Dynamic LiDAR-Camera Fusion

Benjamin Thérien
Department of Computer Science
Waterloo University
Waterloo, ON N2L 3G1
`btherien@uwaterloo.ca`

Abstract

This paper studies four novel approaches to fusing LiDAR and camera modalities for 3D object detection. We follow a line of previous work which decorates point clouds with images features at an early stage. In contrast to previous work, however, our approach incorporates image features at multiple different scales and routing functions designed to appropriately select among them. Specifically, we apply mixture of experts and temperature softmax approaches to dynamically select image features at run-time. We hypothesize that conditioning the selection of image features on point cloud features can effectively model the complex interactions patterns between both modalities. While our approaches do not significantly outperform current solutions, they show improvements to cyclist detection and shed light on a mixture of experts approach to LiDAR-camera fusion, which, to our knowledge, has never been attempted before.

1 Introduction

3D object detection is a heavily researched area in computer vision due in part to its necessity for autonomous driving and wide applicability to robotics in general. Typical autonomous vehicle perception systems process real-time data from both LiDAR and camera sensors. While monocular images contain dense texture information (even at a distance), they lack precise depth information. In contrast, LiDAR point clouds are inherently sparse (even sparser at a distance), but contain precise depth information which is essential for accurate 3D detection. Due to this discrepancy, the space of 3D object detectors is dominated by architectures which leverage the point cloud modality [11, 40, 33, 35, 34]. However, LiDAR is not without its drawbacks. Indeed, faraway objects clearly visible in 2D images tend to have very few LiDAR points if any at all. Consequently, LiDAR-only 3D detectors cannot reliably detect distant objects. In contrast, image detectors can, but fail to accurately estimate depth. Existing work [27, 32, 28, 19, 9, 10, 20, 21, 39, 36, 12, 37, 38, 31] attempts to alleviate the disadvantages of each modality by merging them together. These multimodal methods hope that the dense texture information available in images can improve overall detection in sparse point clouds.

Realizing the promise of LiDAR-Camera fusion, however, is not so simple. Making improvements over precisely tuned single-modal baselines is challenging due to the greedy nature of gradient-based learning, the 2D v.s. 3D modality miss-match, and the nature of the inter-modality information gap. Wu et al. [30] explain that multimodal learners are sometimes greedy with respect to one modality, especially when that modality is easier to learn from. In these cases, the multimodal learner may simply discard the information of secondary modalities, rendering this additional information useless. We hypothesize that LiDAR is the dominant modality in our case, evidenced by the strong performance of LiDAR-only versus camera-only 3D detectors. Another difficulty of merging point clouds and images is the inherent modality miss-match: a point cloud lives in 3D, whereas an image is the projection of 3D points (really photons emanating from 3D points) onto a 2D image plane. Though it is possible to establish a direct correspondence between image and LiDAR points using the camera's projection matrix, the best way to leverage this correspondence for multimodal fusion is

Preprint. Under review.

still an open question. This question is of particular relevance when no corresponding LiDAR points are available for useful image features (e.g. a distant pedestrian which could be detected from the image is not perceived by the LiDAR). We refer to this problem as the inter-modality information gap. Without depth information from the LiDAR point cloud, it is impossible to tell how far away candidate detections in the image are, however, if we fail to incorporate the RGB information in these situations, our multimodal detectors will struggle to outperform LiDAR-only baselines.

RetinaNet Layer	Feature size	Receptive Field
Input Image	480×1568	1×1
Block 1	120×392	35×35
Block 2	60×196	99×99
Block 3	30×98	227×227
Block 4	15×49	419×419
FPN Maxpool	8×25	419×419

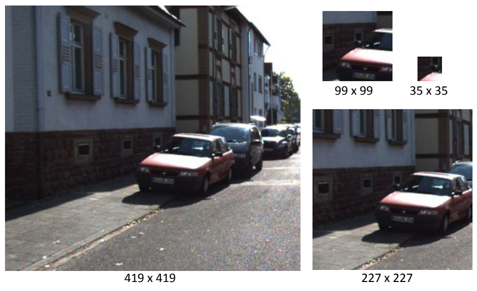


Figure 1: **Receptive fields of different image features.** The feature pyramid network we use outputs feature maps at 5 different sizes. The table (left) shows the size of these features maps and their receptive fields. The figure (right) shows a sample image from the KITTI dataset cropped to the size of the different receptive fields.

Several existing approaches in the literature attempt to tackle exactly these problems. In their prominent approach, [36] explicitly address the problem of the 2D v.s. 3D modality mismatch by augmenting LiDAR point clouds using 2D detections. Any detection containing at least one corresponding LiDAR point is used to augment the point cloud with additional points at a similar depth. Other approaches [39, 20, 21, 31] explicitly address the inter-modality information gap by using two-stage architecture which makes object proposal based on both modalities and extracts ROIs for a final prediction step based on features from both modalities. These methods tend to fuse the modalities at a later stage. In contrast, a third set of approaches forego explicit extractions schemes, assuming, instead, that the network can learn to leverage the complex interactions between both modalities provided features are fused in a principled way. These methods propose to decorate the point cloud with image features at an early stage. In the following work, we add to this growing body of literature by studying fusion mechanisms that select image features at different scales. Specifically, we propose *four different extensions* to the fusion schemes of [27]. Our proposed approaches leverage different routing mechanisms from the conditional computation literature, allowing each LiDAR point to select the scale of the image features it is fused with. We hypothesize that such routing mechanisms should be able to circumvent the problems of LiDAR sparsity by learning to pay attention to image features at an appropriate scale. For instance, it may be helpful for faraway points to select image features at smaller scales since distant objects appear smaller in images, while the opposite is true for points nearby the LiDAR. Our contributions are threefold, we propose four novel approaches to fusing point cloud and image features in a voxel-based encoding scheme, we provide an empirical evaluation of the methods proposed, and detailed analysis and discussion of the results. We note that to our knowledge, we are the first-ever approach to apply a mixture of experts routing approach to fuse image and point cloud modalities.

2 Literature Review

2.1 2D Detection

Object detectors for two dimensional images can be grouped into single-stage [23, 16, 2] and two-stage detectors [6, 5, 24, 13, 14]. Single-stage detectors directly predict the presence of objects in an image by leveraging anchors [16, 23, 13] or keypoint heatmaps [2]. On the other hand, two-stage methods first predict candidate object locations (e.g., with anchors) and then extract regions of interest at each candidate location to refine predictions. These methods tend to be slower than their single-stage counterparts but were known to be slightly more accurate until the introduction of the focal loss [14]. In their foundational work, Lin et al. [14] introduce the focal loss, a variant of the cross-entropy criterion which down-weights easily classified examples, allowing for a greater focus on hard negatives. The paper combines this novel criterion with a modified version of a feature pyramid

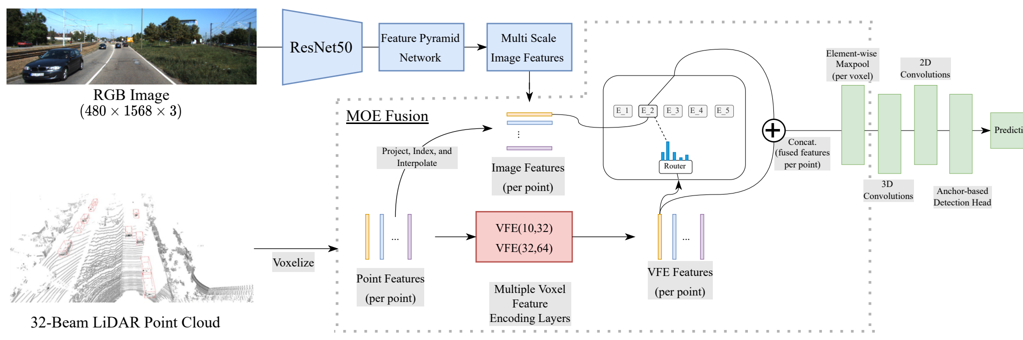

ResNet50
Feature Pyramid Network
Multi Scale Image Features
RGB Image
(480 × 1568 × 3)
MOE Fusion
Project, Index, and Interpolate
Image Features (per point)
E_1
E_2
E_3
E_4
E_5
Router
Concat. (fused features per point)
Element-wise Maxpool (per voxel)
3D Convolutions
2D Convolutions
Anchor-based Detection Head
Predictions
Voxelize
Point Features (per point)
VFE(10,32)
VFE(32,64)
Multiple Voxel Feature Encoding Layers
VFE Features (per point)
32-Beam LiDAR Point Cloud

Figure 2: **Overview of the proposed architecture.** The diagram shows our architecture for Mixture of Experts fusion. We note that other fusion schemes were left out for clarity, but are outlined below. The architecture takes one RGB image and one LiDAR point cloud as input. The coordinates of each point are used to establish a correspondence between the points and image coordinates. VFE features are fed to a router which determines which expert should process corresponding images features. Processed image features are then concatenated to VFE features and the element-wise maximum is taken to represent a voxel. Finally, the features are processed by standard backbones from the literature [33] and predictions are made via a single-stage detector.

network[13], called retinaNet. Feature pyramid networks [13] first extract features at different scales from a backbone network (e.g. the stages of a ResNet [8]), then upsample the image features from the coarsest scale to match outputs of each scale. Features are merged in this top-down fashion and laterally with 1×1 convolutions. The result is one large feature map with features at different scales.

2.2 LiDAR-based 3D Detection

VoxelNet [40] and Pointpillars [11] are foundational voxel-based and pillar-based LiDAR detectors respectively. The voxel-based encoding scheme in [40] produces 4-dimensional tensors, which must be processed with 3d convolutions, making VoxelNet slower, but more accurate than Pointpillars [11] which only uses 2d convolutions. In SECOND [33], authors propose an efficient sparse convolution algorithm for processing voxelized point clouds, a data augmentation technique that involves extracting point cloud objects using their ground truth bounding boxes and placing them in new scenes, and an improved region proposal network architecture. HVNet [34] explore a hybrid VoxelNet architecture that encodes local and progressively more global features at each voxel location. [26] propose PV-RCNN, a two-stage method, where a VoxelNet-based architecture with sparse 3d convolutions is used to generate region proposals and a point net [22] extraction module is used to gather information for the selected ROIs. [35] propose Centerpoint, a VoxelNet-based architecture with a novel detection head that predicts per-class heatmaps in BEV where local maximums are considered to be detections.

2.3 Multimodal 3D Detection

Multimodal 3d detection methods generally fuse LiDAR and camera modalities to improve detection performance. These approaches can be grouped into a few general categories: multi-stage methods [10, 20, 39], point projection correspondence methods [27, 32, 28, 19, 9], multi-scale fusion approaches [12, 37], and point cloud augmentation techniques [36]. Multi-stage methods attempt to extend the proposal and refinement paradigm of 2d detectors to 3d multimodal methods. In AVOD [10], authors first predict object proposals over fused feature maps before extracting multimodal ROIs and fusing them for a final prediction step. Point projection correspondence methods take advantage of the camera's projection matrix to establish a correspondence between LiDAR and image features. Given this correspondence, these techniques decorate the LiDAR point cloud with corresponding image features [27, 9, 28], segmentation maps [32], or object probabilities [19]. In their multiscale fusion approaches [12] and [37] progressively fuse image and LiDAR features during the forward pass, allowing for many interactions between the LiDAR and image features maps at different stages of encoding. Alternatively, [36] add additional points to the LiDAR point cloud based on an image level segmentation. Their technique allows the dense image information to enhance the LiDAR point clouds in places where it may be sparse, and essential characteristics of multimodal systems which maximally benefit from the integration of both modalities.

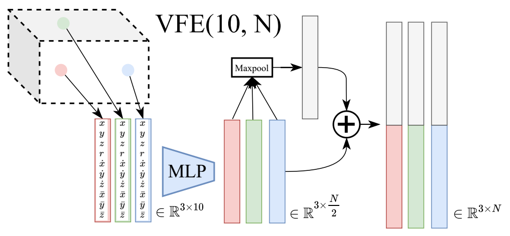

VFE(10, N)
Maxpool
MLP
$\in \mathbb{R}^{3\times 10}$
$\in \mathbb{R}^{3\times \frac{N}{2}}$
$\in \mathbb{R}^{3\times N}$

Figure 3: **VFE Layer Architecture.** We outline the architecture of one voxel feature encoding (VFE) layer as proposed in [40] (though our feature vectors, seen here, differ). The diagram shows the processing of features from one voxel; features describing each point are encoded with an MLP, element-wise maxpooling is applied, and the resulting max-pooled features are concatenated to each point-wise feature to create the point-wise output.

2.4 Conditional computation

Shazeer et al. [25] introduce a conditional computation scheme for routing inputs to different expert MLPs inside a stacked LSTM architecture. The paper demonstrated the feasibility of training a routing network, by selecting k experts per input for computation at any given MOE layer. They also introduce a noise input to the gating and a loss term to encourage exploration of the router. [3] leverage a similar algorithm to the one introduced in [25], except they are the first to show this scheme can work with a single expert. Our work

3 Preliminaries

In the following section, we give a brief overview of existing components from the literature that we use as part of our architecture. This section provides the necessary details to understand how our proposed fusion schemes integrate into the 3D detection architecture we use.

3.1 Pretrained image detector

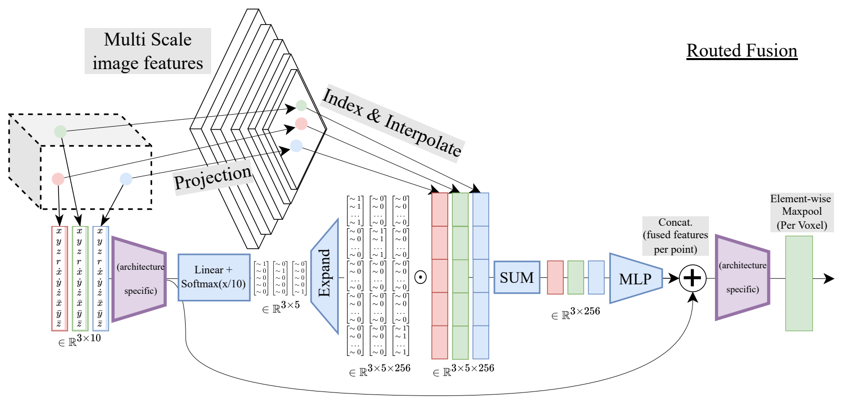
To encode the image modality with rich features at different scales, we utilize a modified version of RetinaNet [14] which uses a ResNet50 backbone with a feature pyramid network (FPN) and an extra maxpooling layer atop the last ResNet50 block. This network was pre-trained on the COCO dataset[15] as part of a faster R-CNN object detection pipeline. In our work, the five features maps (see fig. 1) output by the FPN are used to extract image features to be fused with point cloud features.

3.2 Voxelnet, Voxelization, and Voxel Feature Encoding

VoxelNet [40] is a 3D object detector for processing point clouds. It first *voxelizes* the point cloud, dividing it into a three-dimensional grid of evenly spaced voxels. A representation of each non-empty voxel is computed based on the points that it contains. Specifically, let $\mathcal{V}$ denote a particular voxel, then each point $p \in \mathcal{V}$ is assigned a feature vector f based on its coordinates in 3d space, voxel offset, and centroid offset. We represent these features as $f_p = (x, y, z, r, \dot{x}, \dot{y}, \dot{z}, \bar{x}, \bar{y}, \bar{z})^T$, where x, y, z, r represent the 3d coordinates and reflectance of the point obtained from the LiDAR sensor. $(\dot{x}, \dot{y}, \dot{z})^T$ and $(\bar{x}, \bar{y}, \bar{z})^T$ are the offset of point p from the center of voxel $\mathcal{V}$ and its centroid respectively. We take the centroid to be the coordinate-wise mean of each point in the voxel. Using the feature representations of each point, representations for each non-empty voxel are computed using voxel feature encoding (VFE) layers as illustrated in fig. 3. VFE layers encode $f_p \forall p \in \mathcal{V}$ with an MLP, then apply an element-wise maxpooling operation to the resulting features of each point, and concatenate this global feature to each of the point-wise features. In [40], this operation is performed twice for each point in each voxel to obtain 128 dimensional feature vectors for each point. Finally, an element-wise maxpooling operation is applied to the point-wise feature vectors in each voxel to obtain the final voxel representation. After the voxel feature encoding step, the point cloud is now represented by a sparse 4D tensor ($C \times D \times W \times H$), where $C = 128$ is the voxel feature size.

3.3 Point fusion and Voxel fusion

In their multimodal extension of the VoxelNet architecture, Sindagi et al. [27] propose two mechanism for fusing image and LiDAR features: *point fusion* and *voxel fusion*. Each method establishes a correspondence between LiDAR points and camera features by projecting the LiDAR points onto the image and subsequently interpolating feature maps at corresponding pixel locations from their VGG16 image backbone (in contrast, we utilize a ResNet50 backbone with an FPN). The point fusion



Multi Scale image features
Routed Fusion
Index & Interpolate
Projection
(architecture specific)
Linear + Softmax(x/10)
$\in \mathbb{R}^{3\times 10}$
$\in \mathbb{R}^{3\times 5}$
Expand
$\in \mathbb{R}^{3\times 5\times 256}$
$\in \mathbb{R}^{3\times 5\times 256}$
SUM
$\in \mathbb{R}^{3\times 256}$
MLP
Concat. (fused features per point)
(architecture specific)
Element-wise Maxpool (Per Voxel)

Figure 4: **Routed Fusion approach.** The diagram depicts point features are selecting image features at different scales, where a softmax with temperature 0.1 steers the operation to only selects features at one scale.

approach simply concatenates the interpolated feature vectors for each point p in the point cloud. The resulting feature vector $(f_p, \mathcal{I}_p)^T$ comprises both image and point features and is fed to VFE layers for encoding, following the VoxelNet architecture thereafter. In contrast, the voxel fusion approach fuses image and point features after the VFE stage. Instead of using point coordinates to extract image features, they extract them using the coordinates of each non-empty voxel. The dimension of these features is then reduced using an MLP and concatenated to the voxel feature vector computed from voxel feature encoding of the points. To maintain consistency with VoxelNet, both methods are made to produce voxel features $\in \mathbb{R}^{128}$. To facilitate comparison we also follow this convention.

3.4 Convolutional Middle Layers

After voxel feature encoding, voxel-based architectures must process a sparse 4D tensor to aggregate the local information at each non-empty voxel. 3D convolutions must therefore be used to reduce the 4D tensor to a 3D BEV feature map. We follow the general approach of [33], using a mixture of submanifold [7] and sparse 3D convolutions to aggregate the local features of each voxel and reduce a dimension of the features map. After reducing the feature map's dimensionality, we further encode the feature map using a series of 2d convolutions which downsample the feature map twice using strided convolutions, creating three BEV feature maps at different scales. Finally, a deconvolution layer is applied to the two feature maps at larger scales to spread out the coarser-grained features. Finally, the three feature maps, now of the same size, are concatenated before being input to the region proposal network for prediction. We note that this step also follows the architecture from [33].

3.5 Region Proposal Network

Following single shot detectors from the 2D and 3D detection literature [16, 14, 33], we use three heads to classify, regress bounding boxes, and regress the angle of anchors at each detection position. Following [33] we use $0.8 \times 0.6 \times 1.73$ m, $1.76 \times 0.6 \times 1.73$ m, and $3.9 \times 1.6 \times 1.56$ m anchors for pedestrian, cyclist, and car detection respectively. These sizes are selected based on the mean dimensions of each class in the KITTI training set. For each anchor position, the classification heads predict a softmax distribution over the classes, a softmax distribution over the direction, and regresses the x, y, z coordinates of the object, height, width, length, and yaw angle. Focal loss [14] (eq. 1) is used for object classification with $\gamma = 2$ the smooth L1 loss [5] (2) is used for regression with $\beta = 1/9$, and cross-entropy is used for direction classification. For brevity, we leave further detail of the regression targets to the avid reader and refer them to [33].

$$FL(p_t) = -(1 - p_t)^{\gamma} \log(p_t) \tag{1}$$

$$\text{smooth}_{L_1}(x, y) = \begin{cases} \frac{(x-y)^2}{2\cdot\beta} & \text{if } |x - y| < \beta \\ |x - y| - \frac{\beta}{2} & \text{otherwise} \end{cases} \tag{2}$$

4 Routed Fusion to Different Scales

In the following section, we outline our four novel multimodal fusion approaches in detail. The two first approaches leverage the same temperature softmax routing mechanism, but differ in fusion location. On the other hand, both mixture of experts (MOE) approaches leverage the MOE paradigm from the NLP literature [25, 3] to process different image features. While these conditional computation approaches are all architecturally different, they share one common principle: point features are always used to select image features.

4.1 Softmax Routing

Both our *Routed* approaches leverage the same temperature softmax trick to select between image features at five different image scales (see fig. 1). Specifically, we use a routing multilayer perceptron (MLP) ϕ combined with a softmax and temperature parameters τ (see eq 3) to predict a distribution over the image feature scales from the point features f_p. We set $\tau = 10$ to mimic a discrete distribution over the five different scales while maintaining differentiability. This can be seen as imposing a prior on the network, biasing it towards solutions that only select one scale. As seen in fig. 4 the temperature softmax prediction is expanded to match the corresponding image feature dimensions, a Hadamard product is taken, and the features are summed over the scale channel, leaving only the selected features. These features are then encoded by an MLP before being *fused* via concatenation to their corresponding point features. We note that we monitor the routing distributions during training to ensure that they are approximately one-hot.

$$softmax(\phi(f_p)/\tau)_i = \frac{e^{\phi(f_p)_i/\tau}}{\sum_{i=1}^{5} e^{\phi(f_p)_i/\tau}} \tag{3}$$

4.2 Routed Point Fusion

Our *Routed Point Fusion* approach uses softmax routing to select the image feature scale from the initial point features, otherwise processing features similarly to *point fusion* [27]. In detail, we obtain the initial feature vector $f_p \in \mathbb{R}^{10}$ for each point p in the point cloud and feed these to the routing MLP. The selected image features $\in \mathbb{R}^{256}$ are reduced to $\mathbb{R}^{16}$ and concatenated. The fused features are then processed by two more voxel encoding layers (VFE$(26, 32)$ and VFE$(32, 128)$) before they are finally maxpooled element-wise for each voxel to produce the individual voxel features. The entire procedure is depicted in 4 if we suppose that the first purple trapezoid is the identity function and the second purple trapezoid represents two VFE layers: VFE$(26, 32)$ and VFE$(32, 128)$. These VFE dimensions follow previous work [27].

4.3 Routed Voxel Fusion

Our *Routed Point Fusion* approach uses softmax routing to select between image feature scales from the voxel features, otherwise processing features similarly to *voxel fusion* [27] but with one slight modification. Specifically, we process points with two voxel feature encoding layers (VFE$(10, 32)$ and VFE$(32, 64)$) before feeding these 64 dimensional feature vectors to the routing MLP. However, instead of selecting image features at the pixel location corresponding to the point, this approach selects them at the pixel location corresponding to the center of the voxel. The selected image features $\in \mathbb{R}^{256}$ are reduced to $\mathbb{R}^{64}$ and concatenated. The fused features are then maxpooled per-voxel (this differs from [27] which maxpool their features before fusing them via concatenation).

4.4 MOE Fusion

Our *MOE Fusion* approach uses a router to select between experts, which are in fact linear projection layers. In contrast to softmax routing, which biases the network towards selecting features at a particular scale, the mixture of experts approach is a generalization of softmax routing allowing each expert to learn an appropriate feature selection function. In detail, we process points with two voxel feature encoding layers (VFE$(10, 32)$ and VFE$(32, 64)$) before feeding these 64 dimensional feature vectors to the routing MLP. This MLP predicts a softmax distribution over the experts. We follow the top-1 routing scheme of Fedus et al. [3], selecting only the top-weighted expert. The corresponding image features at each scale are then concatenated (a 1280 dimensional feature vector) and fed to the experts. Our routing implementation works by gathering features computed by t the selected expert. Finally, the image features output from the selected expert are concatenated to their corresponding voxel features and maxpooled element-wise.

4.5 Load Balanced MOE Fusion

Our *Load Balanced MOE Fusion* is an attempt to impose more structure on the network by encouraging it to route the same number of points to each expert. To accomplish this we add a simple load balancing term $\mathcal{L}_{LB} = \alpha N \cdot \sum_{i=1}^{N} f_i \cdot P_i$ to the loss, following previous work [3]. Where N is the number of experts, f_i is the fraction of points that select expert i, P_i is the fraction of the router probability allocated to expert i, and $\mathcal{P}$ is the set of all point features from the batch. $\mathcal{L}_{LB}$ is minimized when routing is uniform over all experts. Though [3]'s motivation for load balancing is mostly for efficiency (for evenly scaling compute across nodes in distributed training), we hypothesize that it could have a beneficial effect on performance by providing more supervisory signal to each expert. Other than the load balancing term, this fusion strategy is identical to *MOE Fusion*. We note that expert routing was monitored during training and that under the load balancing loss, the router quickly selects to route the same number of tokens to each expert.

$$f_i = \frac{1}{|\mathcal{P}|} \cdot \sum_{x \in \mathcal{P}} \mathbb{1}\{MLP(x), i\} \ \ ; \ \ P_i = \frac{1}{|\mathcal{P}|} \cdot \sum_{x \in \mathcal{P}} MLP(x) \tag{4}$$

5 Empirical Evaluation

5.1 The KITTI Dataset

The KITTI dataset [4] contains $7,481$ training data points and $7,518$ testing data points. We further split the training set into a smaller training set of $3,712$ datapoints and a validation set of $3,769$ data points as in previous work [33]. Each data point contains a LiDAR point cloud and two RGB images taken from a pair of stereo cameras. In our work, we discard images from the right camera and only utilize only images from the left camera and the LiDAR point could. KITTI's 3D object detection benchmark contains three different evaluation levels: Easy (fully visible objects with min. bounding box height 40 px.), Moderate (partly occluded objects with min. bounding box height 25 px.), and hard(includes difficult to see objects with min. bounding box height 25 px.)[4]. The annotated objects are cars, pedestrians, and cyclists. We note that the dataset contains many more cars and pedestrians than cyclists.

5.2 Implementation & Training Procedure

Our implementation re-uses many components available in the mmdetection3D framework [1]. For comparison purposes, all models keep default hyperparameters choices from the implementation of multimodal VoxelNet in [1], except for the fusion-specific parameters which we outlined previously. All models were trained on a remote server using four Nvidia V100 GPUs. To deal with the imbalance in training classes, we follow the class-balanced grouping procedure outlined in [41]. Since the KITTI dataset only includes bounding boxes for objects visible to the camera we follow previous work [33], restricting the encoded points to lie in the $(-3, 1) \times (-40, 40) \times (0, 70.4)$m prism along the $z \times y \times x$ axes and use voxels of size $0.05 \times 0.05 \times 0.1$ m width, height, and depth respectively. All networks were trained for 40 epochs using the adamw optimizer [18] with a learning rate of .003 and the cosine annealing scheduler [17] with 1000 warmup iterations. We use a batch size of 5.

5.3 Baselines

To provide baseline results in our empirical study, we re-implement the point fusion approach from [27] and consider a LiDAR-only approach called 'Point Feature Baseline' and an image-only approach called 'Image Feature Baseline'. We note that the image feature baseline is not completely devoid of LiDAR features, as the initial point-wise feature vectors are used to select image features using softmax routing, however, instead of fusing point features to the image features this baseline simply outputs the selected image features. The rest of the architecture is identical.

6 Results

Tables 1, 2, and 3 show class-specific average precision and overall mean average precision results for our different approaches. The performance is evaluated on the standard KITTI validation set after the last epoch of training. To control for stochasticity, we train each detector $3+$ times using different seeds and report the mean and standard error over these trials.

BEV Results The BEV evaluation evaluates detection results for 2D bounding boxes in the bird's eye view of the scene. Consequently, this task tends to be easier than its 3D counterpart, which involves a third bounding box dimension. In the car-specific BEV results, our MOE fusion approach

	Car			Pedestrian			Cyclist		
Fusion Arch.	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Point Feature Baseline	88.85 ± .04	86.13 ± .06	84.64 ± .12	71.56 ± .12	67.26 ± .09	62.29 ± .12	75.18 ± .69	59.56 ± .73	55.21 ± .54
Image Feature Baseline	88.95 ± .23	85.49 ± .25	84.36 ± .08	69.90 ± .24	63.24 ± .64	60.25 ± .28	73.07 ± .37	58.10 ± .65	55.10 ± .74
Point Fusion Baseline	89.19 ± .09	86.27 ± .13	84.67 ± .10	71.35 ± .07	65.83 ± .55	61.80 ± .03	74.21 ± .84	59.10 ± .87	55.92 ± .36
MVX-Net (PF) [27]	89.6	84.8	78.6	N/A	N/A	N/A	N/A	N/A	N/A
SECOND [33]	89.96	87.07	79.66	N/A	N/A	N/A	N/A	N/A	N/A
Routed Point Fusion	88.47 ± .06	85.98 ± .19	84.54 ± .07	70.42 ± .48	64.43 ± .65	61.13 ± .35	75.22 ± .64	60.09 ± .54	56.10 ± .40
Routed Voxel Fusion	89.03 ± .25	86.34 ± .05	84.58 ± .06	70.73 ± .12	63.83 ± .23	61.39 ± .21	75.00 ± .95	60.32 ± .47	56.66 ± .79
MOE Fusion	89.44 ± .08	86.27 ± .31	84.71 ± .11	68.39 ± .66	62.55 ± .99	59.55 ± .59	73.54 ± .67	58.18 ± .33	54.76 ± .46
MOE Fusion LB	88.96 ± .37	85.37 ± .45	84.32 ± .20	69.08 ± .19	62.18 ± .24	59.73 ± .31	73.00 ± .55	56.43 ± 1.10	53.86 ± .62

Table 1: **BEV results on the KITTI Validation set.**

	Car			Pedestrian			Cyclist		
Fusion Arch.	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
Point Feature Baseline	87.02 ± .17	76.87 ± .10	75.21 ± .42	63.15 ± .79	59.17 ± .09	53.81 ± .08	72.51 ± .30	57.71 ± .28	53.24 ± .22
Image Feature Baseline	86.91 ± .25	76.52 ± .15	73.82 ± .17	63.12 ± .62	57.80 ± .34	53.48 ± .60	70.96 ± .41	55.45 ± .68	52.60 ± .28
Point Fusion Baseline	87.49 ± .12	77.06 ± .11	75.07 ± .52	63.28 ± .84	58.89 ± .46	55.16 ± .94	72.24 ± .64	56.72 ± .84	53.93 ± .12
Routed Point Fusion	86.65 ± .09	76.62 ± .15	74.95 ± .31	62.86 ± 1.01	58.06 ± .64	53.10 ± .41	73.79 ± .95	57.67 ± 1.03	54.67 ± .51
Routed Voxel Fusion	87.31 ± .35	77.02 ± .16	75.57 ± .11	62.21 ± .40	57.99 ± .49	53.29 ± .36	73.54 ± 1.23	58.07 ± .96	54.59 ± 1.02
MOE Fusion	87.45 ± .18	76.97 ± .18	74.69 ± .50	60.67 ± .71	56.22 ± .75	51.93 ± .49	71.11 ± 1.12	55.21 ± .58	52.86 ± .78
MOE Fusion LB	86.82 ± .48	76.48 ± .22	73.72 ± .24	61.20 ± .30	57.00 ± .35	52.24 ± .13	71.17 ± 1.06	54.48 ± .66	52.14 ± .62

Table 2: **3D results on the KITTI Validation set.**

performs the best of all our implementations but is bested by previous work in the easy and moderate evaluations. However, we note a great improvement over the previous work in the hard category but this improvement is the same across all our implementations and, thus, cannot be attributed to one particular fusion approach. For the pedestrian-specific BEV results, all approaches are outperformed by the Point Feature Baseline, showing that image features do not provide useful information for detecting pedestrians in BEV. For the cyclist-specific BEV results, our Routed Fusion approaches perform best, suggesting that biasing the network towards selecting features at a particular scale may be useful for cyclist detection. The overall BEV results show the Point Feature Baseline outperforms others on the easy and moderate tasks while being close to the best on hard. These mediocre results show the need for a more principled approach to LiDAR-Camera fusion and may be a symptom of the greedy nature of gradient-based learning. We note that out of our fusion approaches, both softmax routing approaches outperform their MOE counterparts due to their strong performance on small objects. This seems to suggest it is importance to select features at specific scales for detecting pedestrians and cyclists.

3D Results The 3D evaluation evaluates detection results for 3D bounding boxes in the scene. In the car-specific 3D results the point fusion baseline outperforms our approaches in all except the hard category, where Routed Voxel Fusion performs best. The story for 3D pedestrian-specific results differs from the BEV results with the Point Fusion Baseline only performing best on the moderate difficulty. In contrast to BEV results which show all fusion methods performing poorly, on the 3D benchmark the point fusion baseline performs best in easy and hard categories, showing that image features can be beneficial for more precisely localizing the height of objects. In the cyclist-specific 3D results our Routed Point Fusion and Routed Voxel Fusion approaches once again outperform the competition, reinforcing our belief that biasing the network to select features at one scale enables learning more discriminative representations for cyclists (perhaps due to the distinctive wheel shape). Finally, in contrast to the BEV results, overall 3D results are not dominated by the point fusion baseline, though it does do best on the moderate evaluation. Routed Point Fusion and the Point Fusion Baseline perform best on the 3D easy and hard evaluations respectively. This positive trend for fusion approaches shows that image features do provide beneficial information for estimating 3d bounding boxes.

Notable trends There are some overall trends which are not specific to 3D or BEV results. First and surprisingly, the Image Feature Baseline performs decently without any point features at all. However, it struggles to learn quickly compared to approaches that do incorporate point features (see fig. 5). This confirms our hypothesis that the image modality is harder to learn from. Second, MOE Fusion performs well on cars, but poorly on pedestrians and cyclists. Third, the load balancing term seems to decrease the performance of the MOE approach. Lastly, out of all our fusion approaches, Routed Voxel Fusion performs best.

7 Discussion

While we were successful in creating and implementing fusion mechanisms which select image features based on their corresponding point features, the proposed approaches did not significantly

	BEV			3D		
Fusion Arch.	Easy	Moderate	Hard	Easy	Moderate	Hard
Point Feature Baseline	**78.53** ± .27	**70.98** ± .23	67.38 ± .11	74.23 ± .36	**64.58** ± .13	60.75 ± .24
Image Features Baseline	77.31 ± .09	68.94 ± .30	66.57 ± .30	73.66 ± .18	63.26 ± .30	59.97 ± .26
Point Fusion Baseline	*78.25* ± .27	*70.40* ± .40	*67.46* ± .09	74.34 ± .10	64.22 ± .17	**61.38** ± .45
Routed Point Fusion	78.04 ± .26	70.17 ± .15	67.26 ± .08	**74.43** ± .58	64.12 ± .30	60.91 ± .18
Routed Voxel Fusion	*78.25* ± .35	70.16 ± .17	**67.54** ± .22	*74.35* ± .49	*64.36* ± .37	*61.15* ± .36
MOE Fusion	77.12 ± .27	69.00 ± .40	66.34 ± .25	73.08 ± .49	62.80 ± .36	59.83 ± .48
MOE Fusion LB	77.01 ± .21	67.99 ± .46	65.97 ± .17	73.06 ± .59	62.65 ± .19	59.36 ± .22

Table 3: **BEV and 3D mAP results on the KITTI Validation set.** The highest number is **bolded** and second highest it *italicized*.

improve upon previous work. In this section, we elaborate on the shortcomings of each of our approaches and explain how they could be improved.

General shortcomings Given the similarity of our different fusion approaches, there are many common shortcomings to all of them that we can address at once. First, no hyperparameters were tuned during the generation of results. We hoped that this would enable a more fair comparison of the fusion mechanisms. However , deep learning systems are very sensitive to hyperparameter selection, therefore there may be a setting we did not try which our proposed fusion approaches can benefit from. This can be fixed in future work by tuning the hyperparameters of each network. Secondly, the KITTI dataset may be small relative to the complex multimodal interactions we seek to learn from. It is possible that training on a larger dataset may lead to emergent behavior, particularly regarding the routing function which must model complex interactions. In future work, training on a larger dataset such as NuScenes would alleviate this issue. Thirdly, our approach does not explicitly address the greedy nature of gradient-based learning [30]. Gradient descent optimization may be unable to escape the local minima created by the easy-to-learn-from point cloud features. We even have tangible evidence that this issue is present in our systems (see fig.5). In future work, this shortcoming can be solved by implementing the iterative optimization schemes proposed in [30, 29]. Fourthly, the modified version of RetinaNet we use as our image backbone may not provide features with appropriately sized receptive fields. This could be addressed in future work by designing an image backbone architecture with features at more appropriate scales.

Reasons for the weak results of Routed Fusion Due to the complexity of the task, it is hard to tell what exactly we would need to change to see an improvement. However, it is possible that the routing function does not have enough incentive to explore using different scale image features and that it is stuck in a local minimum for this reason. Future work could address this issue by explicitly incorporating scale exploration into the objective by using a block coordinate descent approach that alternates optimization between the entire network and the router. Another potential issue is the features used to select image scales. It is possible that the local voxel information is insufficient to properly direct the routing function. Future work could try to incorporate global information by utilizing same 3D convolutions and only fusing features at the voxel level after some 3D convolutions have incorporated global information into the voxel features.

Reasons for the weak results of MOE Fusion The mixture of experts approach adds sparsely activated experts to the model. Though the MOE paradigm is effective in large-scale NLP settings, it is possible that in our smaller data setting, there is simply not enough signal to train each expert. Future work could try lowering the number of experts or increasing the supervisory signal to them via self-supervision tasks.

8 Conclusion

We explored four novel approaches to multimodal fusion in voxel-based 3D object detectors. While none of our approaches showed a significant performance gain over previous work, our empirical evaluation did shed light on the difficulties of multimodal learning and the potential benefits of integrating conditional computation into these architectures. Indeed, the strong performance of the Point Feature Baseline and the fast learning of the point cloud modality (seen in fig 5) emphasize the need for future approaches to explicitly address greedy gradient-based learning. Additionally, consistent improvements were shown for routed softmax-based approaches applied to cyclist detection, suggesting that conditional computation can effectively be leveraged for multimodal learning in some situations. In future work, we are most excited to explore alternating optimization approaches to routing and design image backbone architectures with features at appropriate scales.

References

[1] MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection, 2020.

[2] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6569–6578, 2019.

[3] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *CoRR*, abs/2101.03961, 2021.

[4] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[5] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[6] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[7] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[9] Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. Epnet: Enhancing point features with image semantics for 3d object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 35–52, Cham, 2020. Springer International Publishing.

[10] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Waslander. Joint 3d proposal generation and object detection from view aggregation, 2017.

[11] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[12] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[13] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[14] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[15] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 21–37, Cham, 2016. Springer International Publishing.

[17] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2016.

[18] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[19] Anshul Paigwar, David Sierra-Gonzalez, Özgür Erkent, and Christian Laugier. Frustum-pointpillars: A multi-stage approach for 3d object detection using rgb camera and lidar. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 2926–2933, October 2021.

[20] Su Pang, Daniel Morris, and Hayder Radha. Clocs: Camera-lidar object candidates fusion for 3d object detection. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10386–10393, 2020.

[21] Su Pang, Daniel Morris, and Hayder Radha. Fast-clocs: Fast camera-lidar object candidates fusion for 3d object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 187–196, January 2022.

[22] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas 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 (CVPR)*, July 2017.

[23] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[24] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

[25] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[26] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[27] Vishwanath A. Sindagi, Yin Zhou, and Oncel Tuzel. Mvx-net: Multimodal voxelnet for 3d object detection. *CoRR*, abs/1904.01649, 2019.

[28] Chunwei Wang, Chao Ma, Ming Zhu, and Xiaokang Yang. Pointaugmenting: Cross-modal augmentation for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11794–11803, June 2021.

[29] Weiyao Wang, Du Tran, and Matt Feiszli. What makes training multi-modal classification networks hard? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[30] Nan Wu, Stanisław Jastrzębski, Kyunghyun Cho, and Krzysztof J. Geras. Characterizing and overcoming the greedy nature of learning in multi-modal deep neural networks, 2022.

[31] Xiaopei Wu, Liang Peng, Honghui Yang, Liang Xie, Chenxi Huang, Chengqi Deng, Haifeng Liu, and Deng Cai. Sparse fuse dense: Towards high quality 3d detection with depth completion. *arXiv preprint arXiv:2203.09780*, 2022.

[32] Shaoqing Xu, Dingfu Zhou, Jin Fang, Junbo Yin, Zhou Bin, and Liangjun Zhang. Fusionpainting: Multimodal fusion with adaptive attention for 3d object detection. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 3047–3054, 2021.

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

[34] Maosheng Ye, Shuangjie Xu, and Tongyi Cao. Hvnet: Hybrid voxel network for lidar based 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[35] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11784–11793, June 2021.

[36] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Multimodal virtual point 3d detection. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 16494–16507. Curran Associates, Inc., 2021.

[37] Jin Hyeok Yoo, Yecheol Kim, Jisong Kim, and Jun Won Choi. 3d-cvf: Generating joint camera and lidar features using cross-view spatial feature fusion for 3d object detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 720–736, Cham, 2020. Springer International Publishing.

[38] Yihan Zeng, Chao Ma, Ming Zhu, Zhiming Fan, and Xiaokang Yang. Cross-modal 3d object detection and tracking for auto-driving. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3850–3857, 2021.

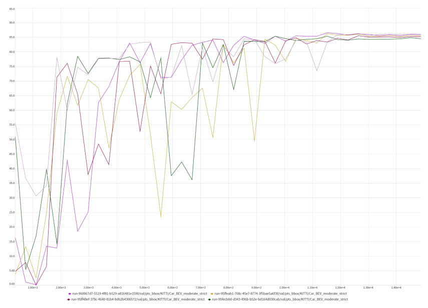
[39] Haolin Zhang, Dongfang Yang, Ekim Yurtsever, Keith A. Redmill, and Ümit Özgüner. Faraway-frustum: Dealing with lidar sparsity for 3d object detection using fusion. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 2646–2652, 2021.

[40] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

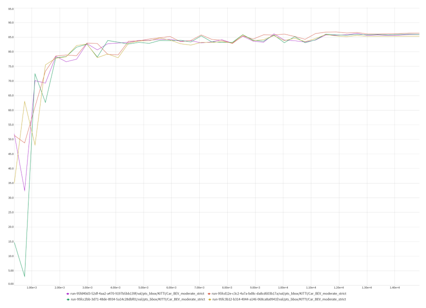
[41] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection, 2019.

9 Appendix

9.1 Extra Figures



(a) Image Feature Baseline



(b) Routed Point Fusion

Figure 5: **Car BEV Moderate average precision on the KITTI validation set plotted during training.** We see that the Image Feature Baseline (left) struggles to learn features which generalize quickly, while Routed Point Fusion learns very quickly.