# Semantic Segmentation trained with learned by Teaching

**Siyuan Zhu**
siz075@ucsd.edu

**Zehua Tang**
zetang@ucsd.edu

**Jiajian Fu**
jif055@ucsd.edu

## 1 Introduction

Semantic segmentation is the process of assigning class labels to each pixel within an image and in this way partitions the image into different segments. Semantic segmentation is an important task in many fields such as computer vision and digital image processing. There are many applications such as autonomous driving, object recognition, medical report analysis and etc.Autonomous driving is a big step of human creation that benefits people for reducing number of traffic accidents as well as providing a convenient environment for human relax during driving vehicle.In recent years, autonomous driving has made great advances to relieve the human burdens on safety driving. To help better understand the surrounding environment of vehicles on roads, image semantic segmentation plays an essential role in that it informs the autonomous driving system what the surrounding objects are and how they are located with each other from the camera view. With aid of precise and information-rich image segmentation, the autonomous driving safety can be more assured. Despite significant breakthrough in semantic segmentation with deep neural networks, such as FCN and CNN, these networks are human designed architectures.However, these architectures have performance that are limited by the architecture design. To solve this problem, researchers proposed neural architectural search (NAS).In addition, researchers have discovered that multi-scale image resolution attention. This leads to models with a large number of parameters.

In this paper, we aims to improve image segmentation performance over existing models by applying an novel training method LBT(Learning by teaching) onto FasterSeg model[7], which is an automatically designed multi-scale semantic segmentation generated by NAS method. In LBT, the teacher model creates a pseudo-labeled dataset and uses it to help train a student model. Based on the performance of student model, the teacher model updates its model weights and generates new pseudo-labeled dataset to teach student model until student model performs well on validation dataset. For the teacher model, we initialize it using the pretrained teacher model `"arch_0"` provided by Chen et al. [7]. Then based on LBT optimization steps, we perform NAS on FasterSeg search space with two initial branches to look for the best architecture and weights for teacher's both backbone and head models. The FasterSeg NAS search space is selected because compared to counterparts, FasterSeg search space is powerful in that it is tailored for real-time segmentation, and multi-resolution branches can be searched and combined; multi-resolution segmentation models have been proven to be effective in the past. For the student model, we will use DeepLabV3 ResNet50 from Pytorch library, which construct a DeepLabV3 model with a Resnet-50 backbone. There are three learning stages in this LBT training framework. In the first stage: the teacher model trains its backbone model weights $W_t$ on its training dataset with its parameter A fixed. A parameter here denotes the architecture of teacher backbone model. In the second stage: the teacher model uses its optimal weights from first stage to generate a pseudo-labeled dataset from a unlabeled dataset and student model learns its weights on this pseudo-labeled dataset and its own training dataset. In the third stage: the teacher model updates its parameter A by minimizing the sum of its validation loss and student model's validation loss on their validation dataset. We will use popular benchmark datasets CityScapes, which contains 34 instance classes from 50 cities of Germany covering spring, summer and fall, for model training and evaluations.

Preprint. Under review.

## 2 Related works

Many efforts have been made to improve the efficiency and performance of image segmentation in recent years. These efforts helps image segmentation through different aspects from better image-level interpretations, such as contextual-awareness methods, to more efficient neural structure designs and training like NAS approaches, etc.

**Relational context method.** Relational context method leverages the relations between pixels or regions to help make predictions on pixel classes for better segmentation understanding. Co-occurrent Feature Model [1] calculates the probability of feature occurrence with current target pixels(object) for prediction with inherent co-occurrence dataset. OCR method [2] structures contextual pixels into object regions and explore the relations between pixel with those regions. People have also tried to utilize non-local blocks, i.e. self attention widely used in natural language processing, in computer vision tasks including semantic segmentation. Lin et al. [3] believes that the attention established based on the dot product model has the characteristics of coupling. Thus authors splits the dot product attention through a disentangled method, the dot product is divided into a whitened pairwise term and a unary term. The disentangled non-local networks introduces non- local network into attention mechanism, and achieved state-of- the-art performances in the semantic segmentation network.

**Edge aware loss optimization.** Boundary loosely method literate using different approaches in order to reduce the boundary awareness loss in order to have a clear boundary shape for segmentation. Edge aware loss [5] assigns weights on those edge pixels which incur a weighted penalty for classifier. Separable convolution[4] using spatial pyramid pooling and encoder-decoder structure combination for faster boundary detection.

**NAS & model training methods** Neural Architecture Search (NAS) methods have shown great potentials in finding efficient scalable network architectures for predicting on dense images. DCNAS [6] designs the first densely connected search space and the first proxyless searching paradigms to eliminate the gap between proxy and real dataset. FasterSeg [7] proposed a novel and broader search space by integrating neural architecture search with multi-resolution branches. self-training[11] could have better performance more flexibility with data augmentation than pre-training on segmentation.

**Attention-based method.** [8] designs an efficient hierarchical multi-scale attention approach which helps with both class confusion and fine detail. Zhang et al. proposed a new backbone network, ResNeSt, to replace the old ResNet backbone. This new method uses a simple architecture to combine the channel-wise attention strategy with multipath network layout. For neural architecture search, this approach augments the search space that can potentially improve neural architecture search performance.

## 3 Experiment setting

With the LBT neural architecture search framework, we need 5 different datasets.

| Notation | Meaning | Specific settings |
|---|---|---|
| A | Architecture of the teacher | FasterSeg architect |
| T | Network weight of the teacher | FasterSeg model |
| S | Network weight of the student | DeepLabV3-RESNET50 |
| $D_t^{(tr)}$ | Teacher training data | Cityscapes training dataset |
| $D_t^{(val)}$ | Teacher validation data | Citscapes validation dataset |
| $D_s^{(tr)}$ | Student training data | Cityscapes training dataset |
| $D_s^{(val)}$ | Student validation data | Citscapes validation dataset |
| $D_u$ | Unlabeled dataset | Citscapes test dataset remove label |
| $D_{pl}$ | Pseudo-dataset | Unlabeled dataset train by teacher model |
| M | model | In Fastseg paper all model are same represent as M |

Table 1: Table to test captions and labels

## 4 Method

**LBT Framework description:** The detailed model training algorithms are as follows: The meanings and specific settings for different parameters are described in Table 1. In the first stage, teacher model finds its optimal weights $T$ with teacher training dataset $D_t^{(tr)}$ with its architecture $A$ fixed:

$$T^*(A) = min_T L(T, A, D_t^{(tr)})$$

$T$ here is a function of teacher architecture $A$.

In the second stage, the teacher model utilizes its best weights $T^*(A)$ to make predictions on unlabeled dataset $D_u$ and generate a pseudo-labeled dataset with its predictions. Then student model learns its optimal weights $T$ by training on the combination of pseudo-labeled dataset $D_u$ from teacher model and student's training dataset $D_s^{(tr)}$:

$$S^*(T^*(A)) = min_S L(S, D_s^{(tr)}) + \lambda L(S, D_{pl}(D_u, T^*(A)))$$

Here optimal student weights $S^*(T^*(A))$ is a function of $T^*(A)$, since teacher model weights is involved in generation of pseudo-labeled dataset $D_s^{(tr)}$.

In the third stage, teacher optimizes its architecture $A$ by minimizing its validation loss on teacher validation dataset $D_t^{(val)}$ and student's validation loss on student validation dataset $D_s^{(val)}$:

$$min_A L(T^*(A), A, D_t^{(val)} + \gamma L(S^*(T^*(A)), D_s^{(val)})$$

Therefore the entire LBT training framework can be summarized as the following iterative steps until convergence:

$$\begin{aligned} & A = min_A L(T^*(A), A, D_t^{(val)}) + \gamma L(S^*(T^*(A)), D_s^{(val)}) \\ & \quad \texttt{s.t.} \quad S^*(T^*(A)) = min_S L(S, D_s^{(tr)}) + \lambda L(S, D_{pl}(D_u, T^*(A)) \\ & \quad \texttt{s.t.} \quad T^*(A) = min_T L(A, T, D_t^{(tr)}) \\ & \quad \lambda \texttt{ here is tradeoff parameter} \end{aligned}$$

Shown below are the detailed update rules for teacher model weights $T$, students model weight $S$ and teacher model architecture $A$:

$$\begin{aligned} T^{'} &= T - \xi_t \nabla_T L(T, A, D_t^{(tr)}) \\ S^{'} &= S - \xi_s \nabla_S (L(S, D_s^{(tr)}) + \lambda L(S, D_{pl}(D_u, T^{'}))) \\ A &\leftarrow A - \eta(\nabla_A L(T^{'}, A, D_t^{(val)} + \lambda \nabla_A L(S^{'}, D_s^{(val)})) \end{aligned}$$

**NAS Algorithm description:** The teacher model perform the architecture search for its weights $T$ and $S$ with respect to a combined loss function over segmentation loss and latency loss on validation dataset $D_t^{(tr)}$, with $\lambda$ as a balancing factor:

$$L = L_{seg}(M) + \lambda Lat(M)$$

$M$ here denote the teacher model(including its weights and architecture). Using this combined loss, the teacher model aims to search for the optimal architecture that not only perform high accuracy on segmentation prediction, but also a low inference latency which enables real-time image segmentation.

During architecture search, the teacher models have weights $T$, $\alpha$, $\beta$, $\gamma$. $\alpha$, $\beta$, $\gamma$ are associated with each operator $O_k \in O$, each predecessor's output $\overline{O}_{l-1}$, and each expansion ratio respectively. These parameter characterize each cell within the teacher model architecture. As described in LBT training steps, in the first step the teacher model updates its weights T by the following gradient:

$$\nabla_T L_{seg}(M|T, \alpha, \beta, \gamma)$$

on training data $D_t^{(tr)}$. The architecture parameters $\alpha$, $\beta$, $\gamma$ are now in the differentiable computation graph, so they also be updated by the following gradient:

$$\nabla_{\alpha,\beta,\gamma} L_{seg} + \lambda \nabla_{\alpha,\beta,\gamma} LAT(M|T, \alpha, \beta, \gamma)$$

The detailed usage for architecture parameters $\alpha$, $\beta$, $\gamma$ are as following: during NAS process, to encourage the search for multiple resolutions, each cell is connected with possibly two predecessors with two different downsmapling rates:

$$\overline{I}_{s,l} = \beta^0_{s,l}\bar{O}_{\frac{s}{2}\to s,l-1} + \beta^1_{s,l}\bar{O}_{s\to s,l-1}$$

Here, $s$ denotes sample rates, l denotes the layer index of cells. And each cell can possibly output to two successors with two downsample rates:

$$\overline{O}_{s\to s,l} = \sum_{k=1}^{|0|} \alpha^k_{s,l} O^k_{s\to s,l}(\bar{I}_{s,l}, X^j_{s,l}, stride = 1)$$

$$\overline{O}_{s\to 2,l} = \sum_{k=1}^{|0|} \alpha^k_{s,l} O^k_{s\to 2s,l}(\bar{I}_{s,l}, X^j_{s,l}, stride = 2)$$

## 5 Discussion

When we are investigating in the task of semantic segmentation by manually designed neural networks, we noticed that the artificial architecture that achieves best performance are the models that have multi-resolution branches. So, we decided that we need a architectural search space that involves multiple resolution branches. And also one common problem for many NAS models is that over time the network tends to evolve into a super complicated network(huge model capacity), which hurts the inference speed of real-time image segmentation.

Therefore, we choose FasterSeg, which not only has a search space that allows free augmentation and flexibility on selecting multi-resolution branches, but also includes a carefully designed accuracy-latency regularization loss that helps alleviate the long latency issue of traditional NAS methods.

Our main work in this paper is that we used the LBT architectural search method to search the search space defined by the FasterSeg model.
The issue with FasterSeg is that their model training and architectural search is separated. The model might converge to a sub optimal weights. And it will take very long for the architecture to converge. We think that the LBT architectural search method is good because it updates the model and architecture every step. In the LBT framework, the teacher can guide the student by labeling unlabeled dataset, and check and upgrade teacher's performance from student's performance.
We also want to investigate on the optimal ratio between the size of student's training dataset and the size of pseudo-labeled dataset generated by teacher model. By trying defferent ratios we hope that we can find the optimal ratio to help further improve the convergence speed of the architecture search.

## 6 Timeline

**Week1-6:** background research and finish project proposal.Write down our algorithms for each stage and models.Assemble the decided models to finish the overall training structure and starting training on decided target dataset.

**Week7-8:** Continue model training and tuning for better performance.

**Week8-10:** Summarize our model performances and finish the project final report.

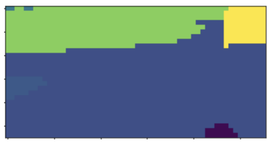
## 7 Experiments

**Data details:** The data set we used in our experiment: CityScapes. The Cityscapes data set contain training date, validation data and testing date, each of those contain 35 classes for semantic segmentation, and each of the data set has same amount of 20k images.In LBT training framework, we divide CityScapes training dataset into smaller trainig dataset for teacher $D^{(tr)}_t$ and student $D^{(tr)}_s$) and validation dataset for teacher $D_t^{(val)}$ and student$D_s^{(val)}$, each of them is half of original CityScapes training dataset. We use CityScapes test dataset as the unlabelled data in LBT model ($D_u$).

**Training steps:** For baseline model, we follow the training steps in FasterSeg paper and we use the model without unroll optimization. and we only perform architecture search using the training _search.py script (without further training the weights), we believe the LBT framework can show its effectiveness in architecture search step only. For both baseline(FasterSeg model) and LBT model(FasterSeg + LBT), we perform architecture search for 30 epochs. For LBT model, we build LBT framework on top of the FasterSeg baseline with unroll optimization. Also for comparison pursposes, we implement LPT(Learning by teaching) onto FasterSeg without unroll optimization.

| Model | IoU accuracy |
|---|---|
| FasterSeg | 20% |
| FasterSeg + LBT | 13% |
| FasterSeg + LPT | 22.5% |

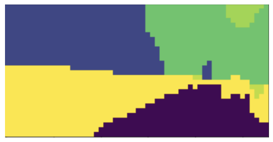
Table 2: Table for model performance comparison


(a) Ground truth 1

(b) Ground truth 2

(c) Ground truth 3

(d) Teacher Prediction 1


(e) Teacher Prediction 2

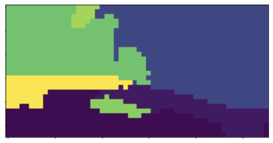

(f) Teacher Prediction 3

Figure 1: Images pairs(each column) for ground truth and predicted map by teacher model

**Results:**

**Analysis:** In this paper, we tested if we use the method of learn by teaching with the FasterSeg neural architecture search space will be better than the method used in the FasterSeg paper. By the experiments that we had, the learn by teaching method doesn't seem to work well with the loss function that was proposed in the FasterSeg paper. I think it's because the FasterSeg paper doesn't use an unrolled model and the latency loss used in their loss function is not differentiable for all the parameters. So, when we are trying to do unrolling for the learn by teaching method. It caused some problems. However, we fixed it by using zero to substitue those gradients. Another problem that we faced during our architectural search is that the problem of gradient explode after about 12 epochs of training. And after that all the parameters becomes not a number in our model. We didn't have much time to fix this and just got the validation results for 12 epochs of searching. After 12 epochs, we got about 0.12 mIoU. I think it's a pretty good result comparing to our FasterSeg results, which is about 0.1 mIoU after 12 epochs.

## 8 Conclusion

In this paper we compare the innovative LBT training framework with the FasterSeg baseline on image semantic segmentation task. In LBT, we used the teacher model for finishing the task and improve itself by teaching student repetitively in order to perform well in the tacvsk. Overall the LBT model have three major stages: teacher learner, student learner which teach by teacher model with those pseudo date that train by teacher model with unable data set, the last stage is teacher model relearn by the result and improve its architecture in order to have better performance and the better result from output task. In fastSeg model, we used its efficient for training image segmentation because it is fast and accuracy.And comparing the result accuracy between original FastSeg, FastSeg with LBT and FastSeg with LPT. We try to show the effectiveness of LBT during architecture search step compared to baseline model, but due to reasons like complicated unroll optimization implementation, training difficulty of too many classes, etc. as described in Analysis part in Experiments section, the model trained with LBT framework couldn't exceed the baseline FasterSeg model. We hope other researchers can take advantage of our work and further prove the effectiveness of LBT framework applied on FasterSeg and other models.

## 9 References

1. Hang Zhang, Han Zhang, Chenguang Wang, and Junyuan Xie. Co-Occurrent Features in Semantic Segmentation. In CVRP,2019.

2. Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-Contextual Representations for Semantic Segmentation. In ECCV, 2020.

3. Minghao Yin, Zhuliang Yao, Yue Cao, Xiu Li, Zheng Zhang, Stephen Lin, and Han Hu. Disentangled Non-Local Neural Networks. In ECCV, 2020.

4. 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.

5.Yuan Jianlong, Zelu Deng, Wang Shu ,and Luo Zhenbo. Multi Receptive Field Network for Semantic Segmentation. In cs.CV, 2020.

6. Xiong Zhang, Hongmin Xu, Hong Mo, Jianchao Tan, Cheng Yang, and Wenqi Ren. DCNAS: Densely Connected Neural Architecture Search for Semantic Image Segmentation. arXiv:2003.11883v1 [cs.CV] 26 Mar 2020.

7. Wuyang Chen, Xinyu Gong, Xian-Ming Liu, Qian Zhang, Yuan Li, and Zhangyang Wang. FasterSeg: Searching for Faster Real-time Semantic Segmentation.

8.Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical Multi-Scale Attention for Semantic Segmentation. arXiv:2005.10821v1 [cs.CV] 21 May 2020.

9.Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R.Manmatha, Mu Li, and Alexander smola.ResNeSt: Split-Attention Networks. arXiv:2004.08955v2[cs.CV]30 Dec 2020

10. Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekini D.Cubuk ,and Quoc V.Le.Rethinking Pre-training and self-training.In NeurIPS, 2020.

11.Parth Sheth ,and Pengtao Xie. Learning by Teaching, with Application to Neural Architecture Search. In IEEE,2020.