Semantic Image Segmentation for Autonomous Driving Scenarios Combining FCNs, DeepLab, and Attention

Peggy (Yuchun) Wang
Stanford University
Stanford, CA
wangyuc@stanford.edu

Khalid Ahmad
Stanford University
Stanford, CA
kahmad@stanford.edu

Abstract

Semantic segmentation is important for autonomous driving scenarios, since it provides accurate road information for the vehicle to use for navigation and planning. Recently, multi-head self-attention, a method widely adopted in the Natural Language Processing community, has been shown to make significant improvements in prediction when augmented with Convolutional Neural Networks. We explore the use of multi-head self-attention to augment popular Convolutional Neural Network architectures, such as FCN and DeepLab, trained on the Mapillary Vistas dataset for street-level semantic segmentation. After 40,000 iterations each, we were able to reach a mean IoU of 21.75% for FCNs, 25.30% for FCNs with attention, 16.05% for DeepLab3+, and a 13.53% for DeepLab3+ with attention. Since we have comparable results and slight improvements in metrics for attention-augmented models, we believe that attention-augmented Convolutional Neural Networks have the potential to improve the accuracy of semantic segmentation. Our results warrants more investigation into the performance of multi-head self-attention after training to convergence on the Vistas dataset.

1. Introduction

The focus of this project is to explore improvements to semantic segmentation methods for autonomous driving scenarios. The problem of semantic segmentation is important because self-driving cars use this information in order to construct an accurate representation of the world around them. For example, it provides information about where the road is, exactly where obstacles and objects on or near the road are located, and lane markings and traffic signs.

The input to our algorithm are street-view images. We then use a Convolutional Neural Network (CNN) to output a predicted segmentation map, consisting of an image of the same number of pixels as the input image. Every pixel in the output image will have a color representing one of the classes in our training dataset.

Our approach is to take methods that have lead to steady advancements in Natural Language Processing (NLP), such as self-attention, and layer them over outputs of layers within CNNs. We then ultimately integrate these attention-augmented layers into state-of-the-art Deep CNN models used for semantic segmentation (semseg).

Self-attention has been widely adopted in the NLP field for its ability to capture long-range interactions and relationships between sequence segments. An example of this is attention's ability to learn the relationship between words in a given sequence and then ensuring that a given translation of that sequence takes into account these long-range interactions, preserving the original sequence's meaning. For this reason, self-attention is believed to have potential for improving the modeling of long-range interactions between image segments which should well complement convolutions' ability to capture localized image segment relationships. Multi-head attention is an even more recent advancement in the NLP field that allows for the training of multiple attention ”heads” which each carry out self-attention over a given feature map. Each head is then able to learn to capture unique interactions between input segments and produces its own unique feature map.

We explore the use of multi-head self-attention to augment popular CNN architectures, such as Fully Convolutional Networks (FCN) and DeepLab, trained on the Mapillary Vistas dataset for street-level semantic segmentation. After training for 40,000 iterations for each model, we reached a mean IoU of 21.75% for FCNs, 25.30% for FCNs with attention, 16.05% for DeepLab3+, and a 13.53% for DeepLab3+ with attention. Based on our results, where models with attention reached comparable accuracy or improved accuracy, we believe that attention-augmented CNNs have the potential to improve the accuracy of semantic segmentation.

2. Related Work

Related work that informed the structuring of this project consist of experiments in three areas of research: exploration of fully convolutional networks for semseg, the application of atrous convolutions to deep convolutional networks, and the augmentation of convolutional networks with attention.

2.1. Fully Convolutional Networks

Fully Convolutional Networks (FCNs) [17] were described in application to semseg in a paper by Long, et al. [13]. Long focused on the removal of linear layers from convolutional networks and replaced them with deeper convolutional layers to produce heat maps as network outputs instead of linear classifications. These heat maps represented a value for each pixel in the input image, its color representing the corresponding pixel's semantic labeling.

FCNs have also proven to be effective in image recognition [19], object detection [8], and medical image segmentation [16], amongst other applications [21][11][22].

For all FCNs' strengths, however, they exhibit weakness that ultimately were corrected for in DeepLab architecures, particularly with the aid atrous convolutions.

2.2. DeepLab and Atrous Convolutions

The state-of-the-art DeepLab model architecture was first introduced in a paper by Chen, et al. [4]. The DeepLab model was a deep convolutional neural network (DCNN) model which employed a newly defined type of convolution, called an atrous convolution, that pushed the DeepLab past the state-of-the-art of the time. An atrous convolution, proposed in Holschneider, et al. [9], allows for a given kernel in a convolution of size k $\times$ k to expand to an arbitrarily large field of few larger than k $\times$ k while still maintaining the number of parameters required for the corresponding filter by replacing the "holes" in the filter with zeros. This method allows for increasing the field of view of a given kernel without the computational penalty, the maintaining of a large feature map for a given input image, and, as demonstrated in [4], a significant performance gain over other semseg models such as FCNs.

Since the first introduction of atrous convolutions into DCNNs, many extensions of the DeepLab model have been produced. The most recent iteration is DeepLab3+, outlined in Chen, et al. [3], which differs largely from the original DeepLab model in that it has been adapted to make use of the advantages of encoder-decoder models. An encoder is employed to capture rich contextual image information, with quicker computation due to elimination of need for feature dilation, while a decoder is used to then recover image object boundaries. The atrous convolutions, which characterize the DeepLab architecture, are incorporated into the encoder segment allowing for preserving of any arbitrary resolution (limited by available computational and memory restraints).

2.3. Attention and Augmented Convolutions

Attention has demonstrated high capability in modeling long-range relationships in diverse applications of neural networks, such as machine translation [20], wave-forming [10], information retrieval [1], and more [6] [24] [23].

The 2017 paper by Vaswani, et al. [20] introduced a particularly powerful attention technique called multi-head attention which allow for the training of multiple attention "heads" which each learn to capture unique interactions between input segments using self-attention, producing unique feature maps for each head. A recent paper by Bello et al. [2] demonstrated the potential performance benefits of utilizing multi-head self-attention in applications of image classification, developing a technique for combining traditional convolutional layers with attention. Their experiments showed that by augmenting powerful CNN models, such as ResNet and SE-ResNet, with multi-head attention layers, performance could be systematically improved for image classification.

Based on these findings, along with attention's clear representational power demonstrated in the field of NLP, we believe that multi-head attention likely is capable of producing the same systematic improvements when added to successful models in semantic segmentation, such as FCN and DeepLab models.

3. Dataset

We utilized the Mapillary Vistas dataset for Semantic Segmentation [14] which consists of 25,000 high-resolution street-level images (at least 1920 $\times$ 1080 resolution) with 65 object categories. Images are taken from a wide range of camera viewpoints, time of day, and weather. Our motivation for choosing this dataset is that Vistas is more diverse in geographic location, contains more classes, and has more finely annotated images than the previous benchmark dataset for semseg of street-level images, Cityscapes [7].

The Vistas dataset was segmented into three portions: training, validation, and test sets. The training set consisted of 18,000 images, the validation set consisted of 2000 images, and the test set consisted of 5000 images. The image resolutions are varied, but all are at least 1920 $\times$ 1080. All images were resampled to 512 $\times$ 1024 resolution for training and validation. We did not perform any data augmentation on the Vistas dataset. Examples of images in the Vistas dataset is shown in Figure 1.

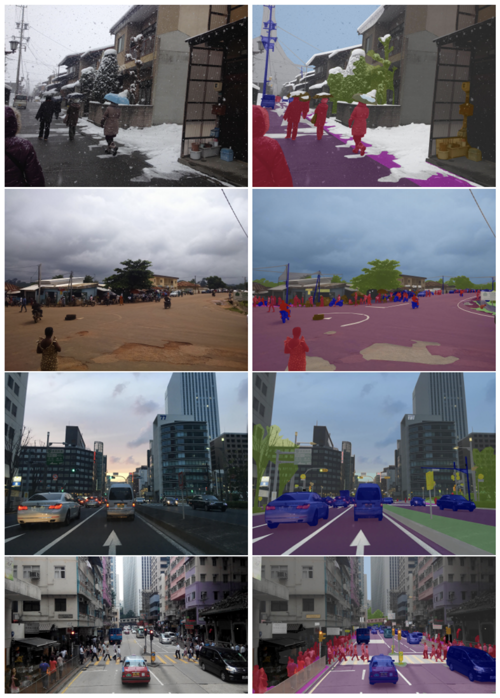


Figure 1: Example of four pairs of original images with corresponding, overlaid color-coded labels in Mapillary Vistas dataset, as outlined in in [14].

4. Methods

The results of Bello, et al.[2] outlined how multi-head attention could be adapted to operate over images, rather than linear sequences, as inputs and built a strong case for its potential value in pushing performance of CNN models in applications outside of image classification. Based on these findings, along with attention's clear representational power demonstrated in the field of NLP, we believe that multi-head attention likely is capable of producing the same systematic improvements when added to successful models in semantic segmentation, such as FCN and DeepLab models.

4.1. Input/Output

For all models, input will be an RGB image of resolution 512 $\times$ 1024, modeled as a tensor of dimensions 512 $\times$ 1024 $\times$ 3, and output will be a segmentation map image of dimensions 512 $\times$ 1024 $\times$ number of classes in training dataset.

4.2. Baseline

Our baseline model consists of a Fully-Convolutional Network (FCN) Implementation as described in [13]. FCN models consist of the same standard structures present in convolutional classification architecture, except that the dense layer which is replaced with an additional series of sequential blocks consisting of only convolutional, pooling, and upsampling layers.

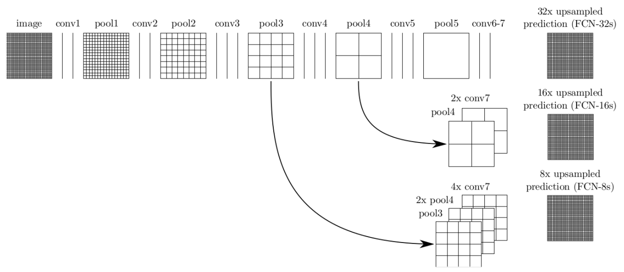


image
conv1
pool1
conv2
pool2
conv3
pool3
conv4
pool4
conv5
pool5
conv6-7
32x upsampled prediction (FCN-32s)
2x conv7
pool4
16x upsampled prediction (FCN-16s)
4x conv7
2x pool4
pool3
8x upsampled prediction (FCN-8s)

Figure 2: FCN Architecture, as outlined in [13]

Figure 2 illustrates the model architecture for FCN32, 16, and 8. The specific FCN architecture used for our baseline model is the FCN8 architecture. This architecture continues from conv7 in a classification architecture, upsamples 2x (with stride 2) and sums the output of conv7, carries out pool4, upsamples 2x (with stride 2) and sums the output of pool4, carries out pool3, and applies a final conv layer (with stride 8) to produce the final output segmentation map.

4.3. Experiments

We trained a total of four models: FCN, FCN with attention, DeepLab3+, and DeepLab3+ with attention. The FCN was used as a baseline. We trained each model to 40,000 iterations, since training to convergence would take more resources in Google Cloud GPU credits and training time that we possessed given the deadline. We used two Nvidia K80 GPUs and trained the models for a week and a half consecutively. The optimizer we used was Stochastic Gradient Descent (SGD) with Momentum. We also used a cross-entropy loss function for all models. Details on the implementation of each model are described below.

4.3.1 DeepLab3+

Our state of the art comparative model consists of an implementation of the DeepLab3+ network as described in Chen, et al [3], the latest refinement of the DeepLab model architecture outlined in [4]. The DeepLab3+ model can be broken into 3 main segments: backbone, atrous spatial pyramid pooling, and decoder.

The backbone and atrous spatial pyramid pooling (ASPP) segments constitute the encoding segment of the DeepLab3+ model.

The backbone used in Chen, et al. [3] is the xception model, as described in [5], modified to include depthwise seperable convolutions with striding instead of max pooling, additional batch normalization and ReLU activate after each 3 $\times$ 3 depthwise convolution, and atrous convolutions

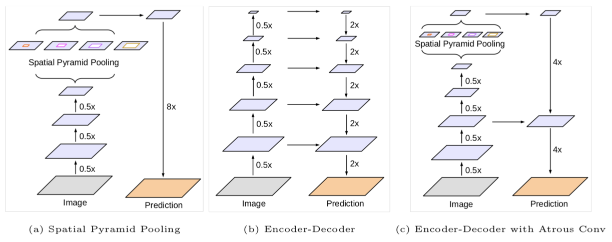


Figure 3: DeepLab3+ Components and Overview, as outlined in [3]

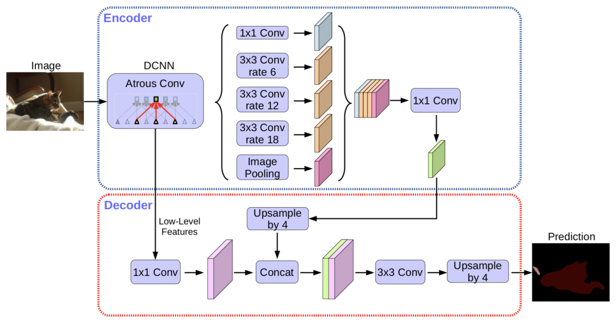


Figure 4: Encoder-Decoder Configuration, as outlined in [3]

at desired states to extract additional feature maps at desired resolutions.

ASPP is a series of atrous convolutions applied on the state output from the encoding bacbone. Atrous convolutions allow for a given kernel in a convolution of size k $\times$ k to expand to an arbitrarily large field of few larger than k $\times$ k while still maintaining the number of parameters required for the corresponding filter by replacing the ”holes” in the filter with zeros. These specialized convolutions are run from minimum to maximum size at the given state, as is illustrated in part a of Figure 3, producing a full feature map of equal size to the state operated over.

The ASPP segment consists of four atrous convolution layers (ASPP1, ASPP2, ASPP3, ASPP4), each with input filters = 2048 and output filters = 256. ASPP1 has a kernel size of 1 and a dilation of 1. ASPP3, -4, and -5 all have kernels of size 3 and have dilations of 6, 12, and 18, respectively. ASPP1, -2, -3, and -4 are all run over X along with globalAvgPooling (an AdaptiveAvgPooling2d pytorch block). Outputs of these 5 blocks are then concatenated together to form a new feature map X_{aspp}. A final convolution, conv1 is then run over X_{aspp} with input filters = 1280, output filters = 256, and kernel size = 1.

Architecture of the DeepLab3+ encoding segment (aligned xception model and ASPP) is illustrated in the top segment of Figure reffig:encoderdecoder, while decoding

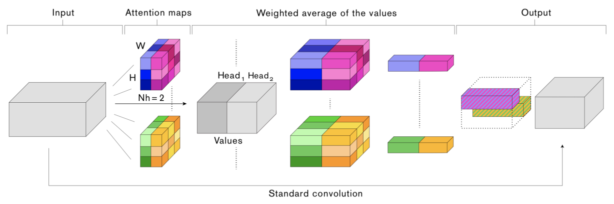


Figure 5: Attention-Augmented Convolutional Block, as outlined in [2]

segment is illustrated in the bottom half.

The low-level features output from the aligned xception backbone, after 1×1 convolution, and the ASPP output, bilinearly upsampled by a factor of 4, are then concatenated within the decoding segment. From there three 3×3 convolution-Batch Normalization-ReLU layers are applied and then again bilinearly upsampled by a factor of 4 to produce the final desired segmentation map.

4.3.2 Attention-Augmented Convolution

For our applications, we implemented a specialized attention-augmented convolution as first introduced in the recent paper by Bello, et al. [2]. That paper outlined an augmented convolution block that consists of both a convolution layer and a multi-head attention layer which are carried out over a given input activation map and combined to produce an output state of shape equal to that of the convolutional layer’s output.

Architecture of the Attention-Augmented Convolutional (AugConv) block is outlined in Figure 5 but will also be formally detailed here. Naming conventions here on out are as follows: H, W, F_{in}, F_{out} refer to activation map height, activation map width, convolution layer input filters, and convolution layer output filters, respectively. N_h, d_v, d_k refer to the number of attention heads, the depth of attention head values, and the depth of both attention head keys and queries, respectively. d_v^h, d_k^h refer to the depth of values and keys/queries, respectively, for a given head h.

Given an input tensor of shape (H, W, F_{in}), we produce a new tensor X of shape $(H \times W, F_{in})$ which allows us to carry out multi-head attention (MHA) over X as detailed in Vaswani, et al. [20]. As outlined in that paper, MHA for head h can be formulated as:

$$O_h = Softmax\Big(\frac{(XW_q)(XW_k)^T}{\sqrt{d_k^h}}\Big)(XW_v) \tag{1}$$

where W_q, $W_k \in \mathbb{R}^{F_in \times d_k^h}$ and $W_v \in \mathbb{R}^{F_in \times d_v^h}$ are learned linear transformations mapping X to queries Q, keys K, and values V for head h.

An additional linear projection is then carried out over all heads, as follows:

$$MHA(X) = XW_O \tag{2}$$

where $O_1, ...O_{N_h}$ are concatenated to form O and $W_O \in \mathbb{R}^{d_v \times d_v}$. Then, MHA(x) is then reshaped from a tensor of shape $(H \times W, d_v)$ into a tensor of shape (H, W, d_v).

From there we then take the output MHA(X) and concatenate it with Conv(X), where Conv(X) is the output of a convolution of input filters F_{in}, output filters F_{out}, and kernel size k, along their third dimension to form MHA+CONV(X). Lastly, a final convolution is run over MHA+CONV(X) with input filters d_v, output filters d_v, and kernel size 1 to produce the output AugConv(X) with shape (H, W, F_{out}).

4.3.3 Attention-Augmented FCN

For our Attention-Augmented FCN (AA FCN) model, AugConv blocks were used to replace all standard convolutional layers in conv7 (defined in Section 4.2), incorporating in attention right before the FCN classification layers. Each convolutional layer in conv7 was replaced with an AugConv layer with identical conv parameters (F_{in}, F_{out} = 512, kernel size = 3), and the following MHA parameters: N_h = 4, d_v = 4, d_k = 40, chosen to match the MHA parameters used in Bello, et a. [2]. All AugConv blocks were made to follow ReLU and Batch Normalization layers, as was done in Bello, et a. [2], to allow, in theory, for the model to learn which features, whether from MHA or from Conv, were of greater value.

Conv7 was chosen as the best location for attention augmentation due to memory restrictions that made attention augmentation in all other purely convolutional blocks infeasible based on testing.

4.3.4 Attention-Augmented DeepLab

For our Attention-Augmented DeepLab (AA DeepLab), AugConv blocks were used to replace all standard convolutional (non-atrous) layers in the ASPP segment of the DeepLab encoder, namely the GlobalAvgPooling block's pooling convolution and conv1 (defined in Section 4.3.1). The GlobalAvgPooling block's pooling convolution was replaced with an AugConv layer with identical conv parameters (F_{in} = 2048, F_{out} = 256, kernel size = 1), and the following MHA parameters: N_h = 4, d_v = 4, d_k = 40, chosen to match the MHA parameters used in Bello, et a. [2]. Conv1 was replaced with an AugConv layer with identical conv parameters (F_{in} = 1280, F_{out} = 256, kernel size = 1), and the following MHA parameters: N_h = 4, d_v = 4, d_k = 40, also chosen to match the MHA parameters used in Bello, et a. [2].

The ASPP encoding segment was chosen as the best location for attention augmentation for two reasons: 1) memory restrictions made attention augmentation in the decoding block infeasible based on testing, and 2) the xception backbone used was pretrained and therefore decided best left untouched due to time constraints.

4.4. Starter Code

We used and edited the FCN implementation in Pytorch from [18], specifically editing the dataloader for the Mapillary Vistas (Vistas) dataset and changing the cross-entropy loss function. We used and edited the DeepLabV3-Plus implementation in Pytorch from [25], again editing the dataloader for the Vistas dataset, changing the cross-entropy loss function and integrating the models into our codebase. We used the Attention-Augmented Convolution block implemented from Bello, et al. [2] as translated to PyTorch by [12].

The FCN codebase was modified to incorporate Attention-Augmented Convolutional blocks as specified in Section 4.3.3. The DeepLab3+ codebase was modified to incorporate Attention-Augmented Convolutional blocks as specified in Section 4.3.4.

5. Results

5.1. Metrics and State of the Art Performance

We use common metrics in semantic segmentation such as overall accuracy, mean accuracy, frequency weighted (FreqW) accuracy, and mean intersection over union (IoU). We use formulations given in Long et al. [13], redefined as follows: let n_{ij} be the number of pixels of class i predicted to belong to class j, where there are n_{cl} different classes, and let $t_i = \sum_j n_{ij}$ be the total number of pixels of class i. We compute:

- Overall accuracy: $\sum_i n_{ii} / \sum_i t_i$
- Mean accuracy: $(1/n_{cl}) \sum_i n_{ii}/t_i$
- Mean IoU: $(1/n_{cl}) \sum_i n_{ii}/(t_i + \sum_j n_{ji} - n_{ii})$
- FreqW accuracy: $(\sum_k t_k)^{-1} \sum_i t_i n_{ii}/(t_i + \sum_j n_{ji} - n_{ii})$

The best performance on the Mapillary Vistas leaderboard across all models is currently a mean IoU metric of 53.37% [1]. Since the Vistas leaderboard uses the mean IoU as a point of comparison, we will hereafter mainly use the mean IoU metric to compare performances between different models.

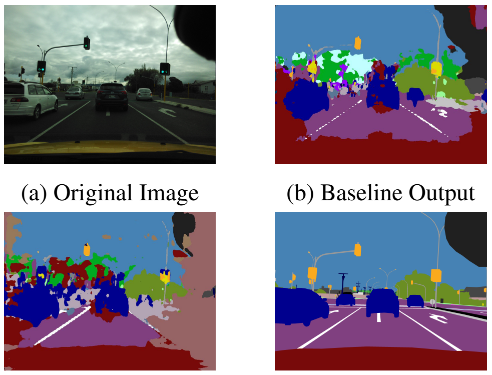
[1] https://blog.mapillary.com/update/2018/01/11/new-benchmarks-for-semantic-segmentation-models.html

FCN Hyperparameters	
Name	Value
Iterations	40,000
Batch Size	4
Validation Interval	1000
Optimizer	SGD w/ Momentum
Learning Rate	1.0e-4
Weight Decay	0.0005
Momentum	0.99
Loss Type	Cross-entropy

Table 1: FCN Hyperparameters Table

FCN Performance		
Metric Name	Baseline	Attention
Overall Acc	0.829990	0.868064
Mean Acc	0.286447	0.304605
FreqW Acc	0.728482	0.780485
Mean IoU	0.217500	0.253031
Final Loss	0.4496	0.6804

Table 2: FCN Performance Table


(a) Original Image (b) Baseline Output

(c) Attention Output (d) Ground Truth

Figure 6: FCN Validation Image Example

5.2. FCN Results

We trained both our baseline FCN model and the attention-augmented FCN model using the FCN8 architecture for 40,000 iterations. Our hyperparameters are described in Table 1. The results for our FCN models after training is shown in Table 2. The metrics are evaluated on our validation set, since the Vistas dataset does not provide ground truth images for the test set. An example of our baseline and attention model output compared with the original image and ground truth semantic labeling in the validation set is shown in Figure 6.

The mean IoU of the FCN baseline was 21.75%, and the mean IoU of the FCN with attention was 25.30%. We can see that our attention model as seen in Table 2 has slightly better prediction accuracy according to all of our metrics. However, we were not able to reach the mean IoU of 53.37% of the best model on the Vistas dataset.

We believe we don't see a significant difference and have not reached the state of the art metric in performance because of training for a relatively low number of iterations and lack of hyperparameter tuning. According to Porzi et al., who used a model similar to DeepLab on the Vistas dataset, their model trained for 192,000 iterations [15]. In comparison, our models were trained for only 40,000 iterations, which is a difference factor of 5. Also, due to time restrictions we chose to utilize the same hyperparameters as used on the Cityscapes dataset, which very well may not have been the best for our models. If both FCN and FCN with attention models were trained to convergence, we believe that FCN with attention would significantly outperform the baseline FCN.

We show a visualization of our predictions in Figure 6 in how our FCN baseline and attention output compares to the ground truth segmentation image.

DeepLab Hyperparameters	
Name	Value
Iterations	40,050
Batch Size	4
Validation Interval	4500
Optimizer	SGD w/ Momentum
Learning Rate	0.01
Weight Decay	0.0005
Momentum	0.9
Loss Type	Cross-entropy
Output Stride	16
Backbone	Xception

Table 3: DeepLab Hyperparameters Table

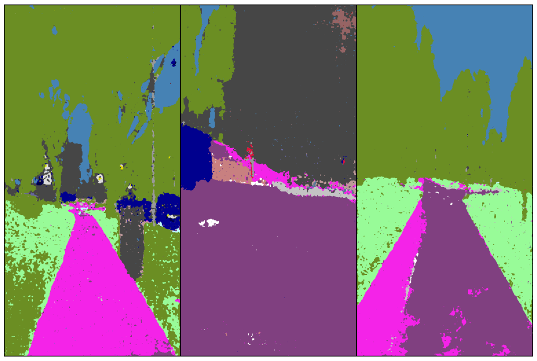
DeepLab Performance		
Metric Name	DeepLab3+	Attention
Overall Acc	0.842424	0.832322
Mean Acc	0.202086	0.169805
FreqW Acc	0.740195	0.724240
Mean IoU	0.160526	0.135338
Final Loss	0.697	0.626

Table 4: DeepLab Performance Table

5.3. DeepLab Results

We trained our DeepLab baseline using the DeepLab3+ architecture for 40,050 iterations. Our hyperparameters are described in Table 3. The preliminary results for DeepLab3+ and DeepLab3+ with attention after training is shown in Table 4. An example of our DeepLab3+ output

(a) Original Image


(b) DeepLab Output

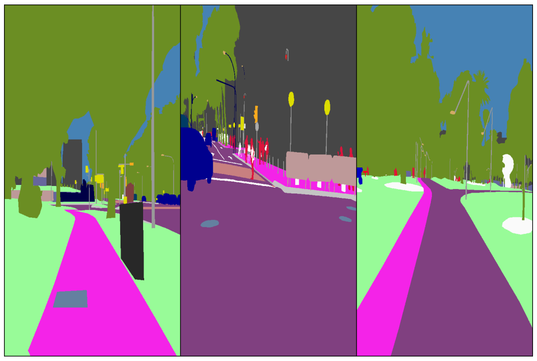
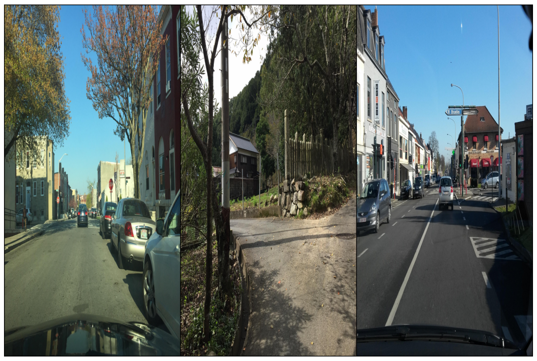
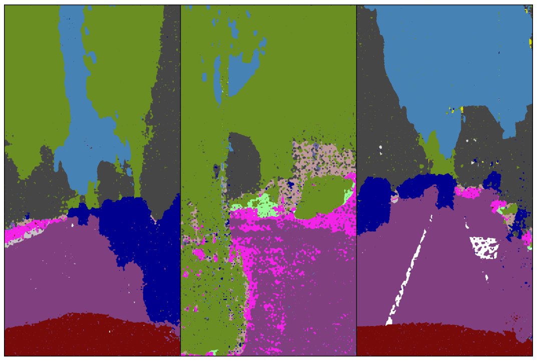

(c) Ground Truth

Figure 7: DeepLab Validation Image Example


(a) Original Image


(b) DeepLab with Attention Output

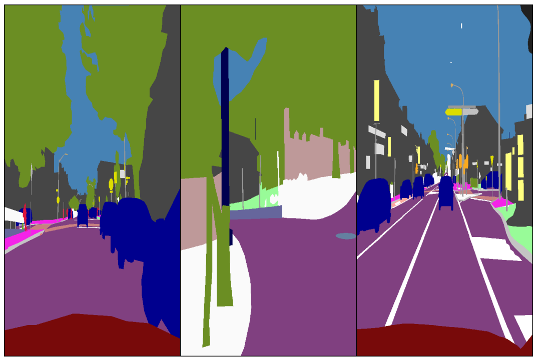

(c) Ground Truth

Figure 8: DeepLab with Attention Validation Image Example

compared with the original image and ground truth semantic labeling in the validation set is shown in Figure 7. An example of our DeepLab3+ with attention model compared with the original image and ground truth semantic labeling in the validation set is shown in Figure 8.

Our DeepLab model failed to outperform our FCN model. The mean IoU for FCN is 21.75% while the mean IoU for DeepLab is 16.05%. This could be due to the low number of iterations trained for as DeepLab may need more iterations to converge and perform at a higher accuracy rate. Due to time restrictions, we chose to also to utilize the same hyperparameters for DeepLab that were used for both FCN models. Lack of hyperparameter tuning could also be responsible for the lower performance of our DeepLab models, particularly with respect to the learning rate. If all four models were trained to convergence, we believe that DeepLab would outperform both FCN models and DeepLab with attention will outperform DeepLab without attention.

DeepLab with attention (with a mean IoU of 13.51%) did not outperform DeepLab (with a mean IoU of 16.06%). However, the accuracy levels are 3% apart and are comparable in terms of performance. This may be due to the low number of training iterations and lack of hyperparameter tuning. We believe if the models were trained to convergence at over 192,000 iterations, as done in Porzi, et al. [15], DeepLab with attention would outperform DeepLab.

6. Conclusion and Future Work

We have found comparable and slight improvements for attention-augmented models. Based on our performance, we believe that attentio-augmented CNNs have the potential to improve the accuracy of semantic segmentation and warrant further investigation into the performance-boost of integrated multi-head attention after training to convergence on the Vistas dataset. We see some promising results for the use of attention-augmented CNNs, however, more work needs to be done to validate their performance. As we were limited on time and resources, we were not able to train all our models to convergence. We believe this to be our most limiting factor in performance and the primary reason why the attention-augmented DeepLab model did not overtake the regular DeepLab model.

We intend to continue with this project in the future. As we were limited on time and resources, we were only able to train for 40,000 iterations. In the future, we will increase the number of training iterations until convergence, which is likely to be at least 192,000 iterations. We also intend to fine-tune hyperparameters, particularly model learning rates. To increase the number of training images, we will also augment images by cropping, reflecting, and distorting images in the Vistas dataset, and will also train with images from the Cityscapes dataset. With this additional work, we are confident that we will be able to show significant results that multi-head attention-augmented models will improve prediction for semantic segmentation over standard models.

Contributions

- Peggy Wang
  - Implemented dataloaders
  - Integrated codebase for DeepLab and FCN
  - Created results tables and predicted figure visualizations
  - Wrote paper
- Khalid Ahmad
  - Implemented attention-augmented convolutions
  - Integrated attention-augmented convolutions into FCN and DeepLab models
  - Set up Google Cloud Virtual Machine
  - Wrote paper

References

[1] I. Bello, S. Kulkarni, S. Jain, C. Boutilier, E. Chi, E. Eban, X. Luo, A. Mackey, and O. Meshi. Seq2slate: Reranking and slate optimization with rnns. *arXiv preprint arXiv:1810.02019*, 2018. 2

[2] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le. Attention augmented convolutional networks. *CoRR*, abs/1904.09925, 2019. 2, 3, 4, 5

[3] L. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *CoRR*, abs/1802.02611, 2018. 2, 3, 4

[4] 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 transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2018. 2, 3

[5] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 3

[6] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio. Attention-based models for speech recognition. In *Advances in neural information processing systems*, pages 577–585, 2015. 2

[7] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 2

[8] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016. 2

[9] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. *Wavelets, Time-Frequency Methods and Phase Space*, -1:286, 01 1989. 2

[10] C. A. Huang, A. Vaswani, J. Uszkoreit, N. Shazeer, C. Hawthorne, A. M. Dai, M. D. Hoffman, and D. Eck. An improved relative self-attention mechanism for transformer with application to music generation. *CoRR*, abs/1809.04281, 2018. 2

[11] J. Johnson, A. Karpathy, and L. Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4565–4574, 2016. 2

[12] M. Kim. Implementing attention augmented convolutional networks using pytorch. *https://github.com/leaderj1001/Attention-Augmented-Conv2d*, 2019. 5

[13] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 2, 3, 5

[14] G. Neuhold, T. Ollmann, S. Rota Bulo, and P. Kontschieder. The mapillary vistas dataset for semantic understanding of

street scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4990–4999, 2017. 2, 3

[15] L. Porzi, S. R. Bulò, A. Colovic, and P. Kontschieder. Seamless scene segmentation. *CoRR*, abs/1905.01220, 2019. 6, 8

[16] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2

[17] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 2

[18] M. P. Shah. Semantic segmentation architectures implemented in pytorch. *https://github.com/meetshah1995/pytorch-semseg*, 2017. 5

[19] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. 2, 4

[21] L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 3119–3127, 2015. 2

[22] L. Wang, L. Wang, H. Lu, P. Zhang, and X. Ruan. Saliency detection with recurrent fully convolutional networks. In *European conference on computer vision*, pages 825–841. Springer, 2016. 2

[23] X. Wang, G. Xu, J. Zhang, X. Sun, L. Wang, and T. Huang. Syntax-directed hybrid attention network for aspect-level sentiment analysis. *IEEE Access*, 7:5014–5025, 2019. 2

[24] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 21–29, 2016. 2

[25] J. Zhang. Deeplab v3+ in pytorch. *https://github.com/jfzhang95/pytorch-deeplab-xception*, 2018. 5