A Model-based Curriculum Learning Strategy for Training SegFormer

A Model-based Curriculum Learning Strategy for Training SegFormer

Daniele Rege Cambrin
Dip. di Automatica e Informatica
Politecnico di Torino
Turin, Italy
daniele.regecambrin@polito.it

Daniele Apiletti
Dip. di Automatica e Informatica
Politecnico di Torino
Turin, Italy
daniele.apiletti@polito.it

Paolo Garza
Dip. di Automatica e Informatica
Politecnico di Torino
Turin, Italy
paolo.garza@polito.it

***Abstract*—The use of Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs) in computer vision opened up new tracks in this area. However, a significant drawback of these models is the large amount of data required to obtain competitive results. This critical issue limits their application in domains where large labeled data collections are unavailable. Some strategies have been proposed to use relatively limited labeled data sets to train CNN-based models. Curriculum learning is one of the currently available strategies to train deep learning models faster and with less data. However, to our knowledge, curriculum learning techniques have never been used at the model level to support ViT training for semantic segmentation. We propose a new curriculum learning technique tailored to ViT models to fill this gap. The results show the effectiveness of the proposed strategy in training ViT models from scratch to solve the semantic segmentation task.**

***Index Terms*—Deep learning, Curriculum learning, Semantic segmentation, and Computer vision**

I. Introduction

In the last decade, deep learning models and Big data collections have revolutionized the field of computer vision. First, through Convolutional Neural Networks (CNNs) and later through Vision Transformers (ViTs), the performance of image processing tasks (e.g., image classification and semantic segmentation) has been greatly enhanced. However, training these models requires vast collections of labeled data, which are limited in some domains (e.g., satellite imagery for emergency management [1] and medical applications). To address this problem, researchers have proposed methods to train deep learning models more effectively using the limited amount of available labeled data.

In this paper, we focus on the task of semantic segmentation and propose a new curriculum learning strategy to support the training of ViT models.

Semantic segmentation is a computer vision task that plays an important role in various application domains such as medical imaging [2], autonomous driving [3], urban planning [4], and emergency management [1]. It differs from image classification, as it focuses on predicting the category of each individual pixel rather than providing a prediction at the image level.

Over the years, several Convolutional Neural Networks (CNNs), such as VGG [5], DeepLab-V3 [6], and U-Net [7], have been introduced and proven effective for semantic segmentation. Recently, Vision Transformers [8] have demonstrated their capabilities in various computer vision tasks, including semantic segmentation. However, training these deep learning models requires a large amount of labeled data, which is not feasible in some domains. Intelligent training methods have been proposed to solve this problem. For example, techniques inspired by human learning (the curriculum learning methods) have been proposed to effectively guide the training of CNNs. These techniques can either improve the quality of results by considering all available data or produce high-quality results with less data. The curriculum learning techniques can be categorized based on the training phase they target. Precisely, curriculum learning strategies can target (i) the architecture of the trained model, (ii) how the data are used, or (iii) the complexity of the subtasks used to teach the model to solve increasingly complex problems.

To our knowledge, some studies on curriculum learning applied to CNNs for semantic segmentation have been proposed (e.g., [9], [10]). Still, they are based on something other than curriculum learning strategies working at the model level. Moreover, nobody attempted to use curriculum learning at the model/architecture level for training ViT models to solve the semantic segmentation task. Only one previous work [11] focused on a curriculum learning approach that, similarly to ours, increases step-by-step the deep learning model complexity. However, it has been applied to a CNN-based generative adversarial network that generates high-quality images, i.e., a different network architecture and task.

Our contribution can be summarized as follows.

- We propose a novel curriculum learning strategy that works at the model level and gradually increases the complexity of the network. Increasing the complexity of the model gradually usually allows faster training of models with less labeled data. The proposed approach is tailored to a (non-symmetric) ViT model. No previous work has proposed model-level curriculum learning strategies that deal with non-symmetric deep learning networks. Previous works targeted CNN-based architectures.
- We have thoroughly evaluated the effectiveness of the proposed curriculum learning strategy in solving the

semantic segmentation task using ViT models. No one has yet attempted to solve this task using ViT models trained with a model-level curriculum learning strategy.
- A publicly available version of the code for reproducibility at https://github.com/DarthReca/curriculum-segformer to favor new research in this area.

The rest of the paper is organized as follows. Section II summarizes the related work. Section III formally introduces the addressed task, while Section IV describes the proposed methodology. The experimental evaluation is reported in Section V. Finally, Section VI draws conclusions and discusses future work.

II. Related Works

A. Vision Transformers

In recent years, the introduction of Vision Transformers has demonstrated their effectiveness in various computer vision tasks. However, training Vision Transformers can be more resource intensive than other deep learning models because self-attention scales quadratically with image resolution. In addition, transformers have difficulty capturing spatial invariance [8] compared to CNNs. Many models leverage pre-training on large datasets such as ImageNet [12] to mitigate this problem. However, pre-training on ImageNet or other well-known datasets cannot be used when dealing with non-RGB data (e.g., Sentinel-2 imagery characterized by 12 channels). In this case, the models must be trained from scratch, and training techniques, such as curriculum learning, are needed to use the limited amount of labeled data available effectively.

B. Curriculum Learning

The curriculum learning methodology was introduced by Bengio et al. in [13]. The concept of curriculum learning has shown its benefits in different fields, including computer vision [11], natural language processing [14], and reinforcement learning [15]. The approach proposed in [13] involves gradually exposing the models to increasingly complex examples/input data during training to mimic human learning behavior. This methodology was applied by starting with simpler samples and progressively introducing more sophisticated examples as the training progressed.

Curriculum learning strategies can act at different levels: (i) at the data level [14], increasing the complexity of the training data while the training progresses, (ii) at the task level [9], [15], [16], proposing tasks of increasing difficulty to the network we are training, or (iii) at the model level [11], updating the model complexity periodically. Regardless of the strategy used, the basic idea is gradually increasing the complexity. Similar to humans, the model first learns to solve simple cases. Then it gradually increases its expertise and solves more complex cases.

1) Curriculum learning at the data level: The application of curriculum learning at the data level has proven successful in many tasks in various domains, such as natural language processing and computer vision. When applying curriculum learning at the data level, the training process is first fed with simple data samples. Then, more data, representative of more complex cases, are used to continue training the model. The amount and complexity of data increases until convergence is achieved. In this type of strategy, the architecture of the trained model does not change over time. Thus, we only use intelligent data-feeding strategies to train a more effective model. Many different criteria were used for evaluating the difficulty of the samples, ranging from simple shape analysis [13] to more complex factors such as the presence of fog [10]. More advanced techniques were explored, such as considering how much we learn from each sample [17] to optimize the training.

2) Curriculum learning at the model level: The complexity of the architecture increases periodically when curriculum learning is applied at the model level. We start with a simple architecture, and after a certain number of steps, the complexity of the architecture increases. For example, we begin with a multilayer perceptron network with N hidden layers and gradually increase the complexity by adding more hidden layers. It is difficult to change the model architecture during training and keep the information learned so far. For this reason, few curriculum learning strategies have been proposed at the model level. The proposed approaches may gradually act on the number of layers of the network [11] or on dropout layers [18] or on the embeddings of convolutions [19]. However, these approaches are often limited to specific models or individual components of deep learning networks. Further research is needed to propose universal curriculum learning strategies independent of the trained architecture/model.

The curriculum learning strategy we propose in this paper acts at the model level. Differently from the previous works [11], it is the first attempt to apply a curriculum learning strategy at the model level on a Vision Transformer.

III. Problem Statement

The addressed task is semantic segmentation, which can be defined as follows. Given a set of classes $\mathcal{C}$ and a set of unlabeled images $\mathcal{U}$, the objective of the semantic segmentation task is to assign a class label $c \in \mathcal{C}$ to each pixel of the images $I \in \mathcal{U}$ by using a model M trained on a set of images $\mathcal{T}$ for which the class labels of their pixels are known.

More precisely, we have at our disposal a set of images $\mathcal{T}$ of size $W \times H \times D$ (where W, H, and D are the width, height, and depth of the images, respectively) and a classification model M trained on the training set $\mathcal{T}$. We have an associated mask of size $W \times H$ for each image in $\mathcal{T}$, where each pixel/cell assumes a value from 1 to N_{cls} (where N_{cls} is the number of distinct classes). The goal consists in training a model M that learns how to predict the masks for the unlabeled images in $\mathcal{U}$. The images in $\mathcal{U}$ and $\mathcal{T}$ have the same features, but $\mathcal{U} \cap \mathcal{T}$ is the empty set.

IV. Methodology

This paper proposes a solution to the semantic segmentation problem based on a novel curriculum learning strategy applied to the well-known SegFormer vision transformer [20]. SegFormer is a well-known model that can effectively address semantic segmentation. For completeness and clarity, Figure 1 provides the architecture of the SegFormer from the original paper [20], which consists of four transformer blocks in the encoder part and some MultiLayer Perceptron (MLP) layers in the decoder to obtain the final prediction. The following refers to the transformer blocks as blocks or layers.

The proposed curriculum learning strategy was applied at the model level. Moreover, other techniques to stabilize the training were also employed (see Sections IV-B IV-C, and IV-D). Specifically, we propose first to train a simple SegFormer using only the first encoder block. Then, after a certain number of iterations, the second encoder block is added to the model to specialize it further (adding a block means activating it). This procedure continues until all encoder blocks are activated, i.e., added to the model (see Section IV-A for details). Compared to a traditional training procedure where the most complex SegFormer model is trained directly with all blocks, our proposed curriculum learning approach aims at a better initialization of the parameters of the first blocks.

To manage the increasing complexity of the model, the training data are split into partitions (see Section IV-B). Specifically, a percentage of the training data equal to $dataPercentageStep$ is used initially to train the initial model based on a single block. Then, each time a new block is added, the percentage of training data considered increases proportionally. This partitioning approach helps preventing overfitting. Finally, embedding smoothing IV-C and a cyclical learning rate scheduler IV-D are used in our solution to optimize the training of the model further.

The procedure can be found in Figure 2, where $maxLayers$ is the number of layers (blocks) of the model to add gradually, $maxTrainingSteps$ is the number of training steps, and $activationSteps$ is the set of steps in which the new layers are added. For consistency, the number of activation steps must be equal to $maxLayers$, i.e., the number of layers to add. The function on line 9 executes the training loop for the batch at step s, having access only to $dataPercentage$ portion of the training data. A detailed description of each step of the proposed solution is provided below. In the following descriptions, we use a running example with a ViT network composed of 4 layers. Specifically, we use the following assumption: $maxLayers = N$, $activationSteps = \{T_0, \ldots, T_{N-1}\}$, and N (the number of layers to add/activate) is 4.

A. Progressive Model Growth

This is the novel curriculum learning strategy that we propose. The idea is to generalize the pattern proposed in [11] to our transformer architecture. Since we use a ViT model, the approach proposed in [11] cannot be directly applied in our scenario, as they add the model layers to a Generative Adversarial Network (GAN) one by one. In our case, these layers are transformer blocks of the encoder.

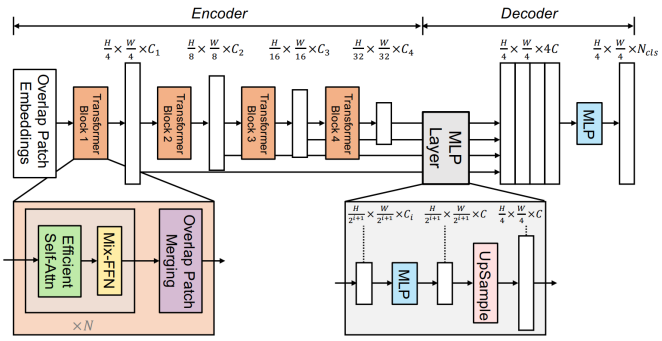


Fig. 1: SegFormer architecture. In *white*, the embeddings and the respective sizes. W and H are the respective width and height of the input image, C_x is a generic number of channels, and N_{cls} is the number of classes to predict in the task.

Require: $maxLayers > 0$
Require: $maxTrainingSteps > 0$
Require: $activationSteps \subseteq \{0, \ldots, maxTrainingSteps\}$
Ensure: $|activationSteps| = maxLayers$
1: $dataPercentageStep \leftarrow 1/maxLayers$
2: $dataPercentage \leftarrow 0$
3: **for** $s \leftarrow 0$ to $maxTrainingSteps$ **do**
4: **if** s in $activationSteps$ **then**
5: `ActivateNextBackboneLayer()`
6: `EnlargeFusionLayer()`
7: $dataPercentage \mathrel{+}= dataPercentageStep$
8: **end if**
9: `TrainingLoop(`$dataPercentage$`)`
10: **end for**

Fig. 2: Curriculum-Enhanced Segmentation Algorithm

At the time T_0 (first activation step), only one block is active (see Figure 3). We start training the model with one single active layer (block). When we reach a stability situation at time T_1 (second activation step), we activate the second encoder layer (see line 5 of Figure 2 and the second image from the top in Figure 3). This action provides a new stimulus to the network since the resulting model, and embeddings will be more complex and capable of capturing details. This is done for each timestep in $activationSteps$, stabilizing the learning of the previous layers and speeding up the convergence of the newly introduced ones. The ViT model is composed of the encoder and decoder parts. The decoder must be updated when further encoder layers are activated in the network. At the time T_0, the deepest layer of the decoder processes a single embedding, which is fed into a Linear Fusion layer (Convolutional layer) (see Figure 4). At timestep T_1, the processed embeddings are two (corresponding to the outputs of the two encoder/transformer blocks 1 and 2 reported in Figure 4). Thus, linear fusion must process a tensor with

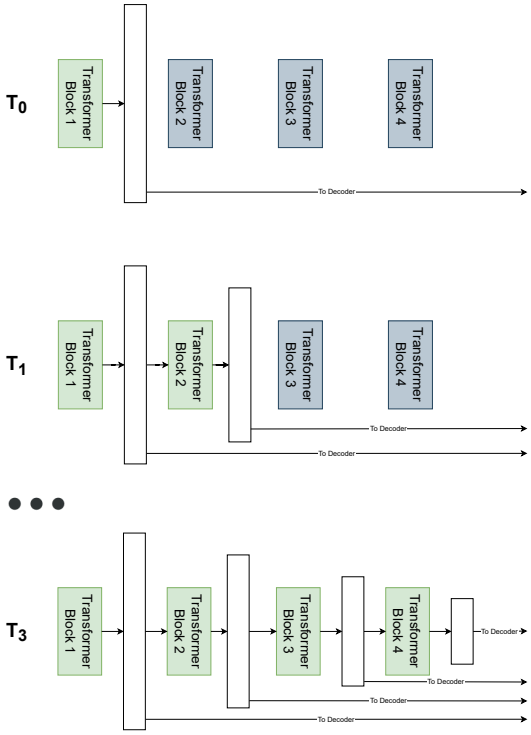


Fig. 3: Curriculum learning applied to the encoder at different timesteps. The active blocks are colored in *green*, while the inactive ones are in *grey*. The embeddings are sent to the decoder.

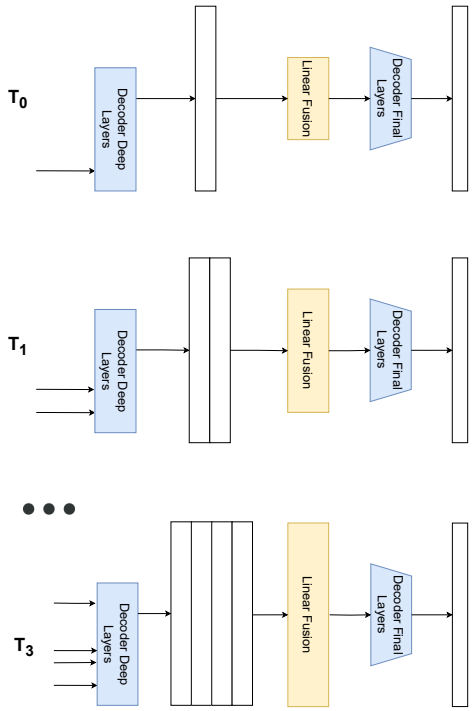


Fig. 4: Curriculum learning applied to the decoder at different timesteps. The linear fusion layer, highlighted in *yellow*, grows at each timestep. In *blue*, the static layers of the decoder. The final prediction is the output of *Decoder Final Layers*.

multiple channels. The simplest, but at the same time effective approach to deal with this change in the number of embeddings is to replace the linear fusion layer with a new layer of a different size (see line 6 and the second figure from the top in Figure 4). Since it is a simple convolution, it can quickly learn from scratch without significantly affecting the rest of the network. The procedure is applied at each time step in $activationSteps$.

The curriculum learning procedure applied to SegFormer is depicted in Figures 3 and 4 with $maxLayers = 4$ and $activationSteps = \{T_0, T_1, T_2, T_3\}$.

B. Dataset Portioning

As the network becomes more complex and powerful over time, we initially provide it with only a subset of the training data. Then we increase the number of samples each time we add a new encoder block, as shown in Figure 2 at line 7. This helps prevent overfitting, as the model gets another stimulus every time a new layer is added. For simplicity, we defined the increment $dataPercentageStep = 1/maxLayers$, to increase the training data uniformly with the increasing complexity of the SegFormer model. The validation set, on the other hand, always remains the same.

C. Embedding Smoothing

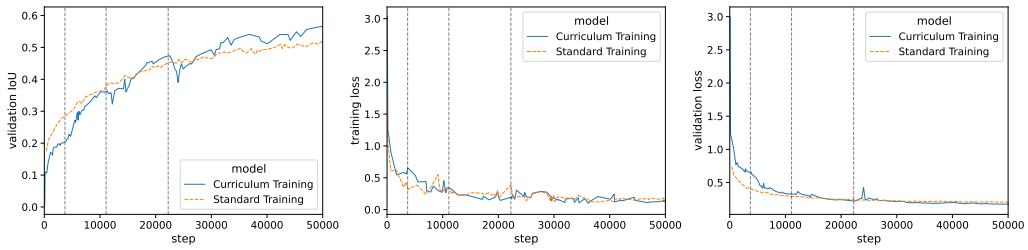
Since the embeddings of a newly inserted layer are noisy, we used a smooth insertion as suggested in [11]. Before linear fusion, the new embeddings are replaced by a linear interpolation between the representation of the previous layer and the current one for a certain number of steps S. In this way, the network can slowly adjust the weights of the randomly initialized layer without spikes in the loss optimization.

D. Cyclical Learning Rate Scheduler

Inspired by the cyclical learning rate scheduler [21], we used this approach to give the network a new exploration push when a new layer is added. When a new layer is added/activated in a time step T_i, the learning rate is set to lr_{min}. Then the learning rate increases linearly up to lr_{max}. When the maximum learning rate is reached, the scheduler decreases the learning rate until it reaches lr_{min}. This is defined as a cycle of the scheduler. After all the layers have been activated, we follow the *triangular2* policy [21] multiplying lr_{max} by 0.5 after the end of each cycle.

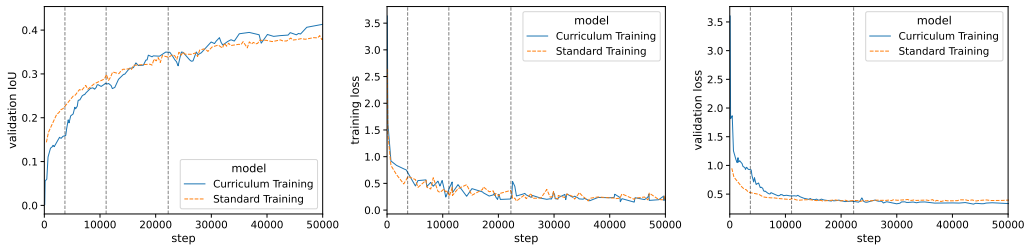
V. Experimental Results

The following section presents the results obtained with a standard benchmark dataset: CityScapes [3]. The evaluation metric is the same one used in CityScapes. More specifically, the mean Intersection Over Union (IoU) or Jaccard index, a standard semantic segmentation metric, was used.



(a) Validation IoU (b) Training loss (c) Validation loss

Fig. 5: CityScapes: 19 classes. Vertical grey dashed lines represent the insertion of a new layer.



(a) Validation IoU (b) Training loss (c) Validation loss

Fig. 6: CityScapes: 34 classes. Vertical grey dashed lines represent the insertion of a new layer.

A. CityScapes Data Set

The data set comprises RGB images of size $1024 \times 2048 \times 3$. It includes 2975 samples for the training set and 500 for the validation set. We used the same data set split used in [3]. The annotation comprises 34 classes, ranging from the class *person* to class *sky*. Some classes, such as *road*, are more present with values higher than 10^6 pixels, while others, such as *bus*, are rare, with values around 10^3 pixels.

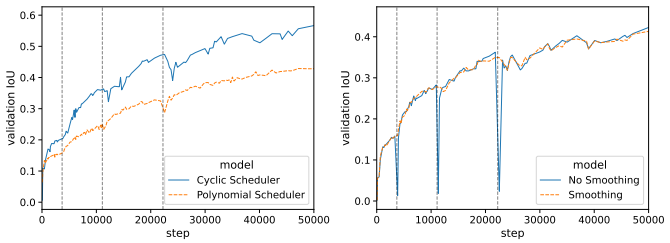
B. Experimental Settings

Since the proposed approach can be applied indiscriminately to any version of SegFormer, we chose the lighter version to train on a single A6000. We use the settings given in the original paper [20]. We trained all models for $maxTrainingSteps = 50000$. We compared the model trained using our curriculum learning with a SegFormer model trained using a traditional learning approach. The input parameters of our approach are determined as follows. Since there are four transformer blocks, we set $maxLayers = 4$. A new layer is activated every 50 training epochs, so we set $activationStep = \{0, 3700, 11100, 22250\}$. Finally, the smoothing parameter was set to $S = 8000$ steps. The gap in terms of steps between consecutive activation steps in $activationStep$ varies because we are gradually increasing the number of training samples (see Section IV-B), so the initial epochs are shorter than the latter. The cyclical learning rate scheduler uses $lr_{min} = 10^{-7}$, $lr_{max} = 10^{-3}$ and $P_{up} = 0.1$. According to the original paper, the Cross-Entropy loss is used, and the batch size was set to 8. To perform a fair comparison, we used the same loss (the Cross-Entropy loss), the same batch size (8), and the same augmentation techniques (horizontal flipping with a probability of 0.5 and cropping to size 1024x1024) as it is used in [20].

C. Results on Cityscapes

The Cityscapes dataset is generally used with a subset of 19 classes. We also report experiments with all 34 classes to widen the evaluation of the proposed approach into a more complex problem. The metrics are computed on the validation set of CityScapes, since no public annotations are available for the test set. Evaluation on the validation set is the usual approach on CityScapes (see [20], [22]).

1) Evaluation on 19 classes: Figure 5a reports the results in terms of IoU on the validation set. For completeness, we also report the training and validation losses (see Figures 5b-5c) In Figure 5, we can see that the training loss is almost the same in both cases (curriculum learning vs. standard learning), whereas the validation loss after 30k steps is slightly better using the curriculum learning. Looking at the IoU (Figure 5a), the curriculum learning model performs better than the standard model after 30k steps, similar to what happens with the validation loss. At the end of the training process, our proposed approach reaches a validation IoU equal to 0.57, which is 3.76% higher than the standard model. Each time a new layer is added/activated, there is a small drop in the performance of the curriculum learning approach for a certain number of steps due to the network's adaptation. This behavior ensures a generalizable approach that allows not to get stuck in local minima for more complex optimization problems.



(a) Comparison between learning rate schedulers.

(b) Effect of embedding smoothing.

Fig. 7: CityScapes: 19 classes. Validation IoU. Impact of learning rate scheduler and embedding smoothing. Vertical grey lines represent the insertion of a new layer.

2) Evaluation on all 34 classes: Figure 6a reports the results in terms of IoU on the validation set when considering all the 34 classes in CityScapes. Figures 6b and 6c report the training and validation losses, respectively. In Figure 6, we see similar behavior for training and validation losses as for the 19-class settings. The same statement holds for the validation IoU. Although the validation IoU reaches lower values than before, the trend is the same. Analogous to the other case, the curriculum learning model outperforms the standard model after a certain point. At the end of the training, the validation IoU of the curriculum learning model is 1.90% higher than the standard model. The deterioration in performance due to adding new layers is less obvious.

D. Ablation Study

In this section, we analyze the impact of the cyclic scheduler and the embedding smoothing approach.

Figure 7a compares two schedulers on the 19 classes setting: cyclic vs. polynomial (a common scheduler). Using the more aggressive cyclic scheduler after each insertion leads to an increase of up to 10% in validation IoU. Figure 7b highlights the effect of embedding smoothing. Looking at the validation IoU, we notice large spikes after inserting a layer when embedding smoothing is not applied. This is an undesirable behavior, as it creates a situation of instability. However, even without smoothing, the model recovers quickly thanks to the curriculum learning strategy. Thus, embedding smoothing has a limited impact.

VI. Conclusion and Future Work

This paper proposes an effective approach for training vision transformers from scratch based on curriculum learning. Experiments conducted on a benchmark dataset demonstrate the quality of the proposed method. In future work, we plan to extend the methodology to other models and new datasets for a more comprehensive and general solution.

References

[1] D. Rege Cambrin, L. Colomba, and P. Garza, "CaBuAr: California Burned Areas dataset for delineation," *IEEE Geoscience and Remote Sensing Magazine*, In press.

[2] S. Monaco, N. Bussola, S. Butto, D. Sona, D. Apiletti, G. Jurman, E. Viola, M. Chierici, C. Xinaris, and V. Viola, "Cyst segmentation on kidney tubules by means of u-net deep-learning models," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 2021, pp. 3923–3926.

[3] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of CVPR'16*, 2016.

[4] H. L. Yang, J. Yuan, D. Lunga, M. Laverdiere, A. Rose, and B. Bhaduri, "Building extraction at scale using convolutional neural network: Mapping of the united states," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 11, no. 8, pp. 2600–2614, 2018.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations*, 2015. [Online]. Available: http://arxiv.org/abs/1409.1556

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, 2018.

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *MICCAI'15*. Springer, 2015, pp. 234–241.

[8] A. Kolesnikov, A. Dosovitskiy, D. Weissenborn, G. Heigold, J. Uszkoreit, L. Beyer, M. Minderer, M. Dehghani, N. Houlsby, S. Gelly, T. Unterthiner, and X. Zhai, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021.

[9] Y. Zhang, P. David, and B. Gong, "Curriculum domain adaptation for semantic segmentation of urban scenes," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2039–2049.

[10] D. Dai, C. Sakaridis, S. Hecker, and L. Van Gool, "Curriculum model adaptation with synthetic and real data for semantic foggy scene understanding," *Int. J. Comput. Vision*, vol. 128, no. 5, p. 1182–1204, 2020.

[11] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *CoRR*, 2017. [Online]. Available: http://arxiv.org/abs/1710.10196

[12] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[13] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009, p. 41–48.

[14] E. Chang, H.-S. Yeh, and V. Demberg, "Does the order of training samples matter? improving neural data-to-text generation with curriculum learning," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. ACL, 2021, pp. 727–733.

[15] T. Matiisen, A. Oliver, T. Cohen, and J. Schulman, "Teacher–student curriculum learning," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 9, pp. 3732–3740, 2019.

[16] S. Monaco and D. Apiletti, "Training physics-informed neural networks: One learning to rule them all?" *Results in Engineering*, vol. 18, p. 101023, 2023.

[17] B. Zhang, Y. Wang, W. Hou, H. WU, J. Wang, M. Okumura, and T. Shinozaki, "Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 18 408–18 419.

[18] P. Morerio, J. Cavazza, R. Volpi, R. Vidal, and V. Murino, "Curriculum dropout," 10 2017.

[19] S. Sinha, A. Garg, and H. Larochelle, "Curriculum by smoothing," ser. NIPS'20. Curran Associates Inc., 2020.

[20] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 12 077–12 090.

[21] L. N. Smith, "Cyclical learning rates for training neural networks," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017, pp. 464–472.

[22] J. Jain, J. Li, M. Chiu, A. Hassani, N. Orlov, and H. Shi, "OneFormer: One Transformer to Rule Universal Image Segmentation," 2023.