Scaling Semantic Segmentation Beyond 1K Classes on a Single GPU

Shipra Jain[1,2], Danda Pani Paudel[1], Martin Danelljan[1], Luc Van Gool[1,3]
Computer Vision Lab, ETH Zurich, Switzerland[1]
KTH Royal Institute of Technology, Stockholm, Sweden[2]
KU Leuven, Belgium[3]
shipra@kth.se {paudel,martin.danelljan,vangool}@vision.ee.ethz.ch

Abstract

The state-of-the-art object detection and image classification methods can perform impressively on more than 9k classes. In contrast, the number of classes in semantic segmentation datasets is relatively limited. This is not surprising when the restrictions caused by the lack of labeled data and high computation demand for segmentation are considered. In this paper, we propose a novel training methodology to train and scale the existing semantic segmentation models for a large number of semantic classes without increasing the memory overhead. In our embedding-based scalable segmentation approach, we reduce the space complexity of the segmentation model's output from O(C) to O(1), propose an approximation method for ground-truth class probability, and use it to compute cross-entropy loss. The proposed approach is general and can be adopted by any state-of-the-art segmentation model to gracefully scale it for any number of semantic classes with only one GPU. Our approach achieves similar, and in some cases, even better mIoU for Cityscapes, Pascal VOC, ADE20k, COCO-Stuff10k datasets when adopted to DeeplabV3+ model with different backbones. We demonstrate a clear benefit of our approach on a dataset with 1284 classes, bootstrapped from LVIS and COCO annotations, with almost three times better mIoU than the DeeplabV3+. Our source code is available at: https://github.com/shipra25jain/ESSNet.

1 . Introduction

With the advent of deep learning, significant progress has been made in various image understanding tasks, including image classification, object detection, and image segmentation. The state-of-the-art methods can impressively classify images into 10k classes [15] and detect 9k different objects [49]. In contrast, segmentation models have been trained for a fairly limited number of common classes. The ability to segment a greater variety of objects, including small and rare object classes, is critical to many real-life applications like autonomous driving [2] and the scene exploration [7]. The scaling of existing segmentation models has several unresolved challenges. One of the challenges is the unbalanced distribution of classes. As mentioned in [21], due to the Zipfian distribution of classes in natural settings, there is a long tail of rare and small object classes that do not have a sufficient number of examples to train the model. The lack of segmentation datasets with a multitude of classes also limits us to develop scalable segmentation models. In fact, one can also argue from the other side. The reason for limited classes in existing segmentation datasets is the discouraging computational demand, alongside the labor-intensive annotations.

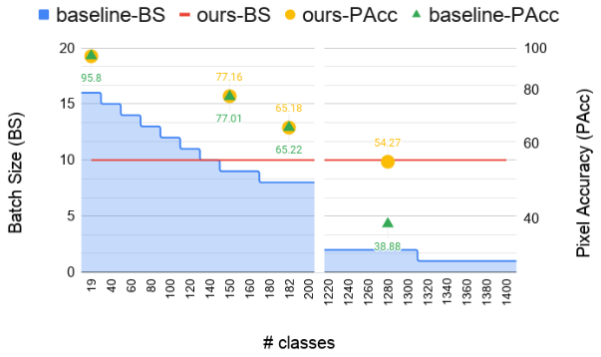

Figure 1. The left y-axis shows the maximum batch size that can fit in a single GPU for DeepLabV3+ model vs number of classes in the dataset. The right y-axis with markers in yellow and green color shows pixel accuracy for our model and baseline for following datasets (number of classes): Cityscapes (19), ADE20k (150), COCO-Stuff10k (182) and COCO+LVIS (1284).

The task of semantic segmentation is essentially a pixel-level classification of an image. Typically, it is performed by predicting an output tensor of $H \times W \times C$ for image size $H \times W$ and C number of semantic classes [36]. This is desirable during the pixel-wise classification by employing cross-entropy loss on the C-dimensional predictions. Unfortunately, the memory demand for such predictions

happens to be a major bottleneck for a large number of classes. Figure 1 also illustrates an example case: the maximum adjustable batch size of 512×512 versus the number of classes, in one standard GPU (Titan XP) while training the DeepLabV3+ model with ResNet50 backbone. As expected, the batch size sharply decreases, leading to only one image per batch for 1320 classes.

Most existing works [53, 63, 20, 8] primarily focus on the accuracy for datasets with a few hundred semantic classes using multiple GPUs. With the release of LVIS dataset [21], efforts are being made in scaling the instance segmentation models with a large number of classes. However, for a rich and complete understanding of the scene, semantic segmentation followed by panoptic segmentation [29] is the way to go forward. Therefore, it stands to reason that the semantic segmentation networks in the real-world will eventually have to get exposed to the classes at least as high as that of classification, *i.e*. 10K. Unfortunately, the benchmark results on ADE20k dataset with 150 classes require 4-8 GPUs during training [65]. Such demand for computational resources hinders researchers in emerging economies and small-scale industries from leveraging these models for research and developing further applications.

Naive approaches for training segmentation models on large number of classes and limited GPU memory may be designed by reducing the image resolution or batch size. Such solutions regrettably compromise the performance. As shown in [55], lower resolutions (or higher strides) result in blurry boundaries and coarse predictions and miss small but essential regions, such as poles and traffic signs. On the other hand, [66] has already demonstrated the need of larger batch size to achieve the state-of-the-art results. While techniques such as gradient accumulation [24] and group normalization [58] help to reduce the effect of low batch size, they fail to solve the problem completely when even a single batch size does not fit into the GPU memory. When more than one GPU is available, the authors in [63] offer a promising synchronized multi-GPU batch normalization technique to increase the effective batch size. Such solutions allow scaling of classes at the cost of scaling the GPUs. However, it is important to seek for the possibility of scaling the training on a high number of classes with a single GPU, which remains unexplored.

In this work, we propose a novel training methodology for which the memory requirement does not increase with the number of semantic classes. To the best of our knowledge, this is the first work to study efficient training methods for semantic segmentation models beyond 1K classes. Such scaling is achieved by reducing the output channels of existing networks and learning a low dimensional embedding of semantic classes. We also propose an efficient strategy to learn and exploit such embedding for the task of semantic image segmentation. Our main motive is to improve the scalability of the existing segmentation networks, instead of competing against, by endowing them the possibility of using only one GPU during training for a very high number of semantic classes. The major contributions of this paper are summarized as follows:

- We propose a novel scalable approach for training semantic segmentation networks for a large number of classes using only one GPU's memory.
- We experimentally demonstrate that the proposed method achieves 2.7x better mIoU scores on a dataset with 1284 classes, when compared against its counterpart, while retaining a competitive performance in the regime of a lower number of classes.
- For efficiency and generalization, we introduce an approximate method to cross-entropy measure and a semantic embedding space regularization term.
- Our method is theoretically grounded in terms of probabilistic interpretation and underlying assumptions.

2 . Related Works

Efficient training for segmentation. Existing methods are often concerned to perform segmentation in constrained devices by using limited floating point [46] to binary operations [67] for neural networks. Other kinds are either compact by design [39, 34] or compressed after training [47, 40, 25]. Strategies like pruning [37, 11] and distilling the knowledge [50, 44] from the large trained model have also been explored. Almost all these approaches are either compromised in accuracy, or discount the need for high training resources [5]. Many works focus on inference time on single GPU [56, 64, 61]. Recently, [10, 62] proposed memory-efficient approaches to preserve local-global information for high-resolution images. However, scalability issues regarding the number classes in semantic segmentation have attained little to no attention. Our method is complimentary in this regard.

Embeddings for segmentation related tasks. Our work is related to works that use embeddings for segmentation related tasks. Bottom-up approaches for instance segmentation use embeddings for one-stage training and improve performance for occluded and thin objects. A branch of work in the instance segmentation [43, 14, 3, 32, 19, 41, 42, 30] trains networks for dense prediction of pixel embeddings, which are later clustered into individual instances. These methods are based on metric learning, which learns embeddings such that pixels belonging to the same instance are close to each other, and vice versa. To predict the class of instances, [43, 42, 32, 19, 41] suggest to predict objectness for each object category and use cross entropy loss. [14, 3] compute the cluster centroids of each class over the

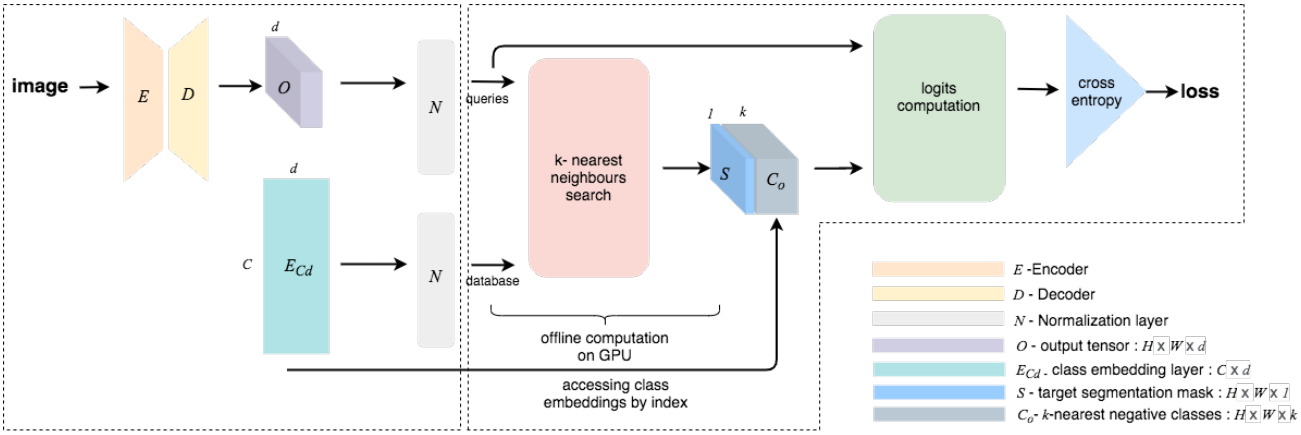

Figure 2. **Overview**: In left, an encoder-decoder-based segmentation network $[E, D]$ with d-channel output (pixel embeddings) and embedding network E_{Cd}, followed by normalization layers N. In right, k-nearest class embeddings from E_{Cd} are searched for every pixel embeddings in O. Logits for target classes in S and nearest classes in C_o are computed for cross-entropy loss.

entire training set. The classes are then inferred by comparing embeddings to the class-wise centroids. To efficiently find clustering seeds, [19, 42] predict the heatmap for every class. To make the network end-to-end trainable, [30] implements a variant of mean-shift clustering using a recurrent neural network. Extensions of these methods can be found in various applications [1, 31]. Differently, we exploit embeddings to capture the semantic information at the class level, unlike in the instance level of the mentioned methods. In context of semantic segmentation, [6] used embeddings for semi-supervised segmentation, [22] refines segmentation masks using similarities between pixel embeddings, [60] uses pixelwise embeddings for zero and few label segmentation and [22] learns embeddings for superpixels. [26] performs segmentation by extracting pixel-wise embeddings and clustering, and uses majority vote of its nearest neighbors from an annotated set to determine semantic class.

Contrastive loss for embedding learning. In recent years, a wide range of work [16, 12, 54, 59] have used metric learning and contrastive losses for representation learning. Our work builds upon the same idea, which can be seen in parallel to recently proposed contrastive cross-entropy loss in [28]. In essence, [28] is a generalization of popular triplet [57] and N-pair [52] losses. Contrastive losses are also very popular in self-supervised and semi-supervised settings [18, 48, 35, 9, 23]. Our loss fundamentally differs from the existing works, since our loss only operates on single-pixel and contrasts them against class embeddings.

3 . Embedding-based Scalable Segmentation

For state-of-the-art segmentation models, the output size is directly proportional to the number of semantic classes C. This poses a significant computational challenge while scaling them for datasets with a higher number of classes. In this work, we propose an embedding-based scalable segmentation method, which outputs a fixed number of channels and thus reduces the space complexity of output from $O(C)$ to $O(1)$. Along with the weights of the segmentation network, the model also learns d-dimensional class embeddings for C classes. We also propose the loss functions to learn and regularize the class embeddings such that the outputs (pixel embeddings) from segmentation network for same class pixels are clustered together and are closer to their respective class embedding. An overview of the proposed method is illustrated in Figure 2. In the following section, we first describe the method to integrate embeddings in existing networks, then provide their probabilistic formulation followed by loss function and algorithm for loss computation.

3 .1. Low Dimensional Embeddings

The key idea of our work is to reduce memory usage by representing the classes for each pixel by their corresponding embeddings. For every input image, we predict output (O in Figure 2) of size $H \times W \times d$ instead of the commonly used $H \times W \times C$, where $d << C$. To do this, we reduce the number of filters in the last convolution layer from C to d. In order to learn the dense target representation for every class, we add a small embedding matrix E_{Cd} consisting of C class embeddings with d dimensionality. The weights of this matrix are learned during training and fixed for inference. Unlike the existing models, where C dimensional output at every pixel represents the pixel's classwise likelihoods, the d-dimensional output in our approach represents the pixel in the semantic space of class embeddings. The embedding dimension can influence the performance of the model as with too few dimensions the model may underfit, and with too many dimensions the model may overfit.

An appropriate embedding dimension is the one to which adding further degrees of freedom would not give gains in performance. The reduction of dimension is followed by normalization along the depth of the output. The embedding layer is also followed by a normalization layer to ensure that embeddings lie on a unit radius hypersphere. Without normalization, a clear correlation between the length of class embeddings with the frequency of classes can be observed. Consistent with findings in [28], normalization of class and pixel embeddings helps the model suppress the bias introduced by class imbalance.

3 .2. Probabilistic Formulation

In our approach, the distribution of pixel embeddings O from the segmentation network is modeled using a gaussian mixture model. It comprises of C gaussians with $\mu_1, \mu_2, \mu_3 ...., \mu_C$ centroids, identical covariance matrix τI and equal mixing probability ρ, such that $C\rho = 1$. The probability of the output embedding x_i for pixel i can be given by Equation (1).

$$p(x_i) = \sum_{n=1}^{C} p(c_n)p(x_i|c_n) = \sum_{n=1}^{C} \rho\mathcal{N}(x_i|\mu_n, \tau I). \quad (1)$$

The prior probability of class c_n is $p(c_n)$. The posterior probability $p(c_n|x_n)$ gives the probability of data point x_i being sampled from the gaussian of class c_n. As a discriminative model, segmentation network maximizes the ground truth class posterior $p(c_{y_i}|x_i)$. To compute the class posteriors, bayes rule is used to derive Equation (2).

$$p(c_{y_i}|x_i) = \frac{p(x_i|c_{y_i}) * p(c_{y_i})}{p(x_i)} = \frac{\mathcal{N}(x_i|\mu_{y_i}, \tau I)}{\sum_{n=1}^{C} \mathcal{N}(x_i|\mu_n, \tau I)}, \quad (2)$$

$$\mathcal{N}(x|\mu, \tau I) = \frac{1}{\sqrt{2\pi\tau}} e^{-\frac{(x-\mu)^2}{2\tau}}. \quad (3)$$

However, Equation (2) requires computation of class-conditional probability for all classes. This makes it equally expensive in terms of computation as the C-channel output prediction. To overcome this problem, we propose to approximate $p(c_{y_i}|x_i)$ using Equation (4). For x_i, we search k nearest class centroids from $\mu_1, \mu_2, \mu_3 ...., \mu_C$ denoted by $\eta(x_i, k) = \{n_1, n_2, n_3 ...., n_k\}$, where $k \leq C$. Our approach is based on the assumption that $p(c_t|x_i) \approx 0$, if $t \notin \eta(x_i, k)$. The approximation error in the worst case is $\frac{1}{k} - \frac{1}{C}$, when all centroids are equidistant to x_i. If $k = C$ or the assumption is satisfied, then the approximation error is zero.

$$\overline{p}(c_{y_i}|x_i) = \frac{\mathcal{N}(x_i|\mu_{y_i}, \tau I)}{\sum_{n \in \eta(x_i,k) \cup y_i} \mathcal{N}(x_i|\mu_n, \tau I)}. \quad (4)$$

This probabilistic formulation motivates our loss functions described in the next section.

3 .3. Loss Functions

3 .3.1 Classification Loss

The cross-entropy loss function is almost the sole choice for classification tasks in practice. It is defined as negative log-likelihood of the target class, where the class likelihood is computed from the network outputs using the softmax function. On reducing the number of channels in output, the network does not provide the classwise logits directly. As shown in Equation (5), we use L2 distance between network outputs and class embeddings scaled by temperature τ to compute the classwise logits and probability $p_i^{y_i}$ for target class c_{y_i}.

$$p_i^{y_i} = \frac{e^{-\|x_i - \mu_{y_i}\|^2/\tau}}{\sum_{m=1}^{C} e^{-\|x_i - \mu_m\|^2/\tau}}. \quad (5)$$

The computation in the above equation's denominator demands a memory complexity of O($C \times D$), which does not align well with our goal. To solve this problem, we use the probabilistic formulation and assumption stated in Section 3 .2. We propose to mine k hard negative classes by searching k-nearest class embeddings for the pixel embedding x_i. In Equation (6), we approximate the target class probability $\overline{p}_i^{y_i}$ by using only k-nearest classes along with the target class for normalization and compute cross-entropy loss for classification.

$$L = \sum_{i=1}^{N} \log \overline{p}_i^{y_i}, \qquad \overline{p}_i^{y_i} = \frac{e^{-\|x_i - \mu_{y_i}\|^2/\tau}}{\sum_{m \in \eta(x_i,k) \cup y_i} e^{-\|x_i - \mu_m\|^2/\tau}}. \quad (6)$$

The idea is to use a value of k such that $O(k \times d)$ is significantly lower than $O(C)$ and can fit in the available memory. The search of the nearest neighbours is done in offline mode on GPU i.e. not included in the computational graph. The memory and speed efficient search algorithms, such as [27], can be used for this purpose.

As cross-entropy loss maximizes target class probability, minimizing it pulls the pixel embedding closer to its target class embedding, thus the pixel embeddings from the same class eventually get clustered together. Similar to previous works in [9, 28] , the appropriate value of temperature τ is critical for the best performance. It represents the allowed variance across the pixel embeddings belonging to the same class and thus the compactness of clusters.

3 .3.2 Regularization Loss

The classification loss models the interaction between pixel and class embeddings. To model the interaction among class embeddings and regularize them, we propose to use a max-margin loss. If class embeddings of two classes are very close, then the pixels belonging to those classes are prone to misclassification and can lead to poor generalization. The proposed loss applies repulsive force on the near-

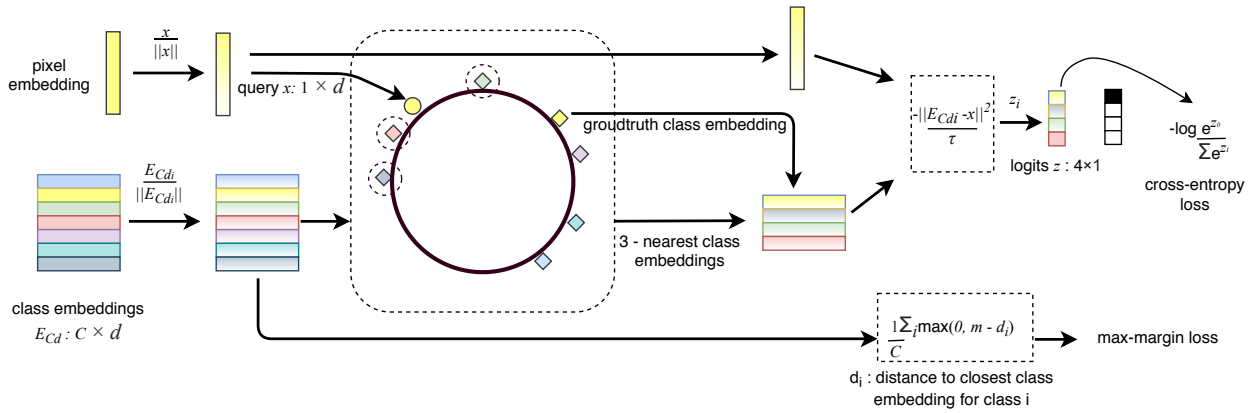

Figure 3. **Loss Computation**: A pixel embedding x and class embedding E_{Cd} are normalized to project on a hyperspherical manifold. For normalized x, k=3 nearest class embeddings are searched (shown by dotted circle). L2 distance between normalized x and class embeddings is used to compute logits for k negative nearest classes and a positive class. Further, classification and regularization loss is computed.

est class embedding for every class if it is closer than the margin distance m. Equation (7) gives the regularization loss where d_{ij} is the L2 distance between embeddings of class i and j.

$$L_r = \frac{1}{C}\sum_{i=1}^{C}\max(0, m - d_i), \qquad d_i = \min_{j\neq i, j\in C} d_{ij}\,. \quad (7)$$

Learning rate scheduler. During training, the weights for the segmentation network and the embedding network are computed and updated simultaneously. The segmentation network adjusts its weight to get pixelwise embeddings closer to corresponding class embeddings, while class embeddings move closer to respective pixel embeddings. We use higher momentum and decay the learning rate of embedding network more aggressively to stabilize the training.

3 .4. The Algorithm

We summarize the loss computation part of the proposed method in Algorithm 1. The loss computation for segmentation network $\mathsf{M_d}$ uses an image I with semantic label S. Note that our algorithm requires an efficient GPU-compatible nearest neighbour search function represented by *kNN()*, which takes a database and query vectors as inputs. Please refer to Figure 3 for visual illustration of the algorithmic steps. The computed loss is then used to train our network illustrated in Figure 2.

Algorithm 1 $\mathcal{L} =$ **LossCompute**$(I, S, \mathsf{M_d}, kNN())$

$O \leftarrow \mathsf{M_d}(I)$, $\quad O$ shape : $B \times H \times W \times d$
Turn off gradient computation
$C_k \leftarrow kNN(E_{Cd}, O)$, $C_o \leftarrow Reshape(C_k)$
Turn on gradient computation
$Z_o \leftarrow Concat(E_{Cd}(S), E_{Cd}(C_o))$
$Z \leftarrow -\|O - Z_o\|^2/\tau$
$P \leftarrow Softmax(Z)$, $P_{gt} \leftarrow P[0]$
$\mathcal{L} \leftarrow mean(\text{-log}\,(P_{gt})) + L_r$
Return $\mathcal{L}$

L_r is computed using Equation (7). Note that the output O and class embeddings E_{Cd} are normalized.

4 . Experiments

Implementation Details. We use DeepLabV3+ as the baseline and integrate our d-channel approach to DeepLabV3+ model. We use margin m of 0.2 in max-margin regularization loss and $\tau = 0.05$. The exact nearest neighbours are searched using the GPU mode of FAISS library. All experiments, unless mentioned, are performed using a single Titan X GPU, and the maximum possible batch size were used. For more implementation details, please refer to supplementary material.

Benchmark datasets. We conducted experiments on five datasets, whose details are given in Table 1. The used four datasets **Cityscapes** [13], **Pascal VOC** [17], **ADE20k** [65], and **COCO-Stuff10k** [4] are standard benchmarks. Due to lack of publicly available large scale dataset with high num-

Dataset	# classes	crop size	B	d	k
Cityscapes	19	400 × 800	14/10	7	6
Pascal VOC	21	512 × 512	14/10	7	8
ADE20k	150	512 × 512	8/10	12	7
COCO-Stuff10k	182	512 × 512	7/10	12	7
COCO+LVIS	1284	450 × 450	2/10	12	8

Table 1. **Dataset details.** Different dataset and their respective hyperparameters used to train models with ResNet50 backbone. The column **B** shows batch size for baseline and our method, **d** is embedding dimension and **k** in number of nearest neighbours.

ber of classes, we merged the COCO and LVIS dataset to demonstrate the capability of our method on 1284 classes.

COCO+LVIS - a merged dataset. We build a large-class segmentation dataset bootstrapped from stuff annotations of COCO [33] and instance annotation of LVIS [21] for COCO 2017 images [33]. LVIS is an instance segmentation dataset whose annotations are sparse for the whole image semantics. To overcome the sparsity, we merge the annotations of the stuff classes from COCO-Stuff dataset. After merging, the COCO+LVIS has the label sparsity of 19.5% (with 18.8% for validation). Note that, this sparsity is on par with benchmark datasets such as Pascal-MT (30.4%) [38] and Cityscapes (28.3%).

We use official split of LVIS, with about 100k train and 20k validation images. Only the semantic labels are used while ignoring the instance ids. LVIS has 1203 thing categories. Similarly, COCO has 91 stuff categories. Between these two datasets, 10 classes are common. This leads to the total of 1284 classes. Labels from LVIS is prioritize over those of COCO, whenever they overlap. Please, refer to our supplementary material for more details. As discussed earlier, datasets with large number of classes have long-tail distributions (causing the the problem of a severe class imbalance). This is also the case for COCO+LVIS. The mean IoU measure is known to be very sensitive to such class imbalance. Therefore, to capture a better picture, we also report frequency weighted IoU (FwIoU), along with the standard metrics: mean IoU (mIoU) and pixel accuracy (PAcc).

4 .1. Ablation Experiments

All ablation experiments are conducted for the Cityscapes with MobileNet [51] backbone, which are reported in Table 2-3 and Figure 4. Table 2 shows that irrespective of k, all models converge at mIoU 71.2 $\pm$ 0.2, while the higher number of nearest neighbours being faster in convergence. Hence, k can be chosen based on the trade-off between training time and the available GPU memory. These experiments do not use regularization loss. Figure 4 shows the increasing performance with the increase in embedding dimension from 4 to 7, followed by a slight drop. It also shows that the nearest neighbours offer better mIoU and convergence, compared to random sampling. Table 3 shows that max-margin loss provides

#NN	mIoU	Pixel Accuracy	iters
4	71.35	95.03	36.3k
6	71.05	95.07	32.1k
8	71.08	95.13	**29.1k**

Table 2. **Number of nearest neighbours vs. performance.** Mean IoU, pixel accuracy, and iterations for different number of nearest neighbours (#NN). Similar performance is achieved for different #NN with difference in convergence iterations.

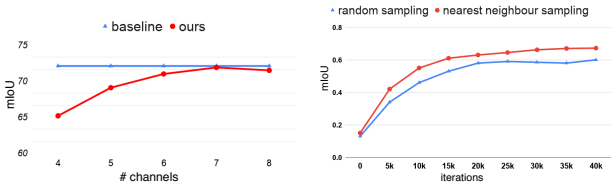

Figure 4. **Number of output channels and neighbours sampling.** Number of output channels d vs. mean IoU (left). The convergence of random sampling of k=7 vs. 7-nearest neighbours (right).

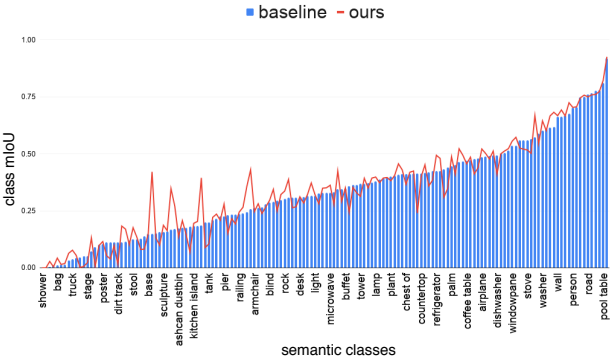

Figure 5. Classwise mIoU for ADE20k dataset with ResNet50 backbone. Our model does slightly better on some rare classes and performs comparable on dominant classes.

marginal improvement in mIoU and normalization of class embeddings contributes significantly towards better mIoU. Ablations on COCO+LVIS dataset are provided in supplementary material.

nn sampling	normalization	max-margin loss	mIoU	pixel accuracy
-	-	-	64.49	93.71
✓	-	-	67.20	94.43
✓	✓	-	72.56	95.14
✓	✓	✓	**73.03**	**95.40**

Table 3. Ablation study shows that our approach benefits from nearest neighbour (NN) sampling, normalization, and max-margin loss. Experiment in first row uses random sampling.

4 .2. Benchmark Results

Quantitative results. For datasets with a lower number of classes, Table 4 shows that the performance of our model with both ResNet50 and MobileNet backbones is comparable to that of the baseline. Figure 5 shows a comparison between classwise IoUs of ADE20k for both models. Classes in the plot are sorted based on IoUs for the baseline. We observe that our model performs better for some rare classes like shower, apparel, and stool over baseline, and these classes occur in some specific context like bathroom or bedroom. We hypothesize that our learned embeddings allows rare classes to implicitly borrow knowledge

dataset	Cityscapes						Pascal VOC				ADE20k				COCO-Stuff10k		COCO+LVIS		
backbone	MobileNet		ResNet50		ResNet101		MobileNet		ResNet50		MobileNet		ResNet50		ResNet50		ResNet50		
metric	mIoU	PAcc	mIoU	PAcc	mIoU	PAcc	mIoU	PAcc	mIoU	PAcc	mIoU	PAcc	mIoU	PAcc	mIoU	PAcc	mIoU	PAcc	fwIoU
baseline	72.11	95.22	75.25	**95.80**	**76.8**	96.22	71.07	92.25	**73.1**	**93.35**	34.02	75.07	**38.93**	77.01	32.56	**65.22**	1.68	38.88	22.66
ours	**73.03**	**95.40**	**75.64**	95.62	76.6	**96.28**	**71.15**	**92.28**	72.8	92.98	**34.11**	**75.19**	38.29	**77.16**	**32.60**	65.18	**4.57**	**54.27**	**39.67**

Table 4. Our model performs comparable to the baseline model for Cityscapes, PASCAL VOC, ADE20k and COCO-Stuff10k datasets. For COCO+LVIS dataset, it outperforms the baseline with large margin. The higher values of mean IoU (mIoU), pixel accuracy (PAcc) and Frequency weighted IoU (fwIoU) is better.

from the associated semantic context. As the frequency of classes increases, both models perform similarly.

For COCO+LVIS, our model clearly outperforms the baseline in terms of both mIoU and pixel accuracy. The low mIoU for both models, when compared to other datasets, can be explained by the long tail of thing classes in LVIS annotations. Figure 6 shows that as we increase number of rare classes, mIoU drops. Among 1284 classes, 220 classes occur in less than ten images in the training dataset. Please recall, the challenge of class imbalance is not within the scope of this work. For further analysis, we also report the frequency weighted IoU for COCO+LVIS. The superior performance of our method for COCO+LVIS can be explained by the five times higher batch size that we can fit in a single GPU. Lower batch size leads to noisy estimation of batch statistics in BatchNorm layer.

To reduce the effect of low batch size in baseline model, we perform experiments using gradient accumulation (GA) [24] and group normalization (GN) [58]. Table 5 shows that GA and GN help to improve the performance of both the models. GA increases the effective batch size of all the layers in network except BatchNorm as the mean and variance for every batch are computed during the forward pass. GN makes the computation of mean and variance independent of batch size. However, these techniques are not the substitute for our approach as our major contribution lies on restricting the number of output channels, thus decreasing the memory complexity from $O(C)$ to $O(1)$. Using GN/GA (with baseline model) alone would not be possible for very high number of classes or larger images as even a single image would not fit into the memory (because of $O(C)$ complexity). To understand the performance loss incurred by limited computational resources for COCO+LVIS, we conduct experiments on 4 GPUs (16 GB each) with synchronized batch norm (no GN). We use batch size of 12 and 40 for baseline and our model, respectively. We did not perform hyperparameter search for this experiment and used embedding size of 16. We believe that mIoU can be further improved by reducing the batch size and by increasing the embedding dimension.

Analysis of memory consumption. In Table 6, we investigate the peak memory usage in GPU during training. We observe that for datasets with low number of classes, like Cityscapes, baseline uses less memory to accommodate the bigger batch size. However, our approach is better suited for datasets with a higher number of classes like ADE20k. In this case, our approach accommodates a bigger batch size for the same memory. Despite of any increase in number of

model	mIoU	FwIoU	PAcc
baseline	1.68	22.66	38.88
ours	**4.57**	**39.67**	**54.27**
baseline + GA	2.76	29.57	46.34
our + GA	**5.01**	**41.87**	**57.05**
baseline + GN	5.15	37.89	53.45
ours + GN	**6.26**	**43.03**	**59.01**
baseline + 4 GPUs	7.86	42.2	58.1
ours + 4 GPUs	**8.78**	**43.8**	**59.3**

Table 5. Results on COCO+LVIS dataset with Gradient Accumulation (GA), GroupNorm (GN) and 4 GPUs. Gradients are accumulated over 5 and 2 steps for baseline and ours, respectively. In GN experiments, we use groups of 16-channels for both methods.

dataset	model	train BS	memory (in GB)
Cityscapes	baseline	**14**	12.1
	ours	12	10.4
ADE20k	baseline	8	10.3
	ours	**10**	10.0
COCO+LVIS	baseline	2	9.94
	ours	**10**	10.4

Table 6. Analysis of peak GPU memory usage and maximum batchsize for 1 GPU. For Cityscapes dataset, baseline has better memory consumption while our model is memory efficient for ADE20k and COCO+LVIS datasets.

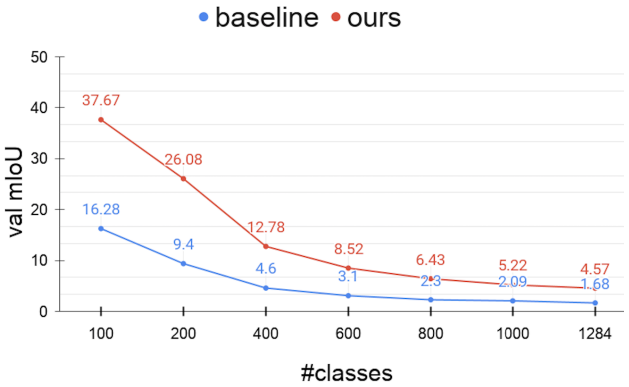

Figure 6. mIoU on COCO+LVIS with increasing number of classes, with most frequent first, for baseline and our method.

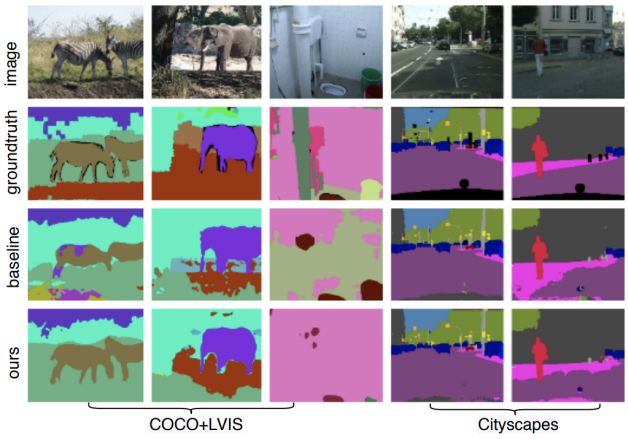

Figure 7. Qualitative results of our method and the baseline. Black color denotes the unlabelled pixels. For COCO+LVIS dataset, both models miss rare classes such as bucket and pipe. Our model performs better than baseline for dominant classes like wall. For Cityscapes, both model provide similar results.

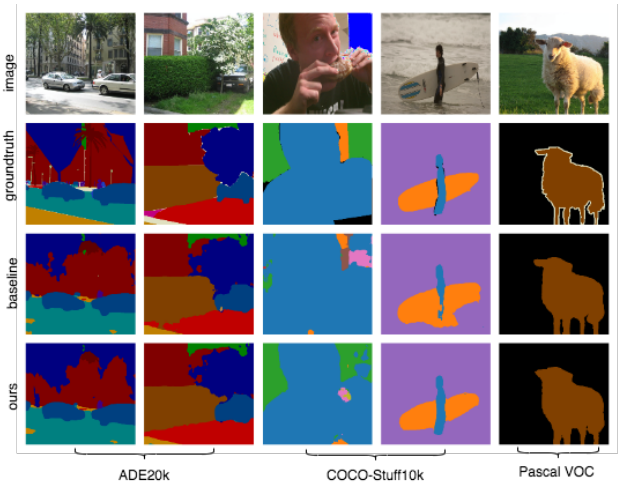

Figure 8. Qualitative results for our method and the baseline. For ADE20k, COCO-Stuff10k, and Pascal VOC datasets both the models provide similar qualitative results.

classes, our model's memory requirement remains almost the same, thanks to the $O(1)$ complexity of the proposed method. This allow us to scale to 1k+ classes and still use the batch size of 10. On the other hand, the baseline model can only fit a batch size of two in a single GPU. For details on inference time, please refer to supplementary material.

Qualitative results. In Figure 7 and 8, we show qualitative results. In COCO+LVIS dataset, rare and small area classes are mostly missed by both the models which reflects in lower mIoU scores. Our model segments the dominant classes like wall and grass much better than the baseline. For CityScapes, Pascal VOC, ADE20k, and COCO-Stuff10k, segmentation masks from both models look very similar. We also notice that almost same set of pixels are misclassified by both the models in many examples.

Figure 9. Synthesized images for Cityscapes. Left to right: real image; generated using: one-hot encoding (FID = **60.47**); random embeddings (FID = **64.14**); our class embeddings (FID = **58.34**).

Semantic class embeddings for image synthesis. Using the learned class embeddings, our method performs well for the task of semantic segmentation. This suggests that our embeddings capture the semantics of the classes and represent them efficiently in lower-dimensional space. In order to demonstrate the utility, beyond segmentation, of our learned embeddings, we conducted experiments with SPADE network [45] to synthesize photo-realistic images. SPADE takes class semantics in the form of a one-hot vector corresponding to the class label for every pixel as input. We conduct three experiments : 1) one-hot vector semantics (19 classes) as input with $B = 3$, 2) randomly initialized 7-dim embeddings as input with $B = 4$, and 3) 7-dim class embeddings from our trained segmentation network with $B = 4$. Figure 9 shows image examples generated for the Cityscapes test dataset using a single GPU. Our embeddings achieve a lower FID score than random embeddings, which suggests that our learned class embeddings can also be used for synthesis. Embedding-based semantic inputs for the memory-efficient generation of images, with a higher number of classes, remains a promising direction for future work. For visualization of our class embeddings, please refer to the supplementary materials.

5 . Conclusions

In this work, we address the problem of memory complexity of existing segmentation approaches with large number of semantic classes. By leveraging our understanding of metric learning and probabilistic mixture models, we proposed a novel approach to train the segmentation models. The proposed method can be used for any number of classes to train the segmentation model in a single GPU's memory. Experiments demonstrate that our method can retain the performance, while improving the scalability; thus allowing us to segment a large number of classes

Acknowledgments: This work was supported by the ETH Future Computing Laboratory (EFCL) financed by a gift from Huawei Technologies, Arbrea Labs AG through provided computational resources, and an Nvidia GPU grant.

References

[1] Ali Athar, S. Mahadevan, Aljosa Osep, L. Leal-Taixé, and B. Leibe. Stem-seg: Spatio-temporal embeddings for instance segmentation in videos. In *ECCV*, 2020. 3

[2] Sara Beery, Yang Liu, Dan Morris, Jim Piavis, Ashish Kapoor, Neel Joshi, Markus Meister, and Pietro Perona. Synthetic examples improve generalization for rare classes. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 863–873, 2020. 1

[3] Bert De Brabandere, Davy Neven, and Luc Van Gool. Semantic instance segmentation with a discriminative loss function. *CoRR*, abs/1708.02551, 2017. 2

[4] H. Caesar, J. Uijlings, and V. Ferrari. Coco-stuff: Thing and stuff classes in context. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1209–1218, 2018. 5

[5] Zachariah Carmichael, Hamed F Langroudi, Char Khazanov, Jeffrey Lillie, John L Gustafson, and Dhireesha Kudithipudi. Performance-efficiency trade-off of low-precision numerical formats in deep neural networks. In *Proceedings of the Conference for Next Generation Arithmetic 2019*, pages 1–9, 2019. 2

[6] Krishna Chaitanya, Ertunc Erdil, Neerav Karani, and Ender Konukoglu. Contrastive learning of global and local features for medical image segmentation with limited annotations. *Advances in Neural Information Processing Systems*, 33, 2020. 3

[7] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33, 2020. 1

[8] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *CoRR*, abs/1802.02611, 2018. 2

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020. 3, 4

[10] Wuyang Chen, Ziyu Jiang, Zhangyang Wang, Kexin Cui, and Xiaoning Qian. Collaborative global-local networks for memory-efficient segmentation of ultra-high resolution images. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 8924–8933. Computer Vision Foundation / IEEE, 2019. 2

[11] Xinghao Chen, Yunhe Wang, Yiman Zhang, Peng Du, Chunjing Xu, and Chang Xu. Multi-task pruning for semantic segmentation networks. *CoRR*, abs/2007.08386, 2020. 2

[12] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*, pages 539–546. IEEE Computer Society, 2005. 3

[13] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 3213–3223. IEEE Computer Society, 2016. 5

[14] B. De Brabandere, D. Neven, and L. Van Gool. Semantic instance segmentation for autonomous driving. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 478–480, 2017. 2

[15] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 1

[16] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1422–1430. IEEE Computer Society, 2015. 3

[17] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010. 5

[18] William Falcon and Kyunghyun Cho. A framework for contrastive self-supervised learning and designing A new approach. *CoRR*, abs/2009.00104, 2020. 3

[19] Alireza Fathi, Zbigniew Wojna, Vivek Rathod, Peng Wang, Hyun Oh Song, Sergio Guadarrama, and Kevin P. Murphy. Semantic instance segmentation via deep metric learning. *CoRR*, abs/1703.10277, 2017. 2, 3

[20] J. Fu, J. Liu, J. Jiang, Y. Li, Y. Bao, and H. Lu. Scene segmentation with dual relation-aware attention network. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–14, 2020. 2

[21] Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1, 2, 6

[22] Adam W. Harley, Konstantinos G. Derpanis, and Iasonas Kokkinos. Learning dense convolutional embeddings for semantic segmentation. *CoRR*, abs/1511.04377, 2015. 3

[23] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9726–9735. IEEE, 2020. 3

[24] Joeri R. Hermans, Gerasimos Spanakis, and R. Moeckel. Accumulated gradient normalization. *ArXiv*, abs/1710.02368, 2017. 2, 7

[25] Andrew Holliday, Mohammadamin Barekatain, Johannes Laurmaa, Chetak Kandaswamy, and Helmut Prendinger. Speedup of deep learning ensembles for semantic segmentation using a model compression technique. *Comput. Vis. Image Underst.*, 164:16–26, 2017. 2

[26] Jyh-Jing Hwang, S. Yu, Jianbo Shi, Maxwell D. Collins, Tien-Ju Yang, X. Zhang, and Liang-Chieh Chen. Segsort: Segmentation by discriminative sorting of segments. *2019*

IEEE/CVF International Conference on Computer Vision (ICCV), pages 7333–7343, 2019. 3

[27] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *CoRR*, abs/1702.08734, 2017. 4

[28] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673. Curran Associates, Inc., 2020. 3, 4

[29] Alexander Kirillov, Kaiming He, Ross B. Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 9404–9413. Computer Vision Foundation / IEEE, 2019. 2

[30] S. Kong and Charless C. Fowlkes. Recurrent pixel embedding for instance grouping. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9018–9028, 2018. 2, 3

[31] Siyang Li, Bryan Seybold, Alexey Vorobyov, Alireza Fathi, Qin Huang, and C.-C. Jay Kuo. Instance embedding transfer to unsupervised video object segmentation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6526–6535. IEEE Computer Society, 2018. 3

[32] Xiaodan Liang, Liang Lin, Yunchao Wei, Xiaohui Shen, Jianchao Yang, and Shuicheng Yan. Proposal-free network for instance-level object segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(12):2978–2991, 2018. 2

[33] 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 David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014. 6

[34] Zhong Qiu Lin, Brendan Chwyl, and Alexander Wong. Edgesegnet: A compact network for semantic segmentation. *CoRR*, abs/1905.04222, 2019. 2

[35] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *CoRR*, abs/2006.08218, 2020. 3

[36] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3431–3440. IEEE Computer Society, 2015. 1

[37] Jian-Hao Luo, Hao Zhang, Hong-Yu Zhou, Chen-Wei Xie, Jianxin Wu, and Weiyao Lin. Thinet: Pruning CNN filters for a thinner net. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(10):2525–2538, 2019. 2

[38] K. Maninis, Ilija Radosavovic, and I. Kokkinos. Attentive single-tasking of multiple tasks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1851–1860, 2019. 6

[39] Vladimir Nekrasov, Hao Chen, Chunhua Shen, and Ian Reid. Fast neural architecture search of compact semantic segmentation models via auxiliary cells. In *Proceedings of the IEEE Conference on computer vision and pattern recognition*, pages 9126–9135, 2019. 2

[40] Vladimir Nekrasov, Chunhua Shen, and Ian Reid. Light-weight refinenet for real-time semantic segmentation. In *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018*, page 125. BMVA Press, 2018. 2

[41] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Fast scene understanding for autonomous driving. *CoRR*, abs/1708.02550, 2017. 2

[42] Davy Neven, Bert De Brabandere, Marc Proesmans, and Luc Van Gool. Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 8837–8845. Computer Vision Foundation / IEEE, 2019. 2, 3

[43] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 2277–2287, 2017. 2

[44] Sangyong Park and Yong Seok Heo. Knowledge distillation for semantic segmentation using channel and spatial correlations and adaptive cross entropy. *Sensors*, 20(16):4616, 2020. 2

[45] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2337–2346. Computer Vision Foundation / IEEE, 2019. 8

[46] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and E. Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *ArXiv*, abs/1606.02147, 2016. 2

[47] Rudra P. K. Poudel, Ujwal Bonde, Stephan Liwicki, and Christopher Zach. Contextnet: Exploring context and detail for semantic segmentation in real-time. In *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018*, page 146. BMVA Press, 2018. 2

[48] Senthil Purushwalkam and Abhinav Gupta. Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases. *CoRR*, abs/2007.13916, 2020. 3

[49] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 6517–6525. IEEE Computer Society, 2017. 1

[50] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 2

[51] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 6

[52] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 1849–1857, 2016. 3

[53] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation. *CoRR*, abs/2005.10821, 2020. 2

[54] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *CoRR*, abs/1906.05849, 2019. 3

[55] Li Wang, Dong Li, Yousong Zhu, Lu Tian, and Yi Shan. Dual super-resolution learning for semantic segmentation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 3773–3782. IEEE, 2020. 2

[56] Yu Wang, Quan Zhou, Jia Liu, Jian Xiong, Guangwei Gao, Xiaofu Wu, and Longin Jan Latecki. Lednet: A lightweight encoder-decoder network for real-time semantic segmentation. In *2019 IEEE International Conference on Image Processing, ICIP 2019, Taipei, Taiwan, September 22-25, 2019*, pages 1860–1864. IEEE, 2019. 2

[57] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244, 2009. 3

[58] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, 2018. 2, 7

[59] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 3733–3742. IEEE Computer Society, 2018. 3

[60] Yongqin Xian, Subhabrata Choudhury, Yang He, Bernt Schiele, and Zeynep Akata. Semantic projection network for zero- and few-label semantic segmentation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8248–8257, 2019. 3

[61] Changqian Yu, Changxin Gao, Jingbo Wang, Gang Yu, Chunhua Shen, and Nong Sang. Bisenet V2: bilateral network with guided aggregation for real-time semantic segmentation. *CoRR*, abs/2004.02147, 2020. 2

[62] Gang Zhang, Tao Lei, Yi Cui, and Ping Jiang. A dual-path and lightweight convolutional neural network for high-resolution aerial image segmentation. *ISPRS Int. J. Geo Inf.*, 8(12):582, 2019. 2

[63] Hang Zhang, Kristin J. Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7151–7160. IEEE Computer Society, 2018. 2

[64] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11207 of *Lecture Notes in Computer Science*, pages 418–434. Springer, 2018. 2

[65] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5122–5130. IEEE Computer Society, 2017. 2, 5

[66] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ADE20K dataset. *Int. J. Comput. Vis.*, 127(3):302–321, 2019. 2

[67] Bohan Zhuang, Chunhua Shen, Mingkui Tan, Lingqiao Liu, and Ian Reid. Structured binary neural networks for accurate image classification and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 413–422, 2019. 2