6. Appendix

6.1. Full Set of Rules

Predicates:

- INTERSECTIONUNDER(object, class, image) - Known predicate (observation). Describes if the area under the object is of a particular class in the image. The classes are 'r' - road, 'pw' - pedestrian walk, 'gr' - ground. During rule weight optimization obtained from ground truth semantic segmentation, during inference obtained from the predictions of a network.
- INTERSECTIONBEHIND(object, class, image) - Known predicate (observation). Describes if the area behind the object is of particular class in the image. The class is 'bckgr' - combined buildings and nature. During rule weight optimization obtained from ground truth semantic segmentation, during inference obtained from the predictions of a network.
- DISTANCE(object1, object2, image) - Known predicate (observation). Describes if two objects are close to each other in the image. Inferred from the RGB input and corresponding depth map.
- HASCOLOR(object, color, image) - Known predicate (observation). Describes if the object has the color in it. The colors are 'white', 'blue', 'yellow', 'red'. Inferred from the RGB input.
- ISTYPE(object, class, image) - Unknown predicate (target). Defines the class of the object. Class can be one of 'ped' (pedestrian), 'tr_sign' (traffic sign), 'tr_signal' (traffic signal)

```
0.768: INTERSECTIONUNDER(O1, 'r', I) >> ISTYPE(O1, 'ped', I) ^2

0.957: INTERSECTIONUNDER(O1, 'pw', I) >> ISTYPE(O1, 'ped', I) ^2

0.789: INTERSECTIONUNDER(O1, 'gr', I) >> ISTYPE(O1, 'ped', I) ^2

3.2E-5: INTERSECTIONBEHIND(O1, 'bckgr', I) >> ISTYPE(O1, 'ped', I) ^2

0.234: ~INTERSECTIONBEHIND(O1, 'bckgr', I) & ~INTERSECTIONUNDER(O1, 'pw', I)
& ~INTERSECTIONUNDER(O1, 'gr', I) & ~INTERSECTIONUNDER(O1, 'r', I)
>> ~ISTYPE(O1, 'ped', I) ^2

0.372: (O1 % O2) & DISTANCE(O1, O2, I) & ISTYPE(O2, 'ped', I)
>> ISTYPE(O1, 'ped', I) ^2

0.278: (O1 % O2) & DISTANCE(O1, O2, I) & ISTYPE(O2, 'ped', I) & DISTANCE(O2, O3, I)
& ISTYPE(O3, 'tr_sign', I) >> ISTYPE(O1, 'ped', I) ^2

0.002: (O1 % O2) & DISTANCE(O1, O2, I) & ISTYPE(O2, 'ped', I)
& ISTYPE(O4, 'tr_signal', I) & DISTANCE(O2, O4, I) >> ISTYPE(O1, 'ped', I) ^2

0.899: ISTYPE(O3, 'tr_signal', I) & DISTANCE(O1, O3, I) & INTERSECTIONUNDER(O1, 'r', I)
>> ISTYPE(O1, 'ped', I) ^2

0.246: ISTYPE(O3, 'tr_signal', I) & DISTANCE(O1, O3, I) & INTERSECTIONUNDER(O1, 'pw', I)
>> ISTYPE(O1, 'ped', I) ^2

0.512: INTERSECTIONUNDER(O1, 'gr', I) & ISTYPE(O3, 'tr_signal', I) & DISTANCE(O1, O3, I)
>> ISTYPE(O1, 'ped', I) ^2

0.213: ISTYPE(O3, 'tr_signal', I) & DISTANCE(O1, O3, I)
& INTERSECTIONBEHIND(O1, 'bckgr', I) >> ISTYPE(O1, 'ped', I) ^2

1.4E-4: DISTANCE(O1, O3, I) & ISTYPE(O3, 'tr_sign', I) >> ISTYPE(O1, 'ped', I) ^2
```

0.317: ISTYPE(O4, 'tr_signal', I) & DISTANCE(O1, O4, I) >> ISTYPE(O1, 'ped', I) ^2

0.542: HASCOLOR(O3, 'white', I) & HASCOLOR(O3, 'blue', I) >> ISTYPE(O3, 'tr_sign', I) ^2

0.561: HASCOLOR(O3, 'yellow', I) >> ISTYPE(O3, 'tr_sign', I) ^2

0.841: HASCOLOR(O3, 'white', I) & HASCOLOR(O3, 'blue', I) >> ISTYPE(O3, 'tr_sign', I) ^2

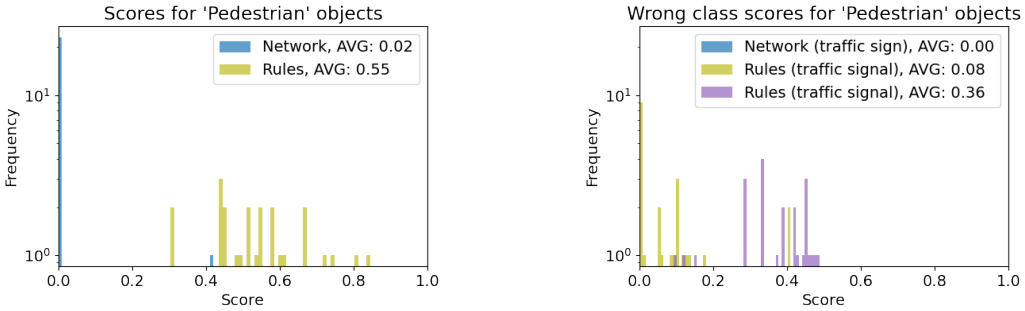
0.181: HASCOLOR(O3, 'red', I) & HASCOLOR(O3, 'white', I) >> ISTYPE(O3, 'tr_sign', I) ^2

0.709: ~HASCOLOR(O3, 'white', I) & ~HASCOLOR(O3, 'blue', I)
& ~HASCOLOR(O3, 'yellow', I) & ~HASCOLOR(O3, 'red', I) >> ~ISTYPE(O3, 'tr_sign', I) ^2

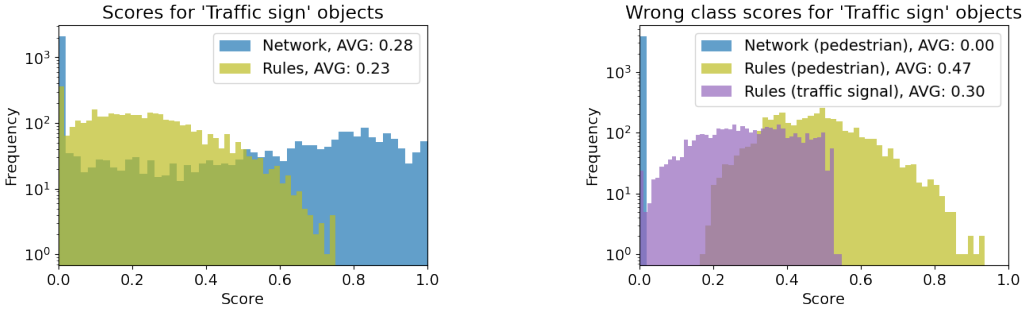
0.476: ~INTERSECTIONBEHIND(O3, 'bckgr', I) >> ~ISTYPE(O3, 'tr_sign', I) ^2

1.0 * ISTYPE(O, +Type, I) = 1.0 .

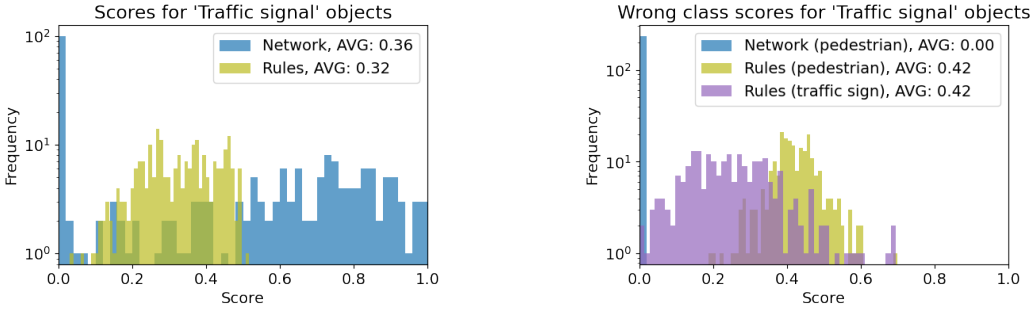
6.2. Evaluation Plots



(a) Scores distribution for the objects ”pedestrians”.

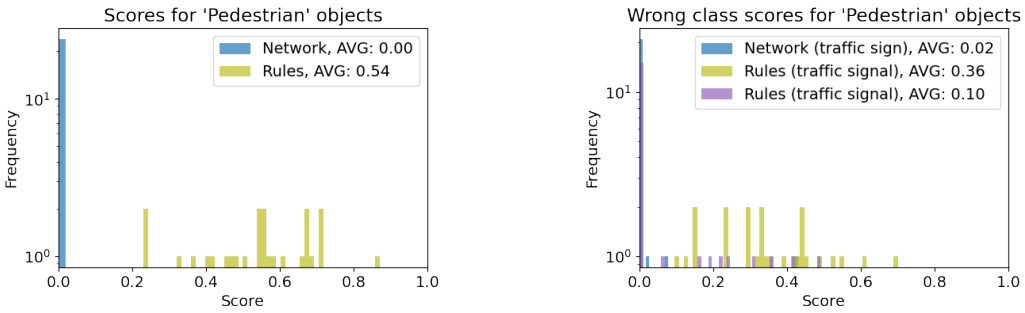


(b) Scores distribution for the objects ”traffic sign”.

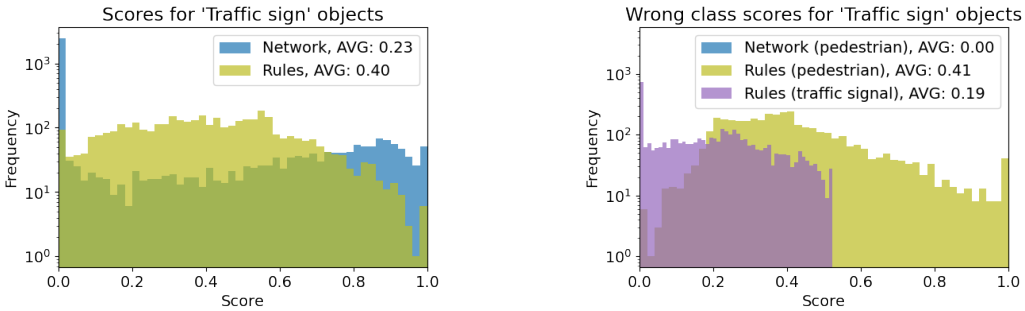


(c) Scores distribution for the objects ”traffic signal”.

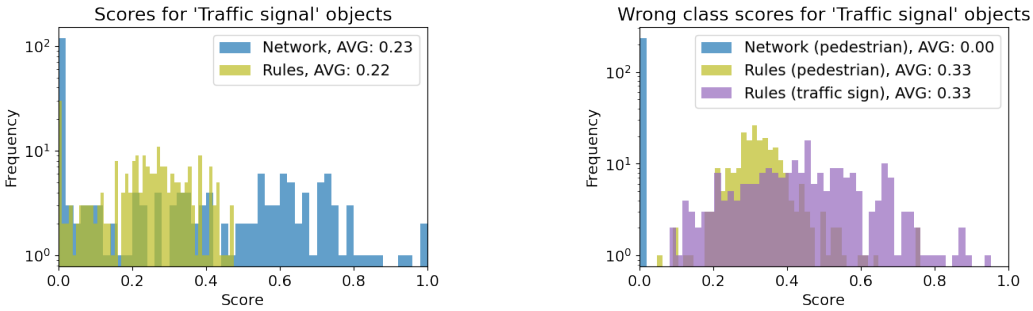
Figure 7: The scores of the network compared to the scores of the rules on the test dataset. It should be noted that the objects distribution is unbalanced in the test set: amount of objects ”pedestrian” is the smallest, followed by ”traffic signal”. The largest number of objects are ”traffic sign”. The network can not distingusih between traffic signs or signals and we thus only report the (joined) result as “tr_sign” for the wrong classes.



(a) Scores distribution for the objects ”pedestrians”.



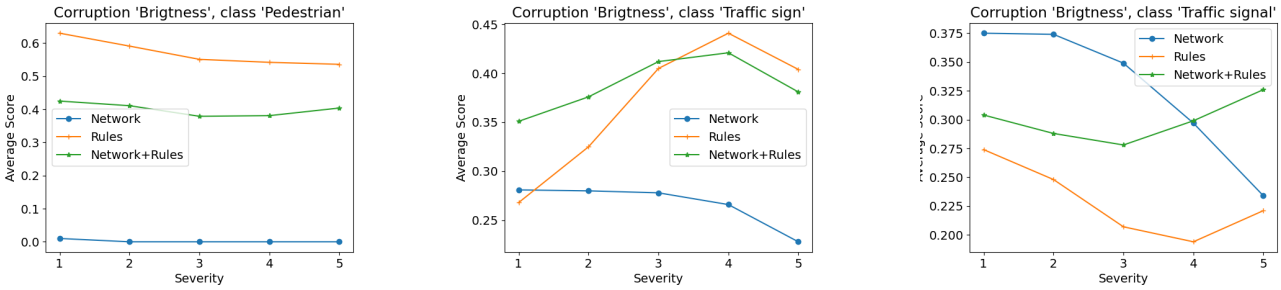
(b) Scores distribution for the objects ”traffic sign”.



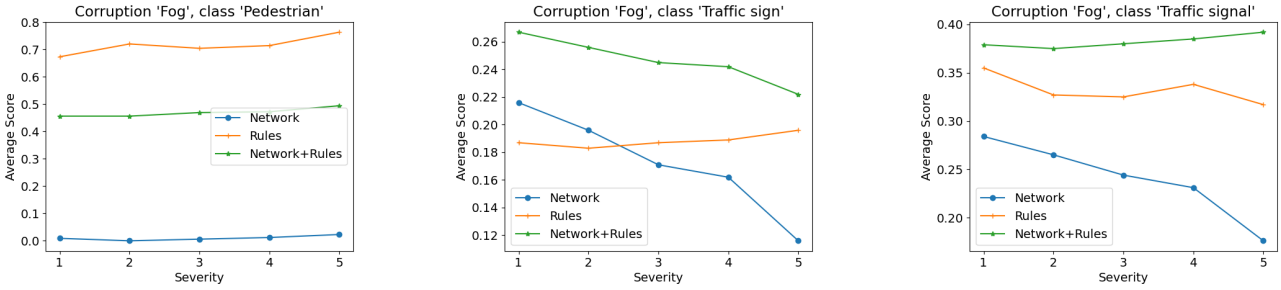
(c) Scores distribution for the objects ”traffic signal”.

Figure 8: The scores of the network compared to the scores of the rules on the corrupted test dataset. The corruption selected here is ”brightness” with the severity 5. One can directly see that the network scores quality drops substantially while rules are nearly not affected by the corruption.

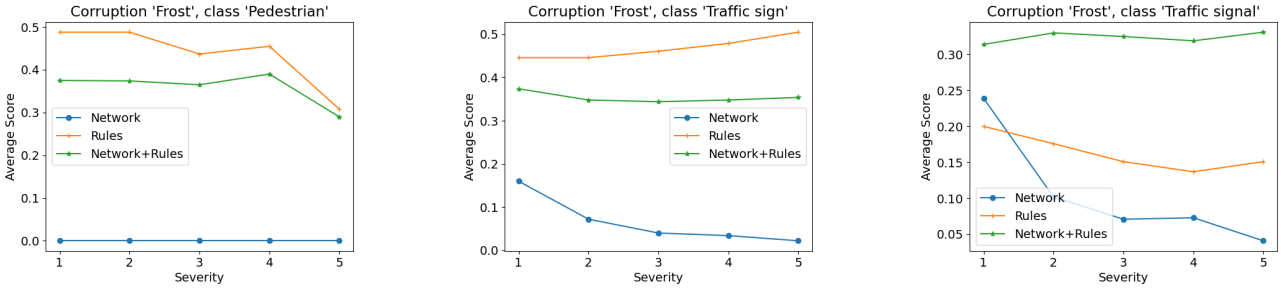
6.3. Corruption Plots



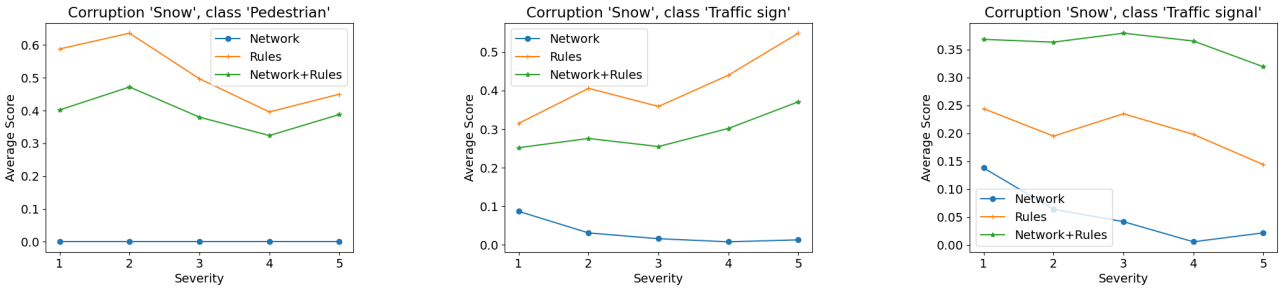
(a) Comparison of the scores under increasing severity of “brightness” corruption.



(b) Comparison of the scores under increasing severity of “fog” corruption.



(c) Comparison of the scores under increasing severity of “frost” corruption.



(d) Comparison of the scores under increasing severity of “snow” corruption.

Figure 9: Different type of corruptions with increasing severity. One can notice that corruptions can affect the rules scores: so brightness and snow add white color to the image, with that the rules for traffic signs are becoming more confident–based on the hasColor predicate. Another aspect is that amount of ’Pedestrian’ objects is very little (20 compared to 2000 of ’Traffic sign’) and because of this network mostly fails to distinguish them.