Superpixel Positional Encoding to Improve ViT-based Semantic Segmentation Models

Roberto Amoroso[1]
roberto.amoroso@unimore.it
Matteo Tomei[2]
matteo.tomei@prometeia.com
Lorenzo Baraldi[1]
lorenzo.baraldi@unimore.it
Rita Cucchiara[1]
rita.cucchiara@unimore.it

[1] University of Modena and Reggio Emilia
Modena, Italy
[2] Prometeia
Bologna, Italy

Abstract

In this paper, we present a novel superpixel-based positional encoding technique that combines Vision Transformer (ViT) features with superpixels priors to improve the performance of semantic segmentation architectures. Recently proposed ViT-based segmentation approaches employ a Transformer backbone and exploit self-attentive features as an input to a convolutional decoder, achieving state-of-the-art performance in dense prediction tasks. Our proposed technique is plug-and-play, model-agnostic, and operates by computing superpixels over the input image. It determines a positional encoding based on the centroids and shapes of the superpixels, and then unifies this semantic-aware information with the self-attentive features extracted by the ViT-based backbone. Our results demonstrate that this simple positional encoding strategy, when applied to the decoder of ViT-based architectures, leads to a significant improvement in performance without increasing the number of parameters and with negligible impact on the training time. We evaluate our approach on different backbones and architectures and observe a significant improvement in terms of mIoU on the ADE20K and Cityscapes datasets. Notably, our approach provides improved performance on classes with low occurrence in the dataset while mitigating overfitting on classes with higher representation, ensuring a good balance between generalization and specificity.

1 Introduction

In this work, we investigate the effectiveness of classical perception-based Computer Vision approaches when combined with more recent, self-attentive, approaches for dense tasks. The rationale behind this idea is that low-level assumptions and knowledge-based models, such as a perceptual grouping based on appearance similarity, can potentially be useful for improving the accuracy of semantic segmentation in generic images.

In recent years, the Transformer architecture [41] has received a relevant interest from the natural language processing (NLP) community [14, 36] and has established a new state-of-

the-art in fundamental vision tasks such as classification [15, 40], detection [7], and segmentation [58]. In particular, recently proposed architectures such as DPT [37], SegFormer [48], and SETR [58], have successfully integrated attentive architectures for dense prediction.

Transformer-based architectures share the common characteristic of employing a positional encoding strategy to encode the relative or absolute position of words or tokens in a sequence. This is particularly important in the context of NLP, where the order of words in a sentence profoundly impacts its meaning. Similarly, in Vision Transformers, positional encoding is used to encode the relative position of patches in an image, facilitating the understanding of spatial relationships among these objects and enabling more accurate predictions.

Although there are several works investigating positional encoding strategies for classification [15, 24, 34, 37, 47, 48], the literature on positional encoding techniques for dense prediction tasks, such as segmentation, is scarce. Moreover, existing positional encoding strategies share a common limitation, as they solely provide information about the position of input patches, while lacking semantic priors that offer insights into the shape and edges between different objects, *i.e.*, semantic classes, within the input image.

To address this limitation, we discuss if and when a grouping strategy based on appearance similarity, such as the one of superpixels [38], can be useful for semantic segmentation when employing self-attentive-based architectures. Superpixels cluster adjacent and perceptually similar pixels in uniform image regions by following edges and color variations. As such, they have often been adopted in segmentation before the appearance of CNN-based methods. Since the seminal work by Ren and Malik [38], tens of approaches have been proposed to create superpixels according to different optimization functions [1, 29], and also Convolutional Networks have been adopted for creating superpixels [50]. Regardless of the generation approach, superpixels can be considered seed areas for segmentation tasks, as they provide a precise indication of where edges lie and have a perceptual meaning. In Transformer-based architectures, therefore, superpixels may be useful to recover precise boundaries between classes.

Building upon these insights, we devise a novel superpixel-based positional encoding (PE) strategy specifically designed for semantic segmentation. Our strategy injects superpixel shape and position priors into the ViT encoder features, creating more boundary-aware semantic latent space representations. In our experimental evaluation, we also investigate an ablation of several positional encoding strategies to encode and embed superpixel information in the encoder features.

Contributions. To sum up, our contributions are as follows:

- We investigate the integration of superpixels in ViT-based architectures for semantic segmentation.
- We propose a comprehensive comparison of several positional encoding strategies based on superpixels centroids and shape.
- We conduct experiments on the ADE20K and the Cityscapes datasets, employing different backbones and architectures. Results outline that the proposed positional embedding provides a significant improvement in terms of mIoU. Surprisingly, our approach is particularly effective in enhancing the segmentation performance for classes with low occurrence in the dataset, while also mitigating the risk of overfitting on classes with higher representation.

2 Related Work

Superpixels. Observing that pixels are not natural and meaningful entities to represent images, the first reference to superpixels as a preprocessing step can be found in [38]. Several alternatives for superpixel extraction followed, which can be classified based on their high-level approach, according to the categorization reported in [1] and [39]. Besides watershed-based algorithms [4, 32, 33], methods based on clustering techniques [1, 27, 42, 46], such as k-means, use pixel color, and spatial information and allow to specify the desired number of superpixels and their compactness. Other strategies instead treat images like a graph and partition their edges based on color similarities [17, 20, 29, 38]. Deep learning-based approaches have recently been proposed for superpixel computation [25, 50], and superpixels have been exploited for preserving edges and improving semantic segmentation [18, 49]. More recently, [53] applies a superpixel algorithm to the input image to extract semantically homogeneous regions and segment the image by per-region prediction using a sequence-to-sequence Transformer. For a comprehensive description and evaluation of state-of-the-art superpixel algorithms, we refer the reader to [39].

Semantic segmentation. Semantic segmentation is the dense prediction task of assigning a semantic label to each pixel in an image and can be seen as an extension of image classification at the pixel level. It is a fundamental problem in computer vision and has numerous applications, such as object detection, scene understanding, and image editing. Recently, several deep learning-based approaches have been proposed for semantic image segmentation. One of the most successful approaches is the fully convolutional network (FCN) [31], which enabled outstanding progress in semantic segmentation, performing pixel-to-pixel classification in an end-to-end manner. Many recent efforts have been focused on improving different aspects of FCN. Several works [9, 10, 11] enlarged the receptive field by adopting dilated convolutions, while others introduced context modeling [21, 28, 35, 52, 54], boundary information refinement [2, 3, 5, 8, 51] and multi-scale feature aggregation [43, 56] to obtain fine-grained predictions. Recently, attention-based models [19, 23, 26, 45, 57] have been employed to learn long-range context information. While these approaches adopt convolutional encoding backbones [22] for feature extraction, more recent methods have proved the effectiveness of employing Transformer-based [41] backbones for semantic segmentation.

Vision Transformers for dense prediction. Transformer-based architectures [41] have gained increasing attention from the computer vision community [15, 40], matching or even improving CNNs performance. Dense prediction tasks, specifically semantic segmentation, have also been tackled by adding convolutional or MLP decoders on top of ViT-based architectures. One key advantage of ViTs is their ability to process high-resolution images without downscaling, allowing the retention of more detailed information about the objects in an image and leading to improved segmentation accuracy. Ranftl *et al*. [37] propose to reshape tokens from different layers of a ViT-like backbone into an image representation with decreasing resolution and increasing channels. Other approaches focus on reducing memory and model parameters, by downsampling the encoder sequence length in subsequent layers [44] and proposing a lightweight MLP decoder [48], by computing self-attention only within local groups of patches [12, 30], or by transposing the query-key interaction to enable a linear complexity with respect to the number of patches [16]. More recently, Zhang *et al*. [53] propose a plain transformer encoder-decoder architecture that employs a novel attention-to-mask mechanism to generate segmentation masks based on the similarity between learnable class tokens and multi-level ViT feature maps.

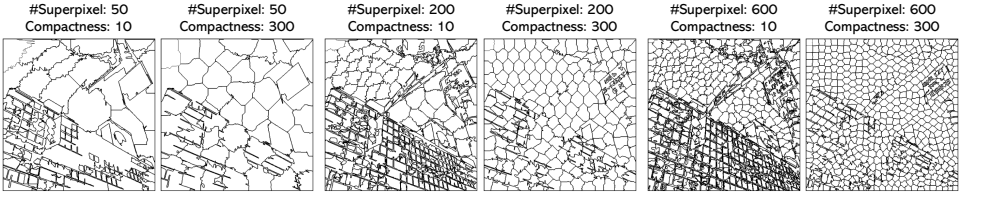

Figure 1: Example of superpixel maps with varying resolutions and compactness.

3 Method

For dense prediction tasks, Vision Transformers are commonly designed following an encoder-decoder architecture [37, 48, 58], where features coming from different encoder layers are used to predict the segmentation map. In this work, we propose a superpixel-based positional encoding that injects shape and position priors into the ViT encoder features, creating more boundary-aware semantic latent space representations. The proposed approaches are *model-agnostic*, *parameters-free* and *plug-and-play*; they do not involve any architectural changes to the encoder or decoder and only require the extraction of a superpixel map over the input image through an algorithm that we call $\mathcal{S}$.

More in detail, given the superpixel algorithm $\mathcal{S}$ and an input image z with shape $H \times W \times C$, we apply $\mathcal{S}$ to z to obtain a superpixel map $\mathcal{L}$ with shape $H \times W$, and a set of centroids $\mathcal{C}$ with shape $N_s \times 2$, where N_s represents the number of superpixels extracted by $\mathcal{S}$. $\mathcal{L}$ contains predicted superpixels, *i.e.*, each pixel in $\mathcal{L}$ is represented by an integer in the range $[0, N_s - 1]$ specifying which of the N_s superpixels it belongs to, while $\mathcal{C}$ contains the (x, y) coordinates of the centroid of each superpixel, relative to the input image.

The generation of superpixels is influenced by an important hyperparameter, namely compactness. Varying the compactness magnitude affects shape, size, and boundary smoothness. Increasing compactness results in smoother superpixels, while decreasing compactness leads to irregular shapes, similar to overfitting. Higher compactness captures spatial information better and aids information extraction. However, excessive values lead to grid-like superpixel maps. Examples are reported in Figure 1.

Superpixel-based Positional Encoding. The literature on ViT-based architectures [15, 47] highlights the importance of incorporating a positional encoding into the input embedding to achieve superior model performance. Traditionally, positional encoding conveys information about the *absolute* [15, 37] or *relative* [15, 48] position of the input patches. Recent approaches also incorporate positional encoding into intermediate layers of the architecture [24, 34], especially when the features extracted by the ViT-based encoder are downscaled in some way, such as through pooling [6]. This underscores the significance of injecting position information not only at the input but also in the intermediate processing of the model.

Our superpixel-based positional encoding (PE) not only provides information about the position of input patches but also offers insights into the shape and edges between different semantic classes within the input image. The proposed method is built upon the map of superpixels $\mathcal{L}$ and their centroids $\mathcal{C}$, as outlined in the following:

❶ For a given input image z, we extract a map of superpixels $\mathcal{L}$ along with their centroids $\mathcal{C}$ (colored dots in Figure 2–top-left). ❷ For each superpixel $\mathcal{L}_i$, with $i \in [0, N_s - 1]$, we compute the encoding of its position PE_i (colored rectangles in Figure 2–top-center) with the same shape d_{model} as the latent vector size of the Vision Transformer. By sharing the same latent shape, our Superpixel-based PE can be added to the features extracted from the ViT encoder. The superpixel position encoding PE_i can be either *absolute*, when employing the

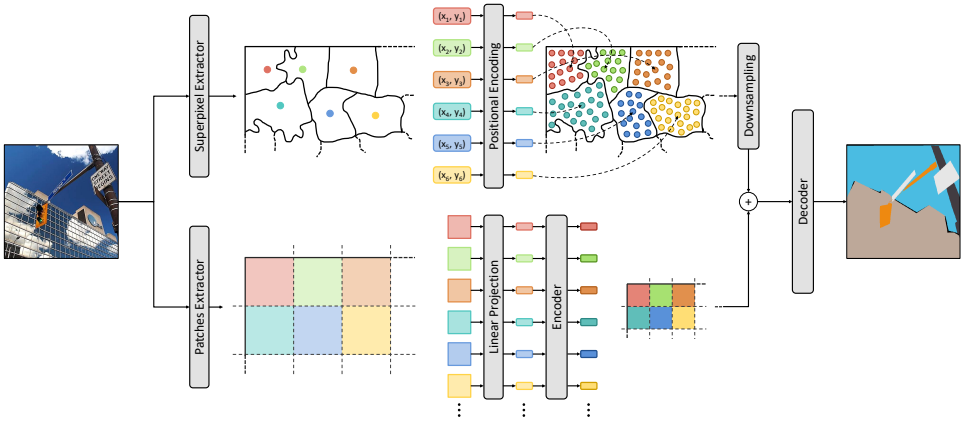

Figure 2: Our superpixel-based positional encoding strategy. We extract a positional encoding that can be *absolute*, when exploiting a sinusoidal encoding of the superpixels centroids, or *relative*, when leveraging the progressive index associated with each superpixel.

coordinates (x,y) of the corresponding centroid $\mathcal{C}_i$, or *relative*, when using the progressive index i associated with the superpixel $\mathcal{L}_i$, as discussed below. ❸ Given the i-th superpixel $\mathcal{L}_i$, we replicate its positional encoding PE_i over every pixel belonging to $\mathcal{L}_i$. In Figure 2–top-right, the colored small circles represent the positional encoding PE replicated over the surface of the corresponding superpixel. Following this approach, we obtain a superpixel-based positional encoding map $PE_{\mathcal{L}}$ with shape $H \times W \times d_{model}$. ❹ At the same time, we also extract N_p squared regular patches (dotted grid in Figure 2–bottom-left) that we feed as input to the ViT encoder. The f^{th} layer of the encoder outputs a feature vector f^j (colored rectangles in Figure 2–bottom-center) with shape $N_p \times d_{model}$, after removing the *cls_token* if adopted. ❺ Finally, we downsample $PE_{\mathcal{L}}$ and sum to the encoder features map f^j, before feeding the result to the downstream segmentation decoder (Figure 2–bottom). These operations are executed both at training and inference time.

In other words, the pixels in $\mathcal{L}$ that belong to a specific superpixel are replaced with the positional encoding PE of that superpixel. This results in a full-resolution map that encodes both the shape and position of the semantically homogeneous regions (*i.e.*, superpixels) computed by $\mathcal{S}$. Since the segmentation decoder usually expects features with spatial shape $\sqrt{N_p} \times \sqrt{N_p} \times d_{model}$ as input, obtained by reshaping f^j, we simply downsample the spatial resolution of $PE_{\mathcal{L}}$ in order to match $\sqrt{N_p} \times \sqrt{N_p}$. This downsampling operation does not affect the other components of the decoder, ensuring that they remain unmodified.

Absolute and relative position encoding strategies. Although superpixel maps provide priors on shape and edges, the position encoding strategy remains to be defined. As a solution, we propose two approaches to inject superpixel position information: an *absolute* positional encoding that exploits the information of the centroids $\mathcal{C}$ and a *relative* positional encoding that leverages the progressive index associated with each superpixel of $\mathcal{L}$.

Inspired by [41], our *absolute* superpixel positional encoding adopts sine and cosine functions of different frequencies to encode x and y coordinates of the superpixel centroids:

$$SinPE^{c}_{(sup,2i)} = \sin(c_{(sup)}/10000^{2i/d_{pe}}), \quad SinPE^{c}_{(sup,2i+1)} = \cos(c_{(sup)}/10000^{2i/d_{pe}}) \qquad (1)$$

where c represents the x or y coordinate, and d_{pe} is equal to $d_{model}/2$. Here sup represents

the index of the superpixel (from 0 to $N_s - 1$), and i covers the d_{pe} dimensions. Afterward, $SinPE^y$ and $SinPE^x$ are concatenated along the channel dimension:

$$SinPE = (SinPE^x \,||\, SinPE^y) \tag{2}$$

with $SinPE$ having shape $N_s \times d_{model}$. After computing $SinPE$ following Equation 2, we obtain a $N_s \times d_{model}$ encoding of the (x,y) centroids coordinates $\mathcal{C}$. We now have a sinusoidal positional encoding for each superpixel extracted from the input image. Then, following the same insight as in ❸, we replicate the *absolute* sinusoidal positional encoding of the i-th centroid $SinPE_i$ over the surface of the i-th superpixel. Finally, we obtain $SinPE_{\mathcal{L}}$, with shape $H \times W \times d_{model}$, from $\mathcal{L}$ by replacing the pixels belonging to a specific superpixel with the sinusoidal positional encoding of its centroid.

We also propose a *relative* superpixel positional encoding strategy which exploits the progressive index sup assigned to each superpixel by the algorithm $\mathcal{S}$. This index is an integer in the range $[0, N_s - 1]$ that follows the row-major order, *i.e.*, index 0 corresponds to the first superpixel in the top-left corner and index $(N_s - 1)$ corresponds to the final superpixel in the bottom-right corner. We compute a linear *relative* positional encoding by normalizing the superpixel indices in the interval $[0, 1]$:

$$LinearPE_{(sup)} = \frac{sup}{N_s - 1} \tag{3}$$

with $LinearPE$ having shape N_s. We replicate the linear encoding of the superpixel index over the channel size to match the latent space size of the encoder, resulting in a $LinearPE$ with shape $N_s \times d_{model}$. As done for the sinusoidal encoding, we follow the procedure described in ❸ and replicate over the surface of the i-th superpixel the relative linear positional encoding $LinearPE_i$ of its index, having shape d_{model}. In other words, we obtain $LinearPE_{\mathcal{L}}$, with shape $H \times W \times d_{model}$, from $\mathcal{L}$ by replacing the pixels belonging to a specific superpixel with the linear positional encoding of its index. For both of the proposed approaches, $PE_{\mathcal{L}}$ is spatially downsampled to match the shape of f^j , and added to it, as depicted in Figure 2.

In our experiments, we compare the proposed superpixel-based PE with several alternative strategies for encoding position and incorporating superpixel shape and position priors into ViT encoder features, as outlined in the following section.

4 Experiments

In this section, we first present the adopted datasets, followed by the superpixel extraction algorithm, and the considered semantic segmentation architectures. Finally, we outline the results of our proposed superpixel positional encoding strategies.

Datasets. We evaluate our proposed superpixel-based techniques over two publicly available datasets: ADE20K [59] and Cityscapes [13]. ADE20K is a challenging scene-parsing dataset with 150 fine-grained semantic concepts. It comprises a training set of 20210 images, a validation set of 2000 images, and a testing set of 3352 images. On the other hand, Cityscapes is a driving dataset for semantic segmentation consisting of 5000 fine-annotated high-resolution images, split into 2975, 500, and 1525 images for training, validation, and testing respectively. It densely annotates 19 object categories in images with urban scenes.

Superpixel algorithm. As our superpixel extraction algorithm, we adopt an optimized variant of SLIC [1], named FastSLIC[1]. To enhance efficiency, we integrate it into the data loading

[1] https://github.com/Algy/fast-slic

process. FastSLIC is designed with various optimization techniques, such as color quantization, integer-only arithmetic, row subsampling, and multicore parallelization. In the Supplementary, we report an in-depth analysis of the impact of superpixel extraction on data loading, training, and inference, and explore alternative superpixel algorithms.

DPT. Dense Prediction Transformer (DPT) consists of an encoder-decoder architecture that employs Vision Transformers for semantic segmentation. The encoder uses a stack of transformer blocks to extract feature maps at multiple stages. The decoder reconstructs image-like feature representations and progressively fuses them into the final dense prediction through residual convolutional units. Our experiments incorporate the proposed superpixel-PE by summing it to the feature representations extracted by the encoder prior to the fusion operation applied by the decoder. We consider the variant of DPT employing a ViT-Base backbone. DPT uses random horizontal flipping and random re-scaling, with a batch size of 16 and square random crops of size 480 and 512 extracted from ADE20K and Cityscapes images. We use a cross-entropy loss, AdamW optimizer, and cosine learning rate scheduler, with a learning rate set to $(0.002 \cdot bs/512)$.

SegFormer. The SegFormer model is based on a hierarchical Transformer encoder to extract four feature maps (c_1, c_2, c_3, and c_4) at different resolutions, ranging from the high-resolution fine-grained features of c_1 to the low-resolution coarse-grained features of c_4. An MLP-based decoder then fuses these features and predicts the dense segmentation mask. We downsample and sum the proposed superpixel-based PE to the encoder outputs and feed the result to the MLP decoder. SegFormer provides encoders of different sizes, from B0 to B5, and we measure the gain of our proposed approach on the B0 and B4 encoders. SegFormer uses random resizing, horizontal flipping, and cropping to 512. It employs an AdamW optimizer for 160K iterations with a batch size of 16 for B0 and 8 for B4 on ADE20K, and a batch size of 8 on Cityscapes for both B0 and B4. The learning rate is set to 6e$-$5.

SETR. We use the multi-level feature aggregation (MLA) variant of SETR, which produces the best results on the segmentation task [53]. SETR-MLA (SETR to shorten) consists of an encoder-decoder ViT-based segmentation architecture. The encoder consists of a series of Transformer blocks that extract visual features at different layers. The encoder reshapes, upsamples, and fuses the visual features through a series of convolutional blocks to generate the final dense prediction. We incorporate our superpixel-PE by summing it to the reshaped visual features before these are fused by the last stack of convolutional layers. We consider two variants of SETR employing a ViT-Tiny and ViT-Small backbone respectively. SETR uses random resizing, horizontal flipping, and cropping to 512. It employs an SGD optimizer for 160K iterations with a batch size of 4 on ADE20K, and for 80K iterations with a batch size of 8 on Cityscapes, for both the Tiny and Small variants. The learning rate is set to 1e$-$3 and 2e$-$3 for ADE20K and Cityscapes respectively, with a momentum of 0.9.

4.1 Results

Superpixel-based PE. Table 1 presents mean IoU results for DPT with ViT-Base backbone, SegFormer with B0 and B4 backbones, and SETR with ViT-Tiny and ViT-Small backbones, with and without our absolute sinusoidal and relative linear superpixel-based positional encoding on ADE20K and Cityscapes datasets. For ADE20K, employing our sinusoidal and linear positional encoding improves mIoU from 44.9 to 45.4 and 45.8, respectively, for DPT-B, from 37.5 to 38.4 and from 49.0 to 49.3, respectively, for SegFormer-B0 and SegFormer-B4, from 35.2 to 36.3 and from 42.7 to 43.4, respectively, for SETR-T and SETR-S. For

Table 1: Results of mIoU on ADE20K and Cityscapes using DPT-Base, SegFormer-B0, SegFormer-B4, SETR-Tiny and SETR-Small with/without superpixel positional encoding.

	#Superpixel	Compact.	Params (M)	ADE20K (mIoU)	Cityscapes (mIoU)
DPT-B [57]	-	-	102.0	44.9	71.0
DPT-B+SinPE$_{\mathcal{L}}$	16,000	20	102.0	45.4	71.7
DPT-B+LinearPE$_{\mathcal{L}}$	28,000	10	102.0	**45.8**	**72.0**
SegFormer-B0 [63]	-	-	3.8	37.5	71.4
SegFormer-B0+SinPE$_{\mathcal{L}}$	16,000	20	3.8	38.2	71.8
SegFormer-B0+LinearPE$_{\mathcal{L}}$	28,000	10	3.8	**38.4**	**72.2**
SegFormer-B4 [63]	-	-	64.1	49.0	78.4
SegFormer-B4+SinPE$_{\mathcal{L}}$	16,000	20	64.1	**49.3**	**78.6**
SegFormer-B4+LinearPE$_{\mathcal{L}}$	28,000	20	64.1	**49.3**	**78.6**
SETR-T [58]	-	-	10.2	35.2	69.3
SETR-T+SinPE$_{\mathcal{L}}$	8,192	10	10.2	**36.3**	**70.1**
SETR-T+LinearPE$_{\mathcal{L}}$	16,384	10	10.2	36.1	70.0
SETR-S [58]	-	-	26.7	42.7	74.6
SETR-S+SinPE$_{\mathcal{L}}$	8,192	10	26.7	43.0	74.9
SETR-S+LinearPE$_{\mathcal{L}}$	8,192	10	26.7	**43.4**	**75.0**

Cityscapes, mean IoU improves from 71.0 to 71.7 and 72.0 for DPT-B with sinusoidal and linear encoding, respectively, from 71.4 to 72.2 and from 78.4 to 78.6 for SegFormer-B0 and SegFormer-B4, from 69.3 to 70.1 and from 74.6 to 75.0 for SETR-T and SETR-S. We observe that a large number of superpixels is necessary to ensure the fine-grained property for $\mathcal{L}$ and Sin/Linear-PE$_{\mathcal{L}}$ consequently. The obtained results show that our positional encoding strategies are effective in improving segmentation performance, as also qualitatively shown in Figure 3. See Supplementary for analysis of varying downsampling strategies and point of injection of superpixel-PE in encoder-decoder architectures.

Pixel-based PE. We investigate how much of the performance boost achieved by our superpixel-PE can be ascribed to positional encoding and how much to the injection of shape and edge priors provided by the superpixels. This involves adapting the sinusoidal positional encoding from Section 3 to individual pixel coordinates. The resulting *Pixel-PE* is added to encoder features, but it only provides a minor performance gain, compared to our superpixel-PE. Our results suggest that the shape and edge priors provided by superpixels are critical for improving segmentation performance (2nd row of Table 2).

Patch-based PE. We also compare with a positional encoding method based on the square shape of the patches extracted from the input image and used by the ViT encoder to generate feature representations. Instead of encoding superpixels, we encode the numeric index that progressively identifies each patch, similar to the LinearPE approach. The number of patches determines the resolution of the positional encoding mask. Patch-size values show similar performance for both datasets, even at an extreme patch size of 1. This positional encoding method lacks shape and edge priors provided by superpixels, resulting in reduced performance compared to our superpixel-based PE, as shown in Table 2 (3rd to 5th rows).

Weighted Superpixel-PE. We introduce the *WeightedPE* method, which combines encoder features and superpixel-based PE through trainable parameters. This approach helps us understand how superpixel priors relate to the different resolutions of encoder features. Instead of downsizing and summing the superpixel-based PE to the four outputs of the SegFormer

Table 2: Comparison of our sinusoidal and linear superpixel-PE with different positional encoding strategies of injecting superpixel shape and position priors in encoder features.

	#Superpixel	Compact.	Params (M)	ADE20K (mIoU)	Cityscapes (mIoU)
SegFormer-B0 [48]	-	-	3.8	37.5	71.4
+PixelPE	-	-	3.8	37.8	71.5
+PatchPE-1	-	-	3.8	37.6	71.7
+PatchPE-4	-	-	3.8	37.5	71.7
+PatchPE-16	-	-	3.8	37.7	71.7
+WeightedPE$_{\mathcal{L}}$	16,000	20	3.8	37.7	**72.4**
+LearnablePE$_{\mathcal{L}}$	4,096	10	4.8	38.3	**72.2**
+SinPE$_{\mathcal{L}}$	16,000	20	3.8	38.2	71.8
+LinearPE$_{\mathcal{L}}$	28,000	10	3.8	**38.4**	**72.2**

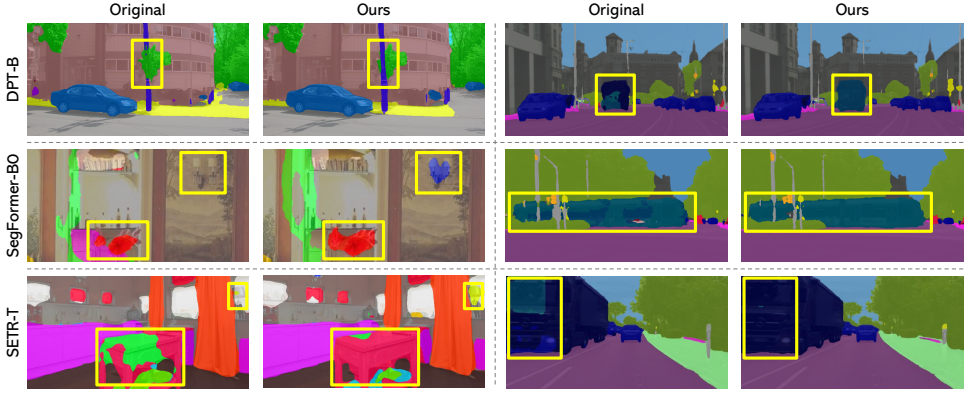

Figure 3: We present sample results obtained employing the SegFormer-B0 [48], DPT-B [37] and SETR-T [58] models, both with and without our superpixel positional encoding.

encoder as in previous experiments, *WeightedPE* uses a weighted sum of encoder features and superpixel-based PE. The learnable weights are four parameters, one for each feature map generated by the encoder, all initialized to 0.10. For ADE20K, the learnable weights β_1, β_2, β_3, and β_4 were found to be 0.60, 0.51, 0.19, and 0.03, while for Cityscapes, they were 0.40, 0.30, 0.12, and 0.01, respectively. These results indicate that superpixel-PE is particularly beneficial for training high-resolution feature maps that preserve edge details between semantic classes. The proposed *WeightedPE* approach improves performance on both datasets, as shown in the sixth row of Table 2.

Learnable Superpixel-PE. The techniques discussed so far can be used in encoder-decoder models without modifying the architecture or adding parameters, except for the four learnable weights in *WeightedPE*. However, to further leverage superpixel information, we propose LearnPE, which uses a learnable superpixel encoding. Each superpixel is assigned a learnable vector, allowing the model to adaptively learn the optimal encoding for each superpixel during training. LearnPE's performance is comparable to SinPE and LinearPE, as shown in Table 2 (7th row), but with an increase in the number of model parameters. In the Supplementary we present a study demonstrating that as the number of superpixels increases, the number of parameters grows significantly, while the mIoU performance tends to decrease, suggesting a trade-off between model complexity and performance.

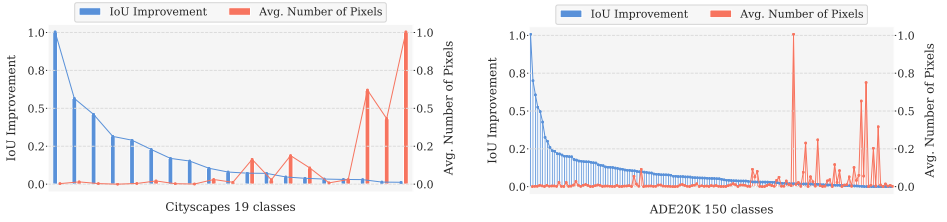

Figure 4: Relationship between per-class normalized IoU improvement and per-class normalized average pixels across Cityscapes and ADE20K datasets.

Impact of superpixels on semantic classes. Our Superpixel PE approach improves mIoU scores on both Cityscapes and ADE20K datasets. We analyze the effect on individual classes and find that classes with lower average pixel count, such as `traffic light`, `flower`, and `sconce` in ADE20K and `bus`, and `train` in Cityscapes, show consistent improvements. This suggests that our approach is effective for infrequent classes and helps avoid overfitting highly represented classes. This represents a significant advantage, as accurate segmentation of less frequent classes is often challenging due to limited data availability and class imbalance issues. We present a comparison improvement in IoU and the average number of pixels per class in Figure 4.

5 Conclusion

In this work, we investigated the effectiveness of integrating superpixels in Transformer-based architectures for semantic segmentation. We proposed a novel superpixel-based positional encoding strategy that injects superpixel shape and position priors into the ViT encoder features, creating more boundary-aware semantic latent space representations. Experimental evaluation on ADE20K and Cityscapes datasets with different architectures demonstrated significant mIoU improvement. Our approach performs well on rare classes while avoiding overfitting on common classes, ensuring a balance between generalization and specificity. Overall, our work highlights the potential of integrating classical perception-based computer vision approaches with self-attentive architectures for dense prediction tasks.

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. PAMI*, 2012.

[2] Roberto Amoroso, Lorenzo Baraldi, and Rita Cucchiara. Assessing the role of boundary-level objectives in indoor semantic segmentation. In *CAIP*, 2021.

[3] Roberto Amoroso, Lorenzo Baraldi, and Rita Cucchiara. Improving indoor semantic segmentation with boundary-level objectives. In *IWANN*, 2021.

[4] Wanda Benesova and Michal Kottman. Fast superpixel segmentation using morphological processing. In *Proceedings of the Conference on Machine Vision and Machine Learning*, 2014.

[5] Gedas Bertasius, Jianbo Shi, and Lorenzo Torresani. Semantic segmentation with boundary neural fields. In *CVPR*, 2016.

[6] Paolo Bruno, Roberto Amoroso, Marcella Cornia, Silvia Cascianelli, Lorenzo Baraldi, and Rita Cucchiara. Investigating bidimensional downsampling in vision transformer models. In *ICIAP*, 2022.

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[8] Liang-Chieh Chen, Jonathan T Barron, George Papandreou, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with task-specific edge detection using cnns and a discriminatively trained domain transform. In *CVPR*, 2016.

[9] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. PAMI*, 2017.

[10] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[11] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[12] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the Design of Spatial Attention in Vision Transformers. In *NeurIPS*, 2021.

[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 *CVPR*, 2016.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2018.

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ICLR*, 2021.

[16] Alaaeldin El-Nouby, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. XCiT: Cross-Covariance Image Transformers. *arXiv preprint arXiv:2106.09681*, 2021.

[17] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 2004.

[18] Gianni Franchi, Nacim Belkhir, Mai Lan Ha, Yufei Hu, Andrei Bursuc, Volker Blanz, and Angela Yao. Robust Semantic Segmentation with Superpixel-Mix. *arXiv preprint arXiv:2108.00968*, 2021.

[19] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *CVPR*, 2019.

[20] Leo Grady. Random walks for image segmentation. *IEEE Trans. PAMI*, 2006.

[21] Junjun He, Zhongying Deng, Lei Zhou, Yali Wang, and Yu Qiao. Adaptive pyramid context network for semantic segmentation. In *CVPR*, 2019.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[23] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *CVPR*, 2019.

[24] Jitesh Jain, Anukriti Singh, Nikita Orlov, Zilong Huang, Jiachen Li, Steven Walton, and Humphrey Shi. Semask: Semantically masking transformer backbones for effective semantic segmentation. *arXiv*, 2021.

[25] Varun Jampani, Deqing Sun, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. Superpixel sampling networks. In *ECCV*, 2018.

[26] Xia Li, Zhisheng Zhong, Jianlong Wu, Yibo Yang, Zhouchen Lin, and Hong Liu. Expectation-maximization attention networks for semantic segmentation. In *ICCV*, 2019.

[27] Zhengqin Li and Jiansheng Chen. Superpixel segmentation using linear spectral clustering. In *CVPR*, 2015.

[28] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *CVPR*, 2017.

[29] Ming-Yu Liu, Oncel Tuzel, Srikumar Ramalingam, and Rama Chellappa. Entropy rate superpixel segmentation. In *CVPR*, 2011.

[30] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. *ICCV*, 2021.

[31] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[32] Vaïa Machairas, Etienne Decencière, and Thomas Walter. Waterpixels: Superpixels based on the watershed transformation. In *ICIP*, 2014.

[33] Peer Neubert and Peter Protzel. Compact watershed and preemptive slic: On improving trade-offs of superpixel segmentation algorithms. In *ICPR*, 2014.

[34] Zizheng Pan, Bohan Zhuang, Jing Liu, Haoyu He, and Jianfei Cai. Scalable vision transformers with hierarchical pooling. In *ICCV*, 2021.

[35] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *CVPR*, 2017.

[36] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training, 2018.

[37] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, 2021.

[38] Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *ICCV*, 2003.

[39] David Stutz, Alexander Hermans, and Bastian Leibe. Superpixels: An evaluation of the state-of-the-art. *CVIU*, 2018.

[40] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[42] Jie Wang and Xiaoqiang Wang. VCells: Simple and efficient superpixels using edge-weighted centroidal Voronoi tessellations. *IEEE Trans. PAMI*, 2012.

[43] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE Trans. PAMI*, 2020.

[44] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions. In *ICCV*, 2021.

[45] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

[46] David Weikersdorfer, David Gossow, and Michael Beetz. Depth-adaptive superpixels. In *ICPR*, 2012.

[47] Kan Wu, Houwen Peng, Minghao Chen, Jianlong Fu, and Hongyang Chao. Rethinking and improving relative position encoding for vision transformer. In *ICCV*, 2021.

[48] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers. In *NeurIPS*, 2021.

[49] Zhiwei Xu, Thalaiyasingam Ajanthan, and Richard Hartley. Refining Semantic Segmentation with Superpixel by Transparent Initialization and Sparse Encoder. *arXiv preprint arXiv:2010.04363*, 2020.

[50] Fengting Yang, Qian Sun, Hailin Jin, and Zihan Zhou. Superpixel segmentation with fully convolutional networks. In *CVPR*, 2020.

[51] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Learning a discriminative feature network for semantic segmentation. In *CVPR*, 2018.

[52] Changqian Yu, Jingbo Wang, Changxin Gao, Gang Yu, Chunhua Shen, and Nong Sang. Context prior for scene segmentation. In *CVPR*, 2020.

[53] Bowen Zhang, Zhi Tian, Quan Tang, Xiangxiang Chu, Xiaolin Wei, Chunhua Shen, and Yifan Liu. Segvit: Semantic segmentation with plain vision transformers. *NeurIPS*, 2022.

[54] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *CVPR*, 2018.

[55] Yifan Zhang, Bo Pang, and Cewu Lu. Semantic segmentation by early region proxy. 2022.

[56] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017.

[57] Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia. Psanet: Point-wise spatial attention network for scene parsing. In *ECCV*, 2018.

[58] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, 2021.

[59] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *CVPR*, 2017.