CS F425 – DEEP LEARNING PROJECT ON

STRUCTURE GUIDED LANE DETECTION

FINAL REPORT

GROUP MEMBERS

SASWATA MUKHERJEE	2019A4PS0162G
SUHAAS MAHAJAN	2019AAPS0315G
ARNAV JAIN	2019A7PS0158G

INDEX

ACKNOWLEDGEMENTS

We would like to express our heartfelt gratitude to Professor Tirtharaj Dash and Professor Tanmay Tulsidas Verlekar for giving us the opportunity to work on this exciting project topic by making CS F425 DEEP LEARNING a project-oriented course. The hands-on working experience we got by carrying out this project helped us to grasp the concepts related to CNNs in a better way and understand the topic better. We would also like to thank the teaching assistants of this course for their constant efforts and inputs that helped us to make this effort fruitful within the stipulated period of time.

GitHub Link:

https://github.com/Saswata13/PAPER---STRUCTURE-GUIDED-LANE-DETECTION

LANE DETECTION: AN OVERVIEW

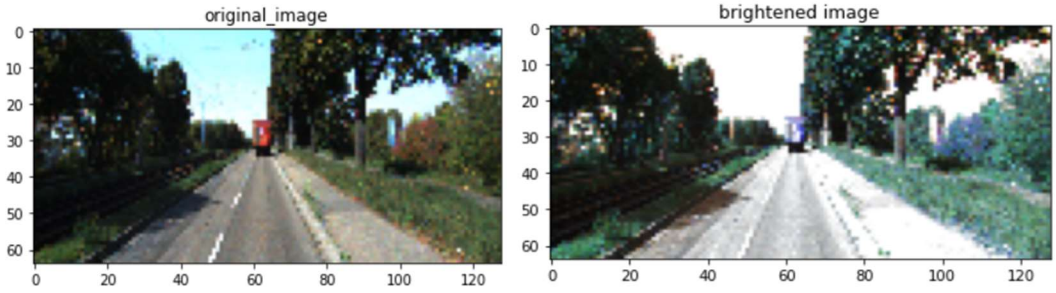
The paper we are trying to implement is based on structure-guided lane detection for autonomous vehicles. The original datasets used by the authors are the **CU Lane** and the **TuSimple** datasets. Both these datasets are very big and computationally expensive to work on. For our purposes we will be using the **KITTI Road dataset** published by Karlsruhe Institute of Technology. The link to the original dataset is: http://www.cvlibs.net/datasets/kitti/eval_road.php

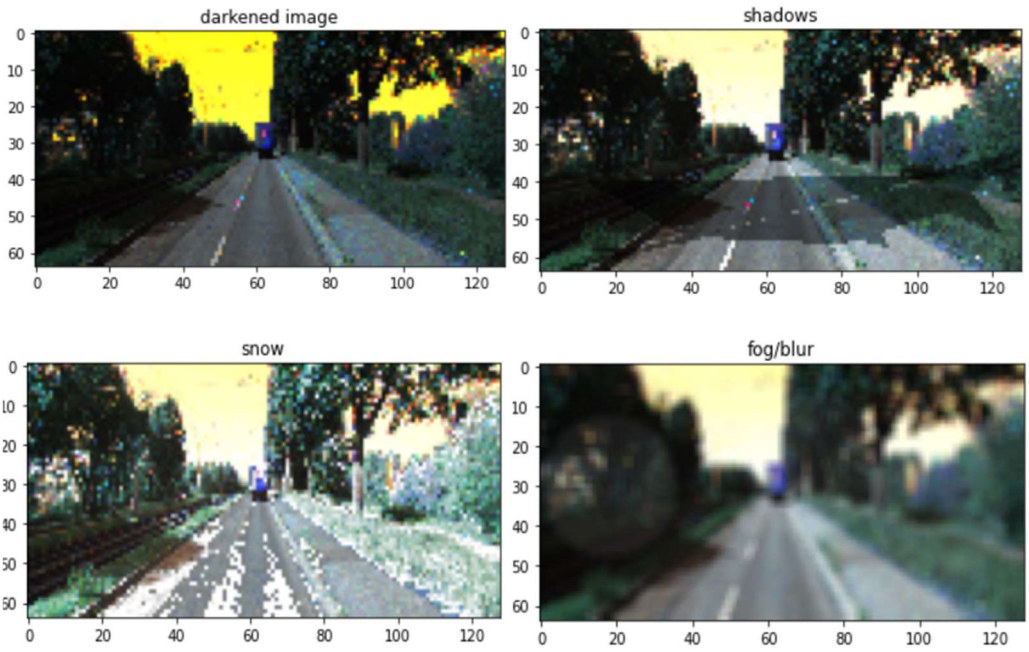
We have used a smaller version of this dataset from the link: https://www.kaggle.com/datasets/sumanyughoshal/kitti-road-dataset.
The paper we are trying to implement can be found at: https://www.ijcai.org/proceedings/2021/0138.pdf

Lane detection is a very common and important segmentation task for autonomous vehicles. However, the common challenges that are still prevalent in this field are:

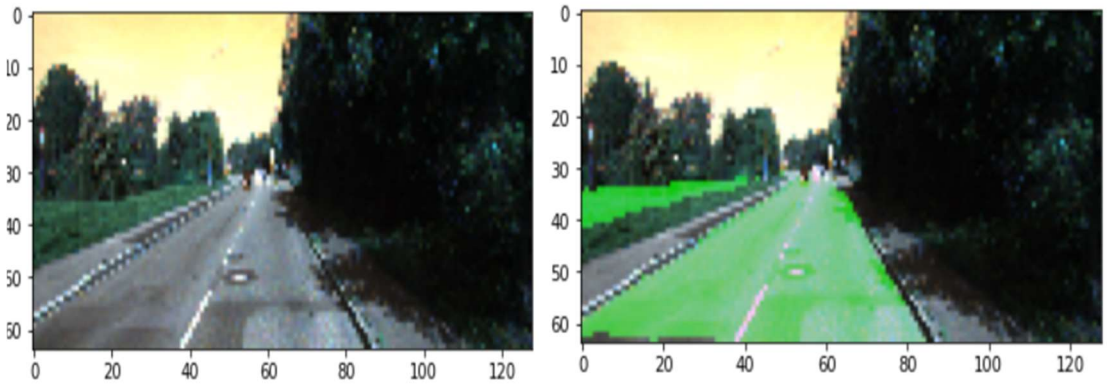
- Characterizing lanes based on different methods of annotations.
- Identification and modelling of relationship between surrounding scene and lanes.
- Supporting more attributes of the scene to come up with better results while detecting lanes.

To address these difficulties, a uniform structure-guided framework – SGNet has been proposed in which a new lane representation has been introduced. A top-down vanishing point guided mechanism is then used to achieve the final aim. In the midsemester submission, we had devoted our efforts towards understanding the nature of our dataset and trying to implement some creative augmentations. We had used the Automold library to generate image augmentations pertaining to real life scenarios an example of which is shown below:



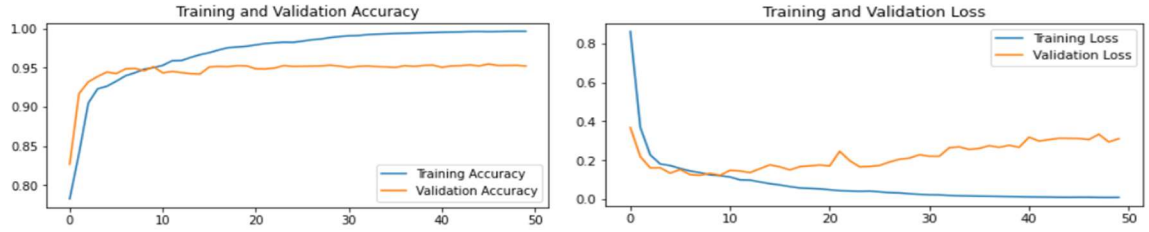


After aggregating the existing dataset with the augmented images, we had fitted the data into a simple U-Net architecture to get results on unseen images like these:



Input Image (Unseen) *Output*

It is to be noted that our testing dataset **does not have any masks to validate** the accuracy but looking at the output, we could identify (at least qualitatively) the extent of correctness of the model's output. However, for our midsemester submission, we had split our training set into training and validation to get the following results on accuracy and loss:



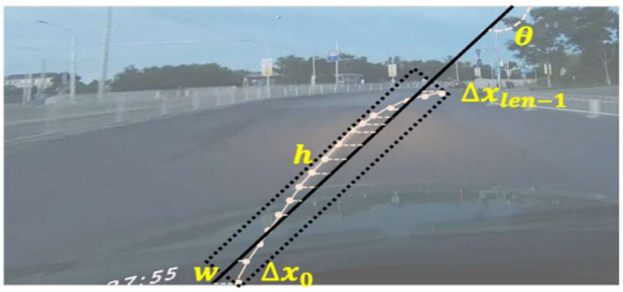
From these plots we had concluded that 50 epochs for training was much greater than the optimal number of epochs required to train. We had arrived at the conclusion that 20 epochs were good enough to train the simple U-Net on our data.

In our final submission, we have tried to build on that and create a comprehensive framework replacing the simple U-Net to perform lane detection. The fundamental difference in the type of dataset we are using and the ones used in the paper has led us to approach a few steps differently, but we have tried to base our framework on the paper to the extent possible. The details of our approach, implementation, frameworks that we have used, results that we have obtained have been explained in the following sections of this report.

LANE REPRESENTATION

For implementing the SGNet framework; a box-line based method is used to represent the lane boundaries. This approach works well for this framework because the annotation style for the CULane and TuSimple datasets is such that only the lane boundaries have been highlighted in the ground truth mask. However, in the dataset which we are using (KITTI Road), the masks are such that the entire lane area has been highlighted.

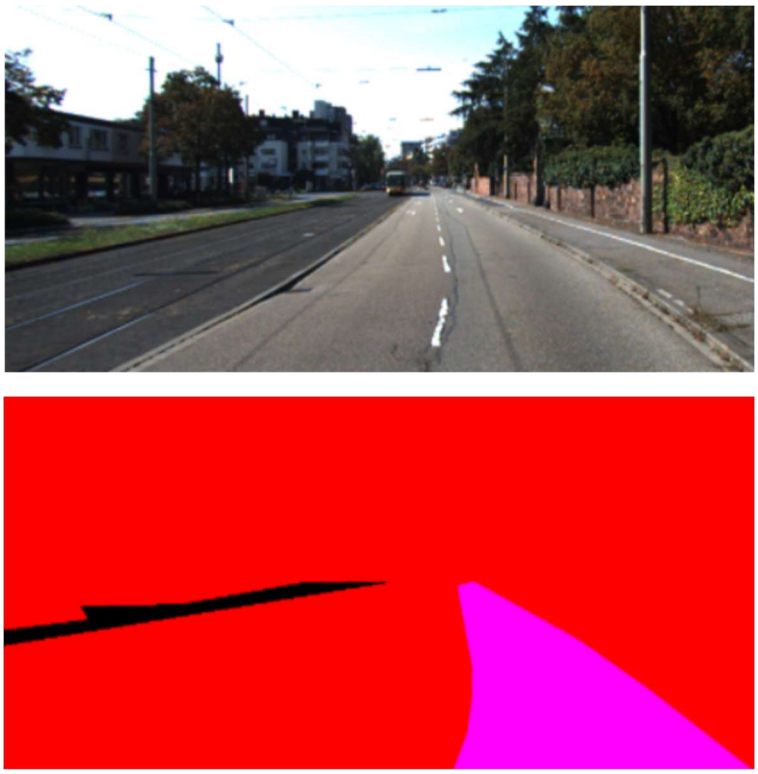
Our intention to use a dataset having this kind of an annotation style is to carry out lane detection in a differently annotated data. To represent lanes by a box-line-based approach in this kind of data would be tedious and computationally heavy. Since our dataset already has image annotations representing the entire road area to be traversed, we have used these masks directly and have by-passed the box-line based annotation approach.



Box-line based annotation approach used in the SGNet paper

The fundamental difference in annotation styles of the two datasets has been illustrated below with the help of examples. The first pair of pictures show the true image and ground-truth mask of an example from the CU Lane dataset and the next pair of images show the true image and its corresponding masks of an example from the KITTI Road dataset. The stark contrast is easily recognizable.

True image and ground-truth mask of an example from the CU Lane dataset

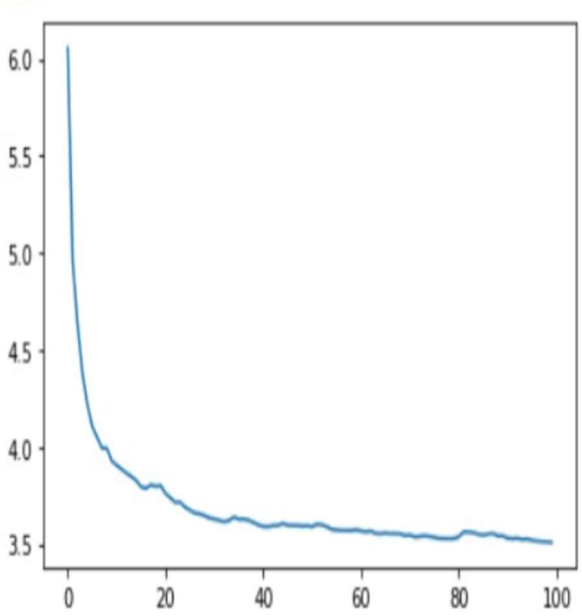


True image and ground-truth mask of an example from the KITTI Road dataset

FEATURE EXTRACTION

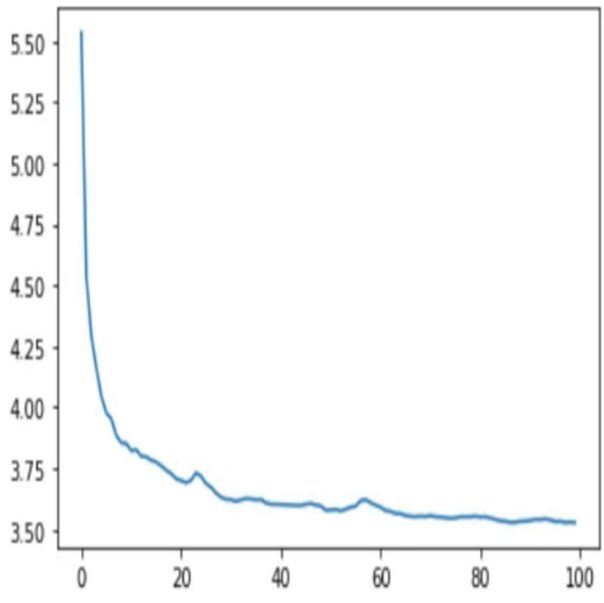
The SGNet framework uses ResNet as a feature extractor, which is modified in a way to remove the last global pooling and fully connected layers for the pixel level prediction task. The feature extractor has five residual blocks for encoding. For achieving larger feature maps, the last residual module is convolved with a convolutional layer of 256 kernels of size 3 x 3 followed by up-sampling (x2) for the decoder block.

We have used a very similar feature-extractor. Our feature extractor consists of five encoder blocks which is followed by a central block formed by convolving the last encoder layer with a convolutional layer of 256 kernels of size 3 x 3. This is followed by a decoder block for which we have used the Conv2DTranspose operation (as an experiment) instead of the Up-sampling operation implemented in the paper. The comparative loss function curves with respect to number of epochs are shown below:



CONV-2D TRANSPOSE

LOSS



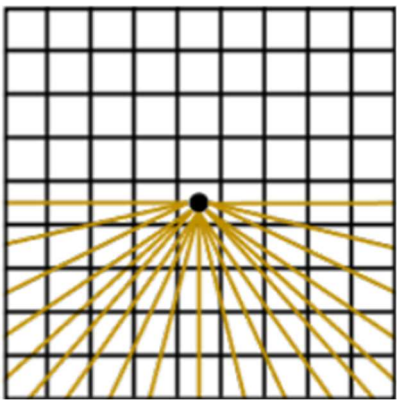
UP-SAMPLING

LOSS

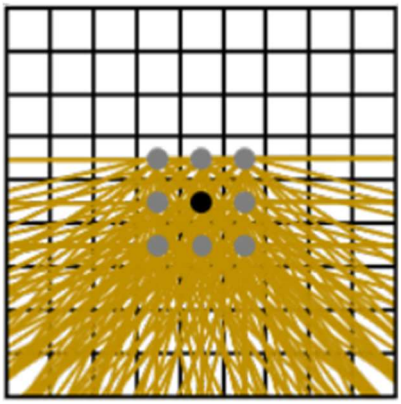
For our purpose, we have used the Conv-2D Transpose operation for creating the decoder block; though there is no significant difference between the variation of the loss function for both the operations, the Conv-2D Transpose shows a slightly smoother nature due to which we choose to go ahead with it.

STRUCTURE GUIDED ANCHORING

The paper has implemented a vanishing point guided anchoring mechanism to learn the lane representation which is based on the box-line based approach. The essence of this approach is to learn the box-line parameters keeping the vanishing point as a reference. The vanishing point (VP) provides strong characterization of geometric scene, representing the end of the road and also the “virtual” point at which the lane boundaries intersect after a long distance.



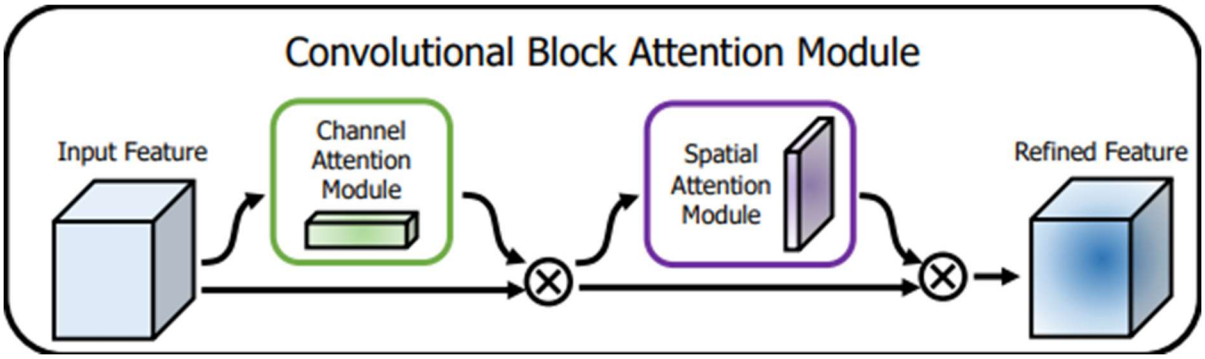
The vanishing point



Vanishing point guided anchoring mechanism where the gray points are the anchors

Since VP is the intersection point of lanes, lanes in the scene must pass through VPs, and lines that do not pass through the VPs are not lanes in the scene with high probability. However, since our dataset uses a different form of annotation (as mentioned earlier), we have not used this approach. Instead, we have tried to implement an attention-based mechanism, the details of which are mentioned in the following sections.

CONVOLUTIONAL BLOCK ATTENTION MODULE



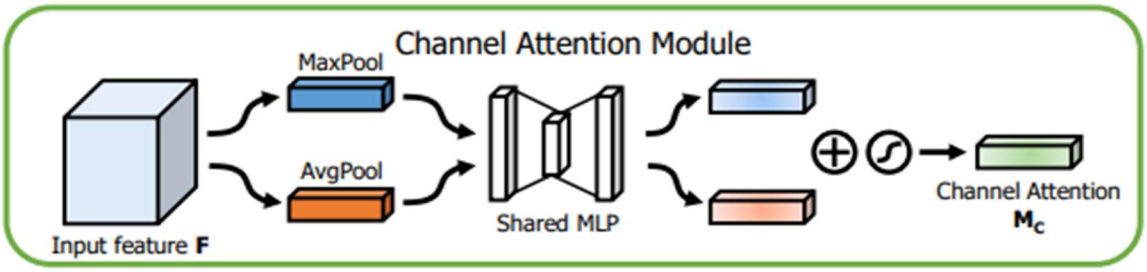
The convolutional block attention module (CBAM) helps to increase the representation power of standard CNNs. It focuses on important features and suppresses the unnecessary ones. Since convolution operations extract informative features by blending cross-channel and spatial information together, we adopt two blocks to emphasize meaningful features along those two principal dimensions: channel and spatial axes. To achieve this, we sequentially apply channel and spatial attention modules, so that each of the branches can learn 'what' and 'where' to attend in the channel and spatial axes respectively. As a result, our module efficiently helps the information flow within the network by learning which information to emphasize or suppress.

Given an intermediate feature map F ∈ R C×H×W as input, CBAM sequentially infers a 1D channel attention map Mc ∈ R C×1×1 and a 2D spatial attention map Ms ∈ R 1×H×W. The overall attention process can be summarized as:

$$\mathbf{F}' = \mathbf{M_c}(\mathbf{F}) \otimes \mathbf{F},$$
$$\mathbf{F}'' = \mathbf{M_s}(\mathbf{F}') \otimes \mathbf{F}',$$

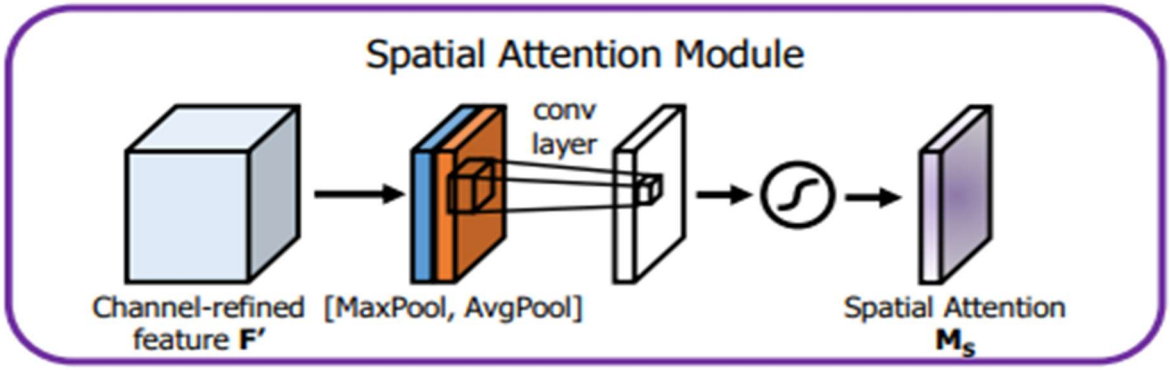
A detailed explanation of the channel and spatial attention blocks has been discussed below.

CHANNEL ATTENTION MODULE



A channel attention map is produced by exploiting the inter-channel relationship of features. As each channel of a feature map is considered as a feature detector, channel attention focuses on 'what' is meaningful given an input image. To compute the channel attention efficiently, we squeeze the spatial dimension of the input feature map. For aggregating spatial information, average-pooling had been commonly adopted before. Historically it has been suggested to use it to learn the extent of the target object effectively and apply it in their attention module to compute spatial statistics. However, the paper on CBAM demonstrates that max-pooling gathers important clues about distinctive object features to infer finer channel-wise attention. Therefore, we use both average-pooled and max-pooled features simultaneously. It has been confirmed experimentally (in the same CBAM paper) that exploiting both features greatly improve the representation power of networks rather than using each independently.

SPATIAL ATTENTION MODULE



A spatial attention map is created by exploiting the inter-spatial relationship of features. Different from the channel attention, the spatial attention focuses on 'where' as an informative part, which is complementary to the channel attention. To compute the spatial attention, average-pooling and max-pooling operations are done along the channel axis and then concatenated to generate an efficient feature descriptor. The pooling operations along the channel axis is effective in highlighting the informative regions. On the concatenated feature descriptor, convolution layer is applied to generate a spatial attention map which encodes where to emphasize or suppress. The detailed mathematical expression is shown below. Channel information of a feature map is aggregated by using two pooling operations, generating two 2D maps. Each map denotes average-pooled features and max-pooled features across the channel. Those are then concatenated and convolved by a standard convolution layer, producing our 2D spatial attention map. In short, the spatial attention is computed as:

$$\begin{aligned}\mathbf{M_s}(\mathbf{F}) &= \sigma(f^{7\times7}([AvgPool(\mathbf{F}); MaxPool(\mathbf{F})])) \\ &= \sigma(f^{7\times7}([\mathbf{F^s_{avg}}; \mathbf{F^s_{max}}])),\end{aligned}$$

The application of these two attention modules can be seen as an alternative to the vanishing point guided anchoring mechanism implemented in the paper. The purpose of the vanishing point guided mechanism is to create anchors around the vanishing point to learn the exact (or approximate) boundaries of the actual lanes present in the scene. The attention blocks used in our project fulfil the same purpose in a slightly different way. As shown above, the channel attention block captures 'what' is important in the input scene and the spatial attention block captures 'where' does the important part lie, thus comprehensively capturing the relevant image entities as well as their locations to detect the area occupied by the lanes effectively.

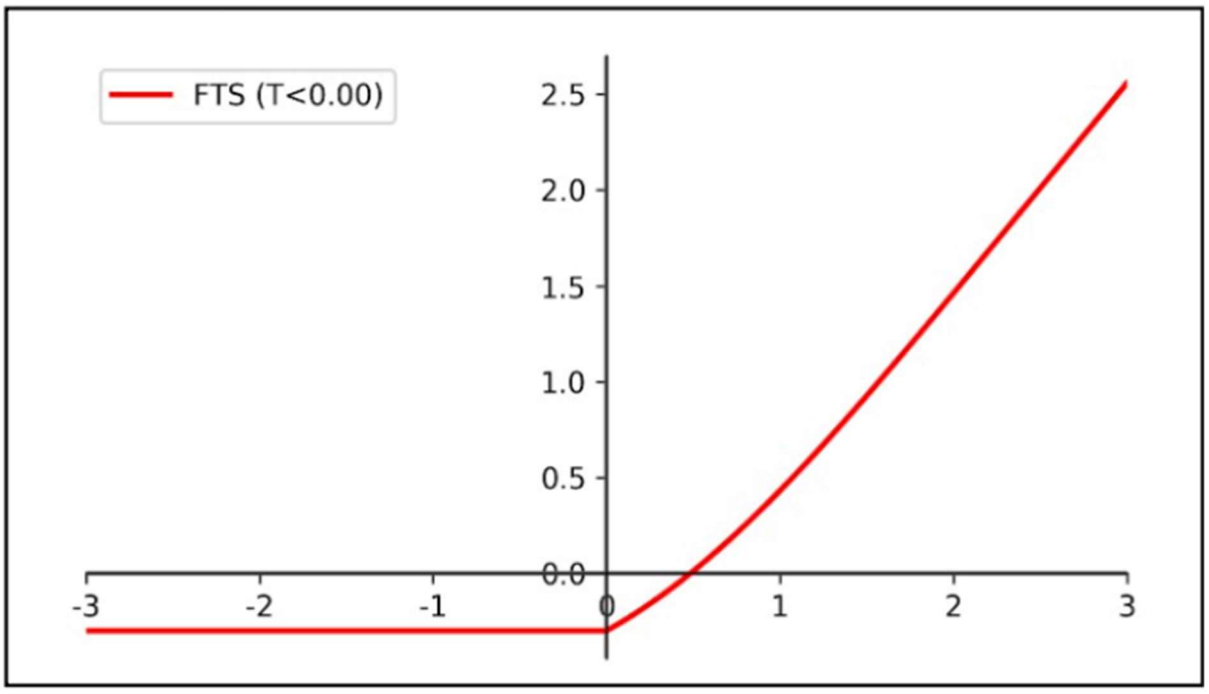
OUR CONTRIBUTIONS

USING FTSWISH+: A FAIRLY NEW ACTIVATION FUNCTION

FTSwishPlus stands for Fixed Threshold Swish with mean shifting. It is a fairly new activation function in the field of deep learning, which is gaining immense popularity owing to drawbacks of ReLu (removal of all negative values and the associated dying gradient).

FTSwish provides a threshold (a fixed negative value) for all the negative inputs and a swish like curve for the positive values.

In Mean Shifting, The core idea is to adjust the starting point in such a manner that it adjusts the activation function to give Standard Normal distribution values of standard deviation and mean, 1 and 0 respectively.



In this case, for FTSwish, it requires an adjustment of -0.1 to get a mean of zero. This adjustment is then applied every time the FTSwish activation is used.

We have used this activation function in multiple classes in our code in the spirit of experimenting as an alternative to the popular ReLU. The results obtained (output images at the end) are quite impressive.

APPLICATION OF THE AUTOMOLD LIBRARY FOR AUGMENTATION

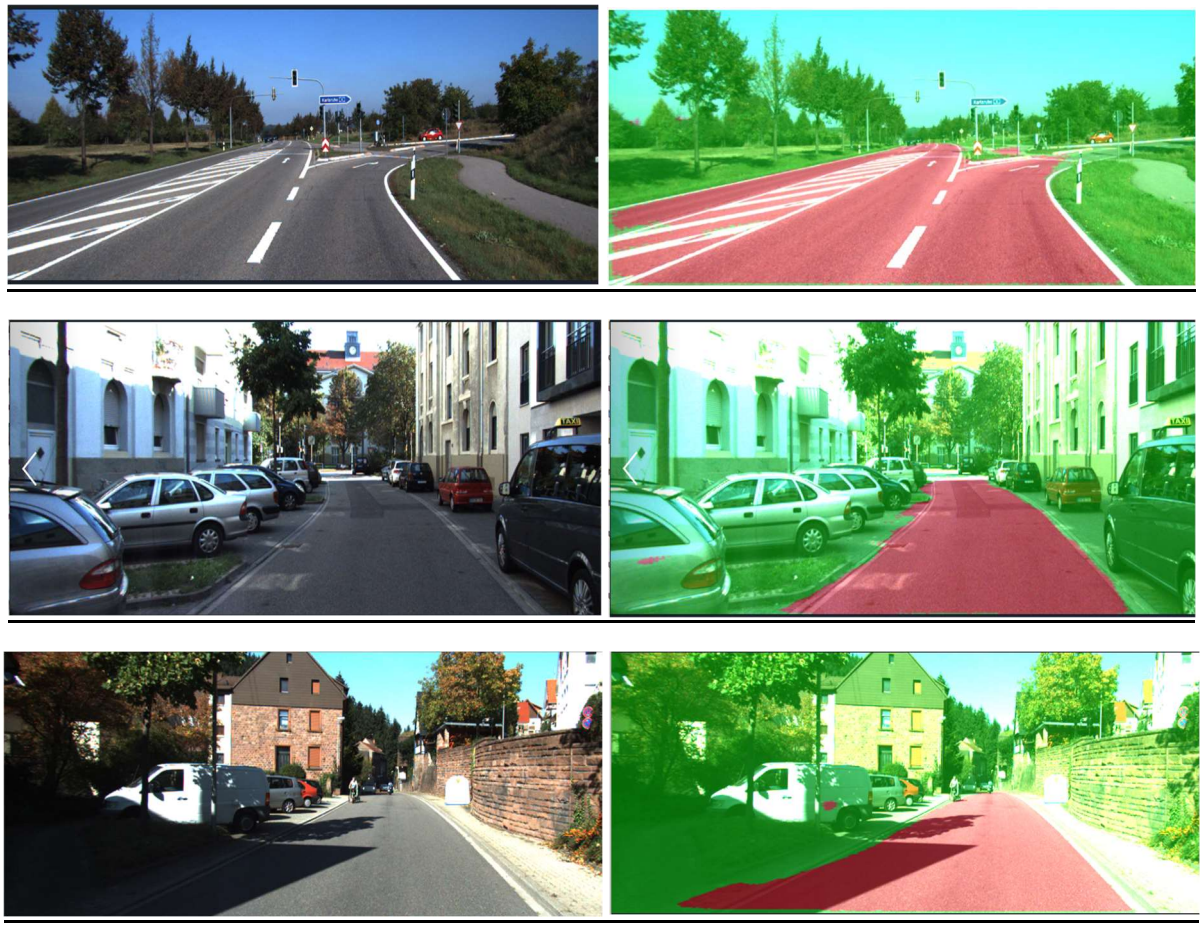
We have used the Automold library to create augmented images having real life image scenarios: dark, sunny, rainy, snowy, shadows, fog/blur. The image augmentations have been displayed at the beginning of this report.

APPLICATION OF ATTENTION BLOCKS

We have used two attention blocks: channel and spatial attention for learning our lane representation more effectively. This serves as an alternative to the vanishing point guided anchoring mechanism and is quite suited to road datasets annotated in the way similar to the KITTI Road dataset.

RESULTS

Test input images (unseen) *Output images obtained*



CONCLUSION AND FUTURE WORK

We have successfully implemented our own framework based on a structure guided lane detection framework SG-Net. Although our framework is a bit different in that ours is more of an attention guided framework as compared to the structure guided framework proposed in the paper. It should be noted that the feature extractor used by us is similar to the one used in the paper and hence it can be concluded that the backbone of our framework is based on the SG-Net framework although the lane representations and the procedure to learn them are a bit different. Keeping an eye on the future prospects of this project, we can look to implement a solution to further dissect multiple lanes (as was done in the original SG-Net paper). This is currently a limitation of our project; it fails to distinguish multiple lines. This is something that we would like to work on in the future.

REFERENCES

- Su, J., Chen, C., Zhang, K., Luo, J., Wei, X., & Wei, X. (2021). Structure guided lane detection. *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*. https://doi.org/10.24963/ijcai.2021/138

- Woo, S., Park, J., Lee, J.-Y., & Kweon, I. S. (2018). CBAM: Convolutional Block Attention Module. *Computer Vision – ECCV 2018*, 3–19. https://doi.org/10.1007/978-3-030-01234-2_1

- Wright, L. (2019, April 25). *Comparison of activation functions for deep learning. initial winner = ftswish+*. Medium. Retrieved May 17, 2022, from https://lessw.medium.com/comparison-of-activation-functions-for-deep-learning-initial-winner-ftswish-f13e2621847