provided in Appendix A.1.

4.2 Comparison with state-of-the-art

Table 1 compares FM-Occ with recent semantic scene completion methods on SemanticKITTI. FM-Occ achieves an IoU of 36.18% and an mIoU of 12.73%. To further evaluate the transferability of ER-LM, we apply it from the sparse 3D voxel-based FM-Occ to the BEV-based DISC framework, improving DISC from 16.78% to 17.17% mIoU and from 45.54% to 45.67% IoU. The resulting 17.17% mIoU is comparable to MRA (17.14%) and higher than SGFormer (16.68%), demonstrating competitive performance against recent methods and the transferability of ER-LM across different scene representations.

Table 1: Quantitative comparison on the SemanticKITTI validation set

Method	IoU	mIoU	road (15.30%)	sidewalk (11.13%)	parking (1.12%)	other-grnd. (0.56%)	building (14.10%)	car (3.92%)	truck (0.16%)	bicycle (0.03%)	motorcycle (0.03%)	other-veh. (0.20%)	vegetation (39.3%)	trunk (0.51%)	terrain (9.17%)	person (0.07%)	bicyclist (0.07%)	motorcyclist (0.05%)	fence (1.90%)	pole (0.29%)	traf.-sign (0.08%)
TPVFormer†(Huang et al., 2023)	35.61	11.36	56.50	25.87	20.60	0.85	13.88	23.81	8.08	0.36	0.05	4.35	16.92	2.26	30.38	0.51	0.89	0.00	5.94	3.14	1.52
MonoScene†(Cao & De Charette, 2022)	36.86	11.08	56.52	26.72	14.27	0.46	14.09	23.26	6.98	0.61	0.45	1.48	17.89	2.81	29.64	1.86	1.20	0.00	5.84	4.14	2.25
OccFormer(Zhang et al., 2023)	36.50	13.46	58.85	26.88	19.61	0.31	14.40	25.09	**25.53**	0.81	1.19	8.52	19.63	3.93	32.62	2.78	2.82	0.00	5.61	4.26	2.86
VoxFormer*(Li et al., 2023)	43.92	13.12	54.44	27.05	16.97	0.28	19.64	26.78	7.66	1.50	0.41	5.68	26.12	6.33	32.80	1.15	0.26	0.00	7.93	8.94	5.41
Symphonies(Jiang et al., 2024)	41.92	14.89	56.37	27.58	15.28	0.95	21.64	28.68	20.44	2.54	2.82	**13.89**	25.72	6.60	30.87	3.52	2.24	0.00	8.40	9.57	5.76
SGFormer(Guo et al., 2025)	45.01	16.68	60.85	31.53	19.32	**2.14**	**26.05**	32.17	20.30	2.95	3.10	11.55	27.11	8.28	38.47	3.66	1.49	0.00	9.28	11.58	7.22
MRA(Yang & Peng, 2026)	**47.28**	17.14	61.92	33.97	**27.35**	0.06	25.89	**34.63**	20.12	0.53	1.69	11.06	**30.08**	**9.54**	38.51	0.54	0.16	0.00	**10.13**	**13.15**	6.27
SparseOcc*(Tang et al., 2024)	35.21	12.15	57.43	27.59	21.18	0.00	14.97	25.32	1.62	0.35	0.92	11.51	18.25	3.29	31.17	3.82	0.40	0.00	6.41	4.06	2.59
DISC*(Liu et al., 2025)	45.54	16.78	**63.65**	**34.35**	24.46	1.56	23.42	32.95	15.55	**3.04**	3.06	8.06	27.41	7.66	**39.01**	3.82	**3.11**	0.00	9.73	11.08	6.88
FM-Occ	36.18	12.73	59.00	29.03	20.55	0.62	15.88	25.56	7.54	0.48	0.73	10.72	18.84	3.48	31.83	2.66	1.69	0.00	6.12	4.31	2.77
DISC + ER-LM	45.67 (+0.13)	**17.17** (+0.39)	62.83 (-0.82)	33.96 (-0.39)	24.66 (+0.20)	2.04 (+0.48)	23.66 (+0.24)	33.00 (+0.05)	19.98 (+4.43)	2.93 (-0.11)	**3.19** (+0.13)	11.65 (+3.59)	27.61 (+0.20)	7.90 (+0.24)	37.71 (-1.30)	**4.02** (+0.20)	2.95 (-0.16)	**0.01** (+0.01)	9.56 (-0.17)	10.97 (-0.11)	**7.56** (+0.68)

Note: * denotes results reproduced by our implementation, and † denotes results taken from Li et al. (2023). For DISC + ER-LM, values in parentheses indicate differences from the reproduced DISC baseline. The best results are highlighted in **bold**, and the second-best results are underlined.

4.3 Ablation Studies

Table 2 compares the proposed components and their combinations. 3S improves IoU and mIoU by 0.49% and 0.26%, respectively. ER-LM primarily improves semantic prediction, increasing mIoU by 0.51% while maintaining comparable occupancy IoU. Adding $\mathcal{L}_{\text{cwe}}$ further increases mIoU to 12.88%. The full configuration improves all five accuracy metrics over the baseline, although the gains of the individual components are not strictly additive. These improvements introduce negligible memory overhead, with measured GPU memory increasing from 13.05 to 13.09 GB.

Table 2: Ablation studies of 3S, ER-LM, and $\mathcal{L}_{\text{cwe}}$

3S	ER-LM	$\mathcal{L}_{\text{cwe}}$	IoU	mIoU	Complex	Tail	Regular	Mem.
			36.13	12.16	23.00	1.36	10.32	13.05
✓			**36.62**	12.42	23.14	1.53	10.80	13.06
	✓		36.11	12.67	23.07	**1.81**	11.39	13.06
	✓	✓	35.85	**12.88**	23.31	1.55	**12.04**	13.06
✓	✓	✓	36.18	12.73	**23.34**	1.39	11.69	13.09

Table 3 further evaluates robustness across five random seeds. Both 3S and ER-LM retain their principal improvements across repeated runs. 3S shows stable semantic performance, while ER-LM provides a consistent mIoU gain and reduces the variation on regular categories. These results indicate that the observed improvements are not dependent on a single favorable training run.

Table 3: Robustness evaluation across five random seeds. Results are reported as mean ± standard deviation

Method	IoU	mIoU	Complex	Tail	Regular
--	35.91±0.16	12.34±0.21	22.96±0.34	1.29±0.16	11.31±0.95
+ 3S	**36.18**±0.32	12.44±0.12	**23.15**±0.36	**1.49**±0.27	10.89±0.46
+ ER-LM	35.91±0.26	**12.56**±0.25	22.78±0.51	1.32±0.15	**11.87**±0.39

Note. All configurations were re-run under the same setting, which differs slightly from Table 2.

4.4 Analysis of 3S

4.4.1 Comparison of Upsampling Feature Responses

We further examine the spatial variation of voxel-wise feature responses produced by different upsampling schemes. As shown in Table 4, 3S achieves a mean spatial coefficient of variation (CV) of 4.03%, compared with 0.41% for sparse transposed convolution, corresponding to a $9.83\times$ increase. Figure 4 further shows clearer response separation between the car and its surrounding background in region A and between vegetation and the ground in regions B and C. The definition and measurement protocol of the feature-response variation metric, together with details of the feature-response visualization and analysis, are provided in Appendix A.2.

Table 4: Comparison of spatial variation in voxel-wise RMS responses

Method	CV (%)	Ratio
Transposed conv.	0.41	1.00×
3S	**4.03**	9.83×

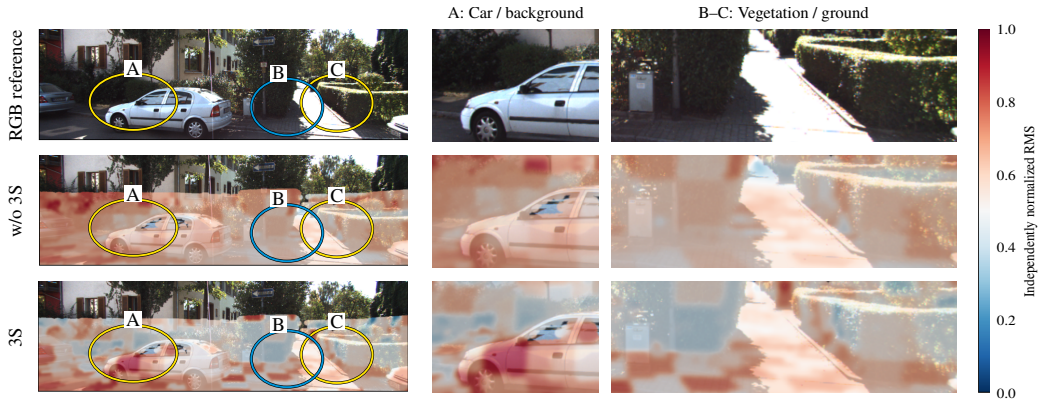

Figure 4: **Feature-response comparison after upsampling.** Full images are shown on the left, with enlarged regions on the right. Each response map is independently min–max normalized for visualization.

4.4.2 Effect of Normalization and Activation Placement

We examine two alternative explanations for the upsampling results: whether the transposed-convolution reference is disadvantaged by the absence of ReLU after normalization, and whether moving normalization after spatial rearrangement improves 3S. As shown in Table 5, adding ReLU to transposed convolution does not improve mIoU, suggesting that ReLU alone does not account for the gains of 3S. Applying GN after rearrangement gives similar mIoU to the original 3S, suggesting that normalization order has a limited effect on this metric in our experiments.

4.4.3 Subvoxel Feature Allocation Variants

As shown in Table 6, permute–reshape interprets the expanded channels as $r^3 \times C_{\text{out}}$, whereas 3S uses $C_{\text{out}} \times r^3$. Its mIoU improvement from 12.16% to 12.32% supports channel expansion and rearrangement, while the 12.42% achieved by 3S validates the proposed feature allocation. Softmax improves mIoU and RegularIoU, but its sum-to-one competition reduces occupancy IoU. Sigmoid improves ComplexIoU and RegularIoU but lowers TailIoU, possibly because it favors dominant classes under imbalanced supervision. We therefore retain deterministic rearrangement for its balanced performance. Both variants are defined in Appendix A.3.

Table 5: Effect of activation and normalization placement

Method	IoU	mIoU	Complex	Tail	Regular
Transposed conv. + GN	36.13	12.16	23.00	1.36	10.32
+ ReLU	36.17	12.10	22.95	1.34	10.18
Shuffle → GN → ReLU	36.22	12.37	23.10	**2.02**	10.20
3S (GN → ReLU → shuffle)	**36.62**	**12.42**	**23.14**	1.53	**10.80**

Table 6: Subvoxel feature allocation

Method	IoU	mIoU	Complex	Tail	Regular
--	36.13	12.16	23.00	1.36	10.32
Permute–reshape	36.17	12.32	22.91	1.49	10.81
3S	**36.62**	12.42	23.14	**1.53**	10.80
+ Softmax allocation	35.92	**12.55**	23.18	1.36	11.33
+ Sigmoid allocation	36.35	12.51	**23.36**	1.02	**11.34**

4.5 Analysis of ER-LM

As shown in Table 7, the results of EMA demonstrate the value of historical information, but simple exponential smoothing yields only limited semantic gains. ER-LM achieves a larger mIoU improvement through learnable memory and forgetting, with more balanced improvements across category groups. We further evaluate applying LayerNorm to the LSTM output before projection, with the corresponding ablation results provided in Appendix A.4.

Table 7: Analysis of historical information aggregation

Method	IoU	mIoU	Complex	Tail	Regular
--	35.78	12.16	22.84	1.15	11.33
EMA	**36.19**	12.37	22.95	**1.63**	10.77
ER-LM	36.09	**12.53**	**22.96**	1.32	**11.57**

Note. All configurations were re-run under the same setting, which differs slightly from Table 2.

We use mean class-wise degradation (mD) and correction (mR) rates to evaluate semantic transitions across decoder layers. As shown in Table 8, ER-LM + $\mathcal{L}_{\text{cwe}}$ reduces degradation and improves correction in most adjacent-layer transitions and all intermediate-to-final comparisons. From the first to the final layer, mD decreases from 21.00% to 16.50%, while mR increases from 1.16% to 1.43%. Detailed definitions and evaluation protocols are provided in Appendix A.5.

Table 8: Class-balanced semantic transition rates (%). Lower mD and higher mR are better

	mD ↓		mR ↑			mD ↓		mR ↑	
Transition	--	Ours	--	Ours	Transition	--	Ours	--	Ours
$1 \to 2$	17.86	**11.60**	**0.99**	0.88	$1 \to 9$	21.00	**16.50**	1.16	**1.43**
$2 \to 3$	**8.60**	8.84	0.64	**0.99**	$2 \to 9$	15.85	**12.12**	0.84	**1.50**
$3 \to 4$	11.06	**6.34**	0.57	**0.72**	$3 \to 9$	13.01	**9.08**	0.73	**1.33**
$4 \to 5$	10.10	**6.45**	0.55	**0.99**	$4 \to 9$	12.56	**9.92**	0.77	**1.39**
$5 \to 6$	9.77	**5.11**	0.51	**0.73**	$5 \to 9$	11.40	**5.95**	0.71	**1.03**
$6 \to 7$	5.89	**4.84**	0.60	**0.87**	$6 \to 9$	9.29	**5.50**	0.65	**0.95**
$7 \to 8$	13.46	**8.50**	0.45	**0.92**	$7 \to 9$	14.22	**8.39**	0.56	**0.97**
$8 \to 9$	4.31	**4.08**	0.38	**0.55**	$8 \to 9$	4.31	**4.08**	0.38	**0.55**

Note. -- denotes the model without ER-LM or $\mathcal{L}_{\text{cwe}}$; Ours denotes ER-LM + $\mathcal{L}_{\text{cwe}}$. Bold indicates the better value within each layer pair.

5 CONCLUSION

We presented FM-Occ for sparse 3D semantic occupancy prediction, addressing feature homogenization during upsampling and semantic degradation across decoder layers. The proposed 3S constructs and jointly normalizes child features before spatial rearrangement, preserving greater variation in voxel feature responses. ER-LM retains historical query information through gated recurrent memory and entropy-guided state modulation, complemented by the class-weighted entropy loss $\mathcal{L}_{\text{cwe}}$. Experiments on SemanticKITTI demonstrate improved occupancy and semantic prediction, while the cross-layer analysis shows better preservation of correct semantic predictions and correction of existing errors on average across classes.

AI USE STATEMENT

In this work, we used generative AI tools to assist with language editing, suggestions for additional experimental analyses, and code generation. We independently drafted the manuscript and identified the relevant literature. Generative AI tools were subsequently used to improve the clarity and fluency of parts of the writing, including the Introduction. We manually reviewed and edited all AI-assisted revisions to ensure technical accuracy and consistency with the cited literature.

After completing the main experiments, we consulted generative AI tools for suggestions on additional experiments that could strengthen the rigor of the empirical evaluation. We independently evaluated these suggestions, selected the analyses that we considered scientifically appropriate, and designed, implemented, and interpreted the resulting experiments ourselves.

We also used generative AI tools to assist with the implementation of some functions based on algorithms and implementation logic specified by us. We manually reviewed and tested all AI-assisted code to verify its correctness and made any required revisions. Generative AI tools were not used to fabricate experimental data, results, citations, or scientific claims. We have reviewed all AI-assisted work and take responsibility for the final content of this paper, including all text, claims, experimental results, and code produced with the aid of generative AI.

REFERENCES

Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9297–9307, 2019.

Anh-Quan Cao and Raoul De Charette. Monoscene: Monocular 3d semantic scene completion. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3981–3991. IEEE, 2022.

Xiyue Guo, Jiarui Hu, Junjie Hu, Hujun Bao, and Guofeng Zhang. Sgformer: Satellite-ground fusion for 3d semantic scene completion. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11929–11938. IEEE, 2025.

Jiawei Hou, Xiaoyan Li, Wenhao Guan, Gang Zhang, Di Feng, Yuheng Du, Xiangyang Xue, and Jian Pu. Fastocc: Accelerating 3d occupancy prediction by fusing the 2d bird's-eye view and perspective view. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 16425–16431. IEEE, 2024.

Yuanhui Huang, Wenzhao Zheng, Yunpeng Zhang, Jie Zhou, and Jiwen Lu. Tri-perspective view for vision-based 3d semantic occupancy prediction. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9223–9232. IEEE, 2023.

Haoyi Jiang, Tianheng Cheng, Naiyu Gao, Haoyang Zhang, Tianwei Lin, Wenyu Liu, and Xinggang Wang. Symphonize 3d semantic scene completion with contextual instance queries. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20258–20267. IEEE, 2024.

Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 12689–12697. IEEE, 2019.

Yiming Li, Zhiding Yu, Christopher Choy, Chaowei Xiao, Jose M Alvarez, Sanja Fidler, Chen Feng, and Anima Anandkumar. Voxformer: Sparse voxel transformer for camera-based 3d semantic scene completion. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9087–9098. IEEE, 2023.

Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: learning bird's-eye-view representation from lidar-camera via spatiotemporal transformers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(3):2020–2036, 2024.

Enyu Liu, En Yu, Sijia Chen, and Wenbing Tao. Disentangling instance and scene contexts for 3d semantic scene completion. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 26999–27009. IEEE, 2025.

Wenze Liu, Hao Lu, Hongtao Fu, and Zhiguo Cao. Learning to upsample by learning to sample. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6004–6014. IEEE, 2023.

Gyeongrok Oh, Sungjune Kim, Heeju Ko, Hyung-gun Chi, Jinkyu Kim, Dongwook Lee, Daehyun Ji, Sungjoon Choi, Sujin Jang, and Sangpil Kim. 3d occupancy prediction with low-resolution queries via prototype-aware view transformation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17134–17144. IEEE, 2025.

Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1874–1883, 2016.

Pin Tang, Zhongdao Wang, Guoqing Wang, Jilai Zheng, Xiangxuan Ren, Bailan Feng, and Chao Ma. Sparseocc: Rethinking sparse latent representation for vision-based semantic occupancy prediction. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15035–15044. IEEE, 2024.

Jiabao Wang, Zhaojiang Liu, Qiang Meng, Liujiang Yan, Ke Wang, Jie Yang, Wei Liu, Qibin Hou, and Ming-Ming Cheng. Opus: occupancy prediction using a sparse set. *Advances in Neural Information Processing Systems*, 37:119861–119885, 2024a.

Jiaqi Wang, Kai Chen, Rui Xu, Ziwei Liu, Chen Change Loy, and Dahua Lin. Carafe: Content-aware reassembly of features. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3007–3016. IEEE, 2019.

Xiaofeng Wang, Zheng Zhu, Wenbo Xu, Yunpeng Zhang, Yi Wei, Xu Chi, Yun Ye, Dalong Du, Jiwen Lu, and Xingang Wang. Openoccupancy: A large scale benchmark for surrounding semantic occupancy perception. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17804–17813. IEEE, 2023.

Yuqi Wang, Yuntao Chen, Xingyu Liao, Lue Fan, and Zhaoxiang Zhang. Panoocc: Unified occupancy representation for camera-based 3d panoptic segmentation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17158–17168. IEEE, 2024b.

Yi Wei, Linqing Zhao, Wenzhao Zheng, Zheng Zhu, Jie Zhou, and Jiwen Lu. Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21672–21683. IEEE, 2023.

Yujie Xue, Ruihui Li, Fan Wu, Zhuo Tang, Kenli Li, and Mingxing Duan. Bi-ssc: Geometric-semantic bidirectional fusion for camera-based 3d semantic scene completion. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20124–20134. IEEE, 2024.

Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018.

Zhiwen Yang and Yuxin Peng. Multi-resolution alignment for voxel sparsity in camera-based 3d semantic scene completion. *IEEE Transactions on Image Processing*, 35:1771–1785, 2026.

Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *2010 IEEE Computer Society Conference on computer vision and pattern recognition*, pp. 2528–2535. IEEE, 2010.

Yunpeng Zhang, Zheng Zhu, and Dalong Du. Occformer: Dual-path transformer for vision-based 3d semantic occupancy prediction. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9399–9409. IEEE, 2023.

Sicheng Zuo, Wenzhao Zheng, Yuanhui Huang, Jie Zhou, and Jiwen Lu. Pointocc: Cylindrical tri-perspective view for point-based 3d semantic occupancy prediction. *arXiv preprint arXiv:2308.16896*, 2023.

A Appendix

A.1 Definition of Category-Grouped Evaluation Metrics

These metrics are computed as the average IoU over predefined category groups, enabling a more fine-grained evaluation beyond overall performance. Formally, given a category group $\mathcal{G}$, the group-wise IoU is defined as:

$$\mathrm{IoU}_{\mathcal{G}} = \frac{1}{|\mathcal{G}|} \sum_{c \in \mathcal{G}} \mathrm{IoU}_c, \tag{15}$$

where IoU_c denotes the IoU of category c.

We divide the 19 non-empty semantic categories into three groups based on structural complexity, boundary ambiguity, and sample imbalance, aiming to enable a more reasonable evaluation and a more accurate analysis of the training results.

ComplexIoU measures performance on categories that typically occupy extended spatial regions, exhibit irregular structures, or have ambiguous semantic boundaries. This group includes road (15.30%), sidewalk (11.13%), parking (1.12%), other-ground (0.56%), terrain (9.17%), vegetation (39.30%), and trunk (0.51%), with the percentages indicating the proportion of each category in the dataset.

TailIoU focuses on long-tail categories with limited samples (all below 0.1%), where reliable feature learning is challenging. This group includes bicyclist (0.07%), person (0.07%), motorcyclist (0.05%), bicycle (0.03%), motorcycle (0.03%), and traffic-sign (0.08%).

RegularIoU evaluates categories with relatively regular geometric structures and well-defined shapes. This group includes car (3.92%), building (14.10%), truck (0.16%), fence (3.90%), pole (0.29%), and other-vehicle (0.20%).

A.2 Additional Analysis of Upsampling Feature Responses

A.2.1 Definition of the Feature-Response Variation Metric

The response measure and spatial CV reported in the main paper are defined as follows. For a frame containing N active voxels, let $\mathbf{x}_n \in \mathbb{R}^C$ denote the feature vector of the n-th active voxel, where C is the number of feature channels. Its root-mean-square (RMS) response is defined as

$$R_n = R(\mathbf{x}_n) = \sqrt{\frac{1}{C}\sum_{j=1}^{C} x_{n,j}^2}, \tag{16}$$

where j indexes the feature channels.

Let μ_R and σ_R denote the mean and standard deviation, respectively, of the voxel responses $\{R_n\}_{n=1}^{N}$ within the frame. The spatial coefficient of variation is then defined as

$$\mathrm{CV} = \frac{\sigma_R}{\mu_R} \times 100\%. \tag{17}$$

The measurements are taken after group normalization at the second sparse upsampling stage over 50 uniformly sampled SemanticKITTI validation frames, and the reported CV is averaged over these frames. The ratio in Table 4 is computed as the mean CV of each method divided by that of sparse transposed convolution.

A.2.2 Feature-Response Visualization Setup and Analysis

Fig. 4 compares the second-stage upsampling outputs before feature fusion. Both outputs use the same grid resolution and projected sample locations. Region A covers a car and its surrounding area, while regions B and C correspond to vegetation and ground areas, respectively. Compared with sparse transposed convolution, 3S exhibits clearer local variation in feature-response magnitudes within these regions. Each response map is independently min–max normalized; the visualization therefore compares relative spatial patterns rather than absolute response magnitudes or semantic prediction accuracy.

A.3 Definitions of Softmax- and Sigmoid-Based Feature Allocation

Let $\mathbf{H}_n \in \mathbb{R}^{r^3 \times C_{\mathrm{out}}}$ denote the candidate child representations of the n-th parent voxel. We apply mean pooling along the child dimension and feed the resulting feature into a two-layer MLP. The MLP produces $\mathbf{a}_n \in \mathbb{R}^{r^3}$, where $a_{n,q}$ is the allocation score predicted for the q-th candidate child voxel. We convert these scores into weights using either Softmax or Sigmoid.

The first variant applies Softmax across the r^3 child positions:

$$g_{n,q}^{\mathrm{softmax}} = \frac{\exp(a_{n,q})}{\sum_{j=1}^{r^3}\exp(a_{n,j})}, \qquad q = 1, \ldots, r^3. \tag{18}$$

The resulting weights sum to one, introducing competition among the candidate child positions.

The second variant applies Sigmoid independently to each allocation score:

$$g_{n,q}^{\mathrm{sigmoid}} = \frac{1}{1+\exp(-a_{n,q})}, \qquad q = 1, \ldots, r^3. \tag{19}$$

Unlike Softmax, Sigmoid does not impose a sum-to-one constraint, allowing multiple child positions to retain relatively large responses simultaneously. In both variants, each weight $g_{n,q}$ is applied to its corresponding candidate representation $\mathbf{H}_{n,q,:}$.

A.4 Effect of Additional LayerNorm in ER-LM

As shown in Table 9, without $\mathcal{L}_{\text{cwe}}$, adding LayerNorm to the LSTM output before projection improves occupancy IoU but reduces semantic mIoU. This may reflect a trade-off: normalizing memory-feature magnitudes can help distinguish occupied from empty regions while weakening magnitude cues useful for distinguishing semantic classes. With $\mathcal{L}_{\text{cwe}}$, omitting this additional LayerNorm yields higher mIoU, TailIoU, and RegularIoU, at a small cost in ComplexIoU. We therefore omit this additional LayerNorm in our final configuration.

Table 9: Effect of additional LayerNorm in ER-LM. Bold numbers indicate the best result in each column.

Method	IoU	mIoU	ComplexIoU	TailIoU	RegularIoU
ER-LM	36.09	12.53	22.96	1.32	11.57
ER-LM + LayerNorm	**36.53**	12.45	23.05	1.26	11.26
ER-LM + LayerNorm + $\mathcal{L}_{\text{cwe}}$	36.37	12.47	**23.28**	1.51	10.82
ER-LM + $\mathcal{L}_{\text{cwe}}$	36.45	**12.82**	23.23	**1.64**	**11.84**

Note. All configurations were re-run under the same setting, which differs slightly from Table 2.

A.5 Evaluation Protocol and Semantic Transition Metrics

Both models are evaluated on the same 815 SemanticKITTI validation frames using all nine decoder outputs. Predictions are evaluated at the ground-truth voxel resolution; invalid labels are excluded, and repeated samples introduced by distributed sampling are counted only once.

For a source layer a and a target layer $b > a$, we compare semantic predictions at the same voxel locations. For each non-empty ground-truth class c, let N_c^{CC}, N_c^{CW}, N_c^{WC}, and N_c^{WW} count correct-to-correct, correct-to-wrong, wrong-to-correct, and wrong-to-wrong transitions, respectively. For brevity, we omit the layer-pair indices $a \to b$ from these counts. Correctness requires an exact match to the ground-truth semantic label. Counts are pooled over the validation set before computing the class-wise degradation and correction rates:

$$D_c^{a\to b} = \frac{N_c^{\text{CW}}}{N_c^{\text{CC}} + N_c^{\text{CW}}}, \tag{20}$$

$$R_c^{a\to b} = \frac{N_c^{\text{WC}}}{N_c^{\text{WC}} + N_c^{\text{WW}}}. \tag{21}$$

Degradation measures the fraction of initially correct predictions that become incorrect, whereas correction measures the fraction of initially incorrect predictions that become correct. To give each category equal weight, we define

$$\text{mD}^{a\to b} = \frac{100}{|\mathcal{C}_D^{a\to b}|} \sum_{c\in\mathcal{C}_D^{a\to b}} D_c^{a\to b}, \tag{22}$$

$$\text{mR}^{a\to b} = \frac{100}{|\mathcal{C}_R^{a\to b}|} \sum_{c\in\mathcal{C}_R^{a\to b}} R_c^{a\to b}. \tag{23}$$

For each metric and layer pair, the class set contains only non-empty categories with a non-zero denominator in both models. Undefined rates are excluded rather than set to zero. Thus, each row compares the models over identical eligible classes, although the mD class set can vary across rows. Lower mD and higher mR indicate more favorable semantic preservation and correction, respectively.