Visual Place Recognition under Image Corruptions

Peter Smit
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
p.j.m.smit@student.utwente.nl

ABSTRACT

The field of Visual Place Recognition (VPR) is concerned with finding out where an input/query photo was taken by retrieving similar geotagged images. Not all queries are perfect; some may contain corruptions like motion blur, compression, bit errors, etc. The goal of this research is to investigate the robustness of current VPR strategies against such corruptions and give insights into how it can be improved in future works. We thoroughly evaluated the robustness by introducing novel metrics. Out of the three network architectures investigated, ResNeXt-101 32x8d performs the best. Also, we found that a GeM pooling layer and Generalized Contrastive Loss function often improve corruption robustness over traditional methods. We also give some insights on evaluating long-term corruptions in VPR.

Keywords

Visual Place Recognition, Robustness, Corruption, CNN

1. INTRODUCTION

In the last decade, research on Visual Place Recognition (VPR) has progressed tremendously. VPR is a field concerned with (mainly) finding out where a place is, given an input (or often called a query) image. This can be done either by classifying the image, or by retrieving similar images which contain a location in its metadata. The second method is generally favoured, since a large amount of classes would be necessary to properly locate a query depending on the purpose. An example of how the process works is illustrated in fig. 1. An example of the usage of VPR is in robotics, where VPR can be used to identify their location without traditional techniques and limitations like GPS. It can be used for indoor navigation or outdoor, in autonomous cars for example.

Convolutional Neural Networks are a specific kind of neural network which is widely used in the field of computer vision, and thus VPR as well, due to its performance compared to other types of neural networks in this field [10, 20]. Network architectures like the many variations of ResNet[6], or AMOSNet & HybridNet[3] have achieved good results in VPR specifically. Aside from these architectures, alternatives for pooling layers in the network have also been researched. Novel pooling layers like GeM pooling[16], NetVLAD[2] and Patch-NetVLAD[5] have all shown improved performance over traditional average and max pooling in some cases.

Photos taken by the average person will not always be perfect. Often, photos can be out of focus or they can be taken on a whim, leading the photographed subject to be blurry. Photos are also usually compressed to save storage space, or to reduce the amount of data that needs to be sent when an image is transferred. As mentioned before, VPR is also used in robotics, where the camera is not always stationary, causing motion blur or zoom blur. This could lead to the image becoming harder to recognize by a neural network[7]. An image can also be noisy due to bad lighting conditions when taken or due to errors in the stored image itself. The performace of neural networks under common corruptions has been benchmarked before in Hendrycks and Dietterich's work (think of different types of blur and noise, brightness, etc.), along with the creation of modified versions of the ImageNet dataset, namely ImageNet-C[7]. Building on this, ImageNet-$\overline{\text{C}}$[13] was created which featured more experimental corruptions that are more akin to filters like hue shift, chromatic abberation, sparkles, etc.

Some research has already been done on query images under different conditions than the images the models are trained for, like weather, season, time of day, etc.[1, 8, 14, 15, 18] This research focuses on the specific problem of corruptions occurring in query images. These corruptions are different in the sense that they occur due to a difference in quality of the query rather than the conditions the query image was taken in. To the extent of our knowledge, there has been no previous research on this specific issue in the field of VPR.

The problem will be further laid out in section 2. Some related works regarding corruptions in VPR are discussed in section 3. In section 4 the approach of the research is explained in detail, with section 4.1 describing the corruptions, the execution of the experiments in section 4.2 and the evaluation metrics in section 4.3. Then, the results are laid out in section 5, and the discussion of the results is in section 6. Finally, we draw some conclusions and possible future work is discussed in section 7.

2. PROBLEM STATEMENT

In real world cases, not all query images are without problem. Corruptions like blur, weather conditions, compression, etc. are hard to account for when training CNNs for VPR. These types of corruptions can occur due to a variety of things like bad camera quality, unstable capturing of the image or due to the way the image is stored as a file. There has been some research on different weather

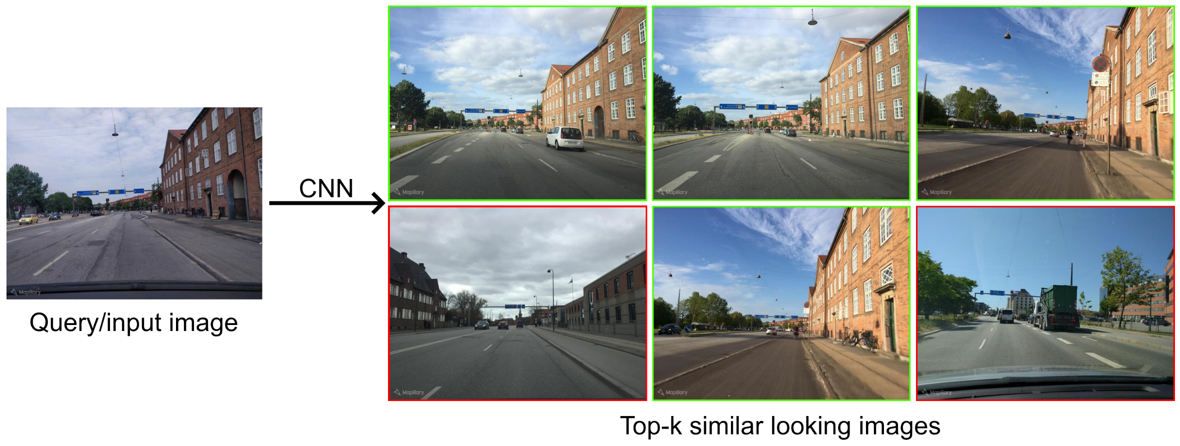

Figure 1. Example of the image selection process usually used in VPR.

conditions however, see section 3. These corruptions are estimated to occur less in indoor conditions, since it is a more controlled environment, thus leading to less motion or no image compression. For this reason we focus exclusively on outdoor VPR.

To investigate corruptions in VPR we have to answer the following research qustions:

- What types of corruption are relevant?
- What CNNs should be investigated and how will they be evaluated?
- How will the chosen CNNs perform with VPR under queries that have been affected by certain types of corruptions?

To answer these questions, we will first look at what types of corruption functions already exist and, if needed, create new ones. These corruptions will be chosen such that they are relevant for VPR specifically. Next, existing CNN models are chosen to be evaluated. Due to a lack of time in the research process, we were unable to train any models of our own. Then to evaluate the performance of these models, two novel metrics are used inspired by the metrics introduced in Hendrycks and Dietterich's ImageNet-C paper[7].

3. RELATED WORK

Touched upon in the introduction, images that were taken in different conditions such as weather and nighttime could also be though of as corruptions. There have been various papers that studied VPR under such conditions. In particular, Porav et al. developed a way to (reversibly) transform an image's conditions like tranforming from day to night, summer to winter, etc.[15]. Later, Anoosheh et al. created ToDayGAN, a Gerative Adversarial Network (GAN) to achieve the same for night-to-day transforming. These methods were used for the localization of robots. A newer method for general VPR under low light is Jenicek and Chum's normalization method[8] based on U-Net[17] which significantly helps CNNs in identifying locations.

As mentioned before, Mintun et al. have developed corruptions that expand upon the corruptions developed by Hendrycks and Dietterich[7] and have worked these into datasets called ImageNet-$\overline{\text{C}}$ and CIFAR-10-$\overline{\text{C}}$[13], based on the ImageNet[4] and CIFAR-10[9] datasets respectively. These corruptions are - in contrast to Hendrycks and Dietterich's - not corruptions that would occur naturally, but serve as a benchmark for neural networks to be able to generalize to any kind of corruption, wether it be man made or not.

4. METHODOLOGY

4.1 Corruptions

The corruptions that were used to make ImageNet-C[7] were created with computer vision in general in mind. A selection was made to accomodate for corruptions that could occur in VPR specifically. These corruptions are: **Shot noise**, which can naturally occur in photos. **Defocus blur**, which occurs when the subject is out of focus. **Motion blur** and **Zoom blur** can occur if a fast moving robot is taking a photo. **Snow**, **frost** and **fog** are all types of weather corruptions. **Brightness** can occur with poor camera quality on a sunny day. **Elastic transform** is meant to simulate a perspective change. **JPEG compression** appears when an image is saved in the lossy JPEG format. **Rotate** occurs when the camera is tilted to one side. Finally, **Crop** is used to investigate how networks would react if a portion of the image is unavailable. An example of all corruptions used is shown in fig. 2. Each corruption's severities were also adjusted to more accurately portray realistic scenarios. A visualization of all corruptions and their severities can be found in appendix A.

In addition to the corruptions listed above, we added two corruptions of our own that we think will be relevant to VPR. The first corruption is *Rotate*. This corruption simulates a camera being tilted and having some of its view obstructed. The second is *Crop*, which crops the image to a specific ratio depending on the severity to a random position. The goal of this corruption is to examine how well the networks still perform when there are less objects to recognize. These corruptions are also shown in fig. 2 and appendix A.

4.2 Experimental setup

Firstly, ResNet-50 will be used as a basis as we will be expanding Leyva-Vallina et al.'s work[11], but also due to its wide usage and overall performance under normal conditions. The chosen pooling methods are Global Average and GeM[16] pooling. For loss functions, a binary Contrastive Loss function and Leyva-Vallina et al.'s Generalized Contrasive Loss function[11] are chosen. All combinations of network, pooling layer and loss function are experimented with for ResNet-50 & ResNet-152. These networks are variations of the ResNet architecture[6]. The only other network is evaluated is ResNeXt-101 32x8d[12], which will

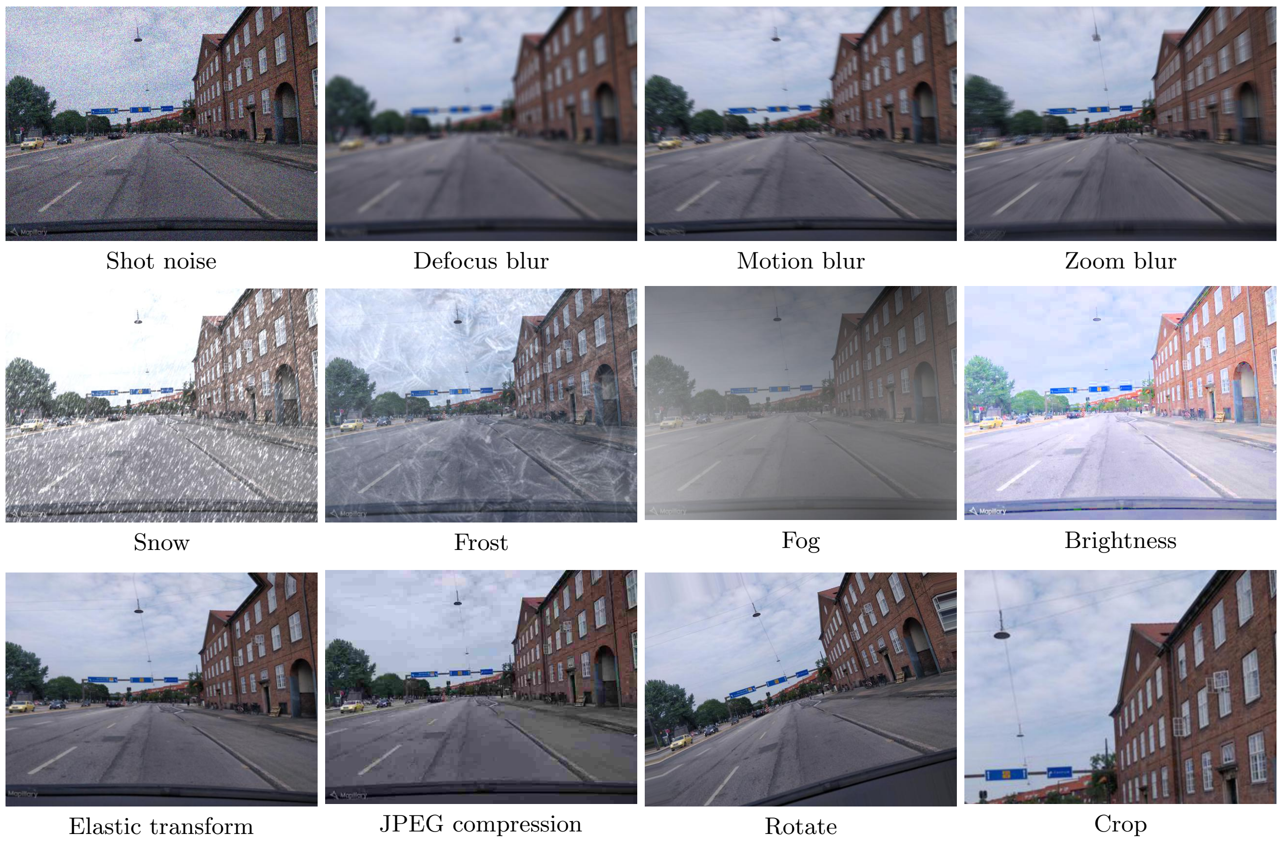

Figure 2. Every type of corruption used in the experiments. See appendix A for all severity levels. Original image taken in Copenhagen, from the MSLS validation set.

be called ResNeXt afterwards for the sake of brevity. No other networks are evaluated due to the availability of existing models, and since the research period only spans 10 weeks there was insufficient time to train our own models.

Since, as described in section 1, most corruptions occur due to bad photography conditions, we decided to focus on outdoor VPR. We focused on the Mapillary Street-level Sequences[19] (MSLS) dataset, which is a vast set of images from cities around the world. This dataset was chosen such that the results could be easily compared to existing results from Leyva-Vallina et al.'s work[11]. The models are trained on MSLS's training set and evaluated on both cities (Copenhagen and San Francisco) in the validation set. As per Hendrycks and Dietterich's recommendations, the models are trained on clean non-corrupted images such that the results accurately reflect real world robustness. The images used in the experiments are 640×480 pixels.

To apply the corruptions, code from Hendrycks and Dietterich's robustness repository[1] was used and under the Apache-2.0 License as a reference implementation. The following changes were made to the original code:

- Corruption functions were changed to accomodate for any size image instead of being restricted to 224×224 pixels.
- Crop and Rotate corruptions were added
- For corruptions that include randomness, a `RandomState` object is used and initialised with a seed. This ensures that for every run of the corrupt function, the same image is output, while the corruption itself will be different for every image

[1]`https://github.com/hendrycks/robustness`

As outlined before, the existing models from Leyva-Vallina et al. are used. The experiments were run on a PowerEdge R730 and an NVIDIA Jetson AGX on ITC's CRIB platform [2].

4.3 Evaluation metrics

The results will be evaluated per severity of corruption on a few metrics. Since the models will perform a nearest neighbour search as the final step of the image retrieval process, we can evaluate both *Top-k Recall* or *R@k* for short and *Top-k mean Average Precision* or *mAP@k* for short. Top-k Recall is the percentage of queries for which at least one correct image is chosen from its k nearest neighbours, while Top-k mean Average Precision is the average amount of correct neighbours over all k chosen neighbours. Both of these metrics will be able to display how well the models perform under these corruptions.

There is a lot of data that we can compile, since each of the 12 corruptions has 5 different severities, each with their own results. To represent the data in a meaningful way, we will create a metric similar to the Corrupt Error rate introduced in the ImageNet-C paper[7].

To compile the performance of a model's performance on a corruption we introduce the *Corrupt top-k Recall* for corruption c and pretrained model f ($\mathrm{CR@}k_c^f$).

$$\mathrm{CR@}k_c^f = \left(\sum_{s=1}^{5} R@k_{s,c}^f\right) \Big/ \left(\sum_{s=1}^{5} R@k_{s,c}^{ResNet50avgCL}\right)$$

The recall at k is summed for each severity and is standardized to a baseline model. In our case we chose ResNet-50, with average pooling and a binary Contrastive Loss func-

[2]`https://crib.utwente.nl`

Model	mCR@1	mCR@5	mCR@10	mCR@20	mCAP@1	mCAP@5	mCAP@10	mCAP@20
ResNet-50 avg CL	100	100	100	100	100	100	100	100
ResNet-50 GeM CL	125	117	114	111	125	130	130	128
ResNet-50 avg GCL	146	135	129	124	146	152	155	154
ResNet-50 GeM GCL	190	158	146	137	190	209	211	209
ResNet-152 avg CL	128	125	122	119	128	132	134	134
ResNet-152 GeM CL	150	135	129	124	150	160	162	162
ResNet-152 avg GCL	190	164	153	143	190	214	220	220
ResNet-152 GeM GCL	214	174	160	147	214	242	247	244
ResNeXt avg CL	163	146	137	130	163	176	177	175
ResNeXt GeM CL	183	157	146	136	183	205	205	202
ResNeXt avg GCL	238	190	171	156	238	274	283	280
ResNeXt GeM GCL	257	197	176	159	257	314	326	324

Table 1. Mean top-k Corrupt Recall and mean top-k Corrupt Average Precision for k's 1, 5, 10 and 20 in percentages, tested on the MSLS validation set. All models were trained on the MSLS test set.

Model	**R@1**	**mCR@1**	Shot	Defocus	Motion	Zoom	Snow	Frost	Fog	Bright.	Elastic	JPEG	Rotate	Crop
ResNet-50 avg CL	100	100	100	100	100	100	100	100	100	100	100	100	100	100
ResNet-50 GeM CL	116	125	125	120	117	122	151	142	123	122	117	119	122	120
ResNet-50 avg GCL	134	146	137	185	141	147	157	157	146	140	134	144	137	132
ResNet-50 GeM GCL	148	190	192	237	191	185	250	208	178	174	157	182	163	160
ResNet-152 avg CL	120	128	134	138	128	129	145	132	123	118	118	126	127	123
ResNet-152 GeM CL	131	150	157	147	156	150	187	156	146	136	135	152	138	139
ResNet-152 avg GCL	147	190	196	247	202	190	237	201	179	173	158	177	165	160
ResNet-152 GeM GCL	159	214	221	275	228	212	294	224	198	186	174	198	181	175
ResNeXt avg CL	133	163	173	184	162	162	235	168	165	159	134	146	146	120
ResNeXt GeM CL	141	183	185	217	191	180	259	197	179	167	150	162	162	146
ResNeXt avg GCL	163	238	253	334	259	231	330	245	217	208	182	217	191	184
ResNeXt GeM GCL	172	257	278	365	269	248	390	264	231	221	188	236	199	190

Table 2. Clean Recall@1, mean Corrupt Recall@1 and Recall@1 for each corruption type in percentages, tested on the MSLS validation set. All models were trained on the MSLS test set. Clean Recall@1 is standardized to ResNet-50 avg CL's clean Recall@1.

tion. This is done to make it easier to interpret the results. The different CR@k's for each corruption can then be averaged to a single mean Corrupt Recall@k (mCR@k) so summarize a model's performance such that it can easily be compared to the non-corrupt Recall@k. Mean Corrupt top-k Average Precision will also be calculated this way by replacing $R@k^f_{s,c}$ with $mAP@k^f_{s,c}$

Aside from this, we also introduce another metric that will more accurately portray inherent corruption robustness, rather than robustness through increased accuracy. For each corruption c and pretrained model f we can calculate the *Relative Corrupt top-k Recall*, or *Relative CR@k* for short.

$$\text{Relative CR@}k^f_c = \frac{\sum_{s=1}^{5} R@k^f_{clean} - R@k^f_{s,c}}{\sum_{s=1}^{5} R@k^{base}_{clean} - R@k^{base}_{s,c}}$$

Where again, the baseline model *base* is ResNet-50 with average pooling and binary Contrastive Loss.

This metric shows how much the corruption c has degraded performance of the model, standardized to how the base model's performance is degraded by the corruption. Similar to the previous metrics, we can average Relative CR@k^f_c for each corruption to summarize a model's robustness against the corruptions compared to the base model, we will call this Relative mCR@k^f. If a model's performance has not degraded significantly compared to the baseline, its Relative mCR@k will also be low. This means that the lower Relative mCR@k^f is, the more robust model f is.

5. RESULTS

The mean top-k Corrupt Recall and mean top-k Corrupt Average Precision were evaluated for each model in table 1. In table 2 all top-1 Corrupt Recall results are shown for each corruption. As shown before in section 4.3, the Corrupt Recall values are standardized to the corrupt recall of ResNet-50 with average pooling and binary Contrastive Loss. To quickly compare the models' performance to its corrupt performance, the leftmost columns contain the clean top-1 Recall (*R@1*), also standardized, and the mean top-1 Corrupt Recall (*mCR@1*).

fig. 3 shows each investigated model's Relative mean top-1 Corrupt Recall compared to its clean top-1 Recall. Again, both axes have been standardized to demonstrate its relative performance to the baseline. A full listing of the *Relative mCR@1* is shown in appendix B.

6. DISCUSSION

In this analysis we mostly focus on top-1 (Corrupt) Recall as the metric to evaluate a network. Also focusing on other metrics like top-5, top-10 or top-20 Recall or mean Average Precision would complicate the analysis of the results much more when comparing performance of specific corruptions.

6.1 Analysis of results

As shown in table 1 and table 2, for each different architecture, a model with GeM pooling outperforms average pooling and Generalized Contrastive Loss (GCL) outperforms binary Contrastive Loss (CL). This is not just the case for each *mCR@k* and *mCAP@k* but also for each corruption's *CR@1.* Two corruptions stand out and deserve to be mentioned, namely Defocus blur and Snow. These two

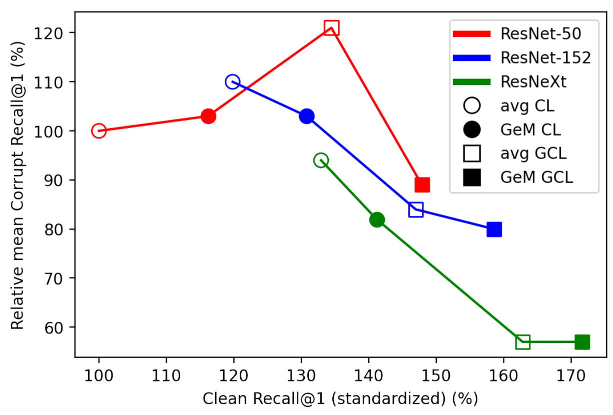

Figure 3. Relative mean Corrupt Recall@1 for each tested model. Each point's shape and fill indicate the variations of each basic architecture. An empty point means avg pooling, a filled point means GeM pooling. A circle means binary CL, a square means Generalized CL.

corruptions proved very challenging for the base network, resulting in a CR@1 of 16% and 14% respectively while its mCR@1 is 28% (each baseline CR@1 is listed in appendix B). The other models are much more robust against these corruptions, which is the reason why the CR@1s for these corruptions are so high compared to the rest.

As mentioned in section 4.3 before however, while a high mCR@1 technically shows robustness against corruptions, it is possible that this robustness is only through increased Recall, instead of being inherently robust against corruptions. This is why *Relative mean top-1 Corrupt Recall* (Relative mCR@1) is also calculated and can be seen in fig. 3. This graph shows some interesting insights. Keep in mind that a lower Relative mCR@1 indicates more robustness. This graph shows mixed results between different architectures. The robustness of ResNet-50 is worsened with the inclusion of GeM pooling and GCL separately, but combined it is improved. This is not the case with ResNet-152 and ResNeXt however, as each addition improves its Relative mCR@1. ResNeXt seems to follow a similar pattern to ResNet-152, though the Relative mCR@1 is the same for both its avg GCL and GeM GCL variants. The lowest measured Relative mCR@1 is 57% It would also be interesting to see how architectures other than these three ResNet variations perform in possible future research.

While we attempted to make the corruptions as realistic as possible, they are of course only an estimation of the real world and these digital filters will not match cover all possible situations. The five different severities of each corruption are also just an estimation and were determined by examining a few query images and seeing how they would be affected by the corruptions, then the severities were changed accordingly.

6.2 On long-term VPR performance

The previous results results give insight into a model's robustness against corruptions, though only against short-term corruptions specifically. In VPR, a distinction is usually made between short-term and long-term. With short-term meaning momentary changes to the scenery such as the corruptions we covered and long-term meaning changes over a large period of time such as a building being under construction/renovated or seasonal differences. The MSLS dataset used contains seasonal differences and though images are tagged with their date taken, there is often enough not much visual difference between images since the weather was often just cloudy or sunny. How would the models perform in the situation where, for example, there is a layer of snow all over the city?

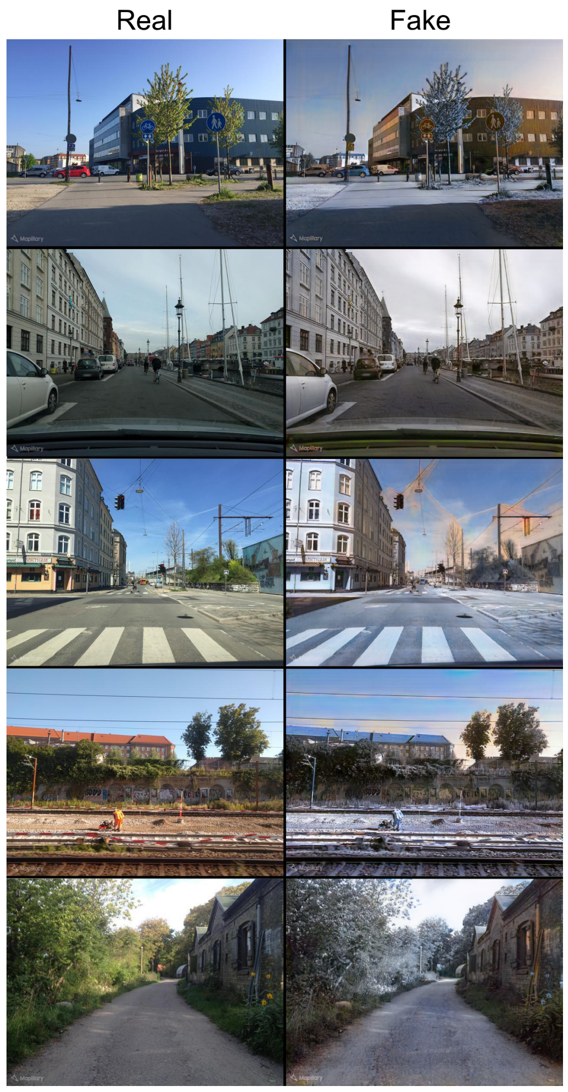

Figure 4. A selection of the winter images generated by CycleGAN originally from the MSLS validation set, taken in Copenhagen.

We investiged the possibility of using a Generative Adversarial Network (GAN) to easily generate snowy versions of the original query images. This proved to be a difficult problem however. Due to time restrictions, we only focused on unpaired image generation, such that only one input image is necessary. The only reliable method we found was CycleGAN[21], using its summer to winter Yosemite model. The model's results are still suboptimal however. As is indicated in the name, this model was trained on mountainscapes as opposed to cityscapes which is likely the reason it performs poorly. A few examples can be seen in fig. 4. As a small experiment, we transformed the Copenhagen subset of images from the MSLS vaidation set with the Yosemite summer2winter model and evaluated its top-1 recall for the base model and best performing model. The clean recall of our base model is 44.3%. With winter images it dropped to 28.1%. The best performing model's clean and winter recall are 76.1% and 64.7% respectively.

This leads to a CR@1 of 230% for the winter corruption (severities are disregarded for this experiment). These results are comparable to the CR@1 of ResNeXt GeM GCL for the other corruptions.

7. CONCLUSIONS AND FUTURE WORK

In this study, we examined some common corruptions that occur in VPR and evaluated the performance of some existing models under these corruptions. The chosen corruptions are mostly based on Hendrycks and Dietterich's[7] and two of our own were made. These are corruptions that commonly occur under different weather, lighting, stability and other conditions.

We investigated three different ResNet variations, two of which with combinations of average pooling & GeM pooling and binary Contrastive Loss (CL) & Generalized Contrastive Loss (GCL). We found that, in general, each model with a higher accuracy on uncorrupted images will also perform better when under corruptions. Each model did not perform unexpectedly bad compared to the baseline model. We found that, while also being the most accurate model, ResNeXt with GeM pooling and GCL has the most inherent robustness against corruptions, with a Relative mean top-1 Corrupt Recall of 57%. The addition of GeM and GCL did not seem to have a significant effect on ResNet-50's inherent robustness, while for ResNet-152 both improved it. While these results are focused on short-term VPR, we also discussed the possibilities of evaluating long-term VPR, specifically seasonal changes.

For future studies, it would be good to include more different network architectures since in this study only three architectures with different variations were investigated. We saw that inherent corruption robustness differs largely with each model so it would be interesting to see how robust technologies like NetVLAD[2] and HybridNet[3] will be against these corruptions. As outlined before in section 6.2, more in-depth research can be done on long-term corruptions. For seasonal changes, it would be interesting to evaluate the same models on a large scale dataset specifically designed with seasonal changes in mind. Another way to evaluate this could be a GAN to style transfer existing images to different season, though this GAN would have to be designed for dealing with cityscapes, since outdoor VPR occurs mostly in cities.

References

[1] A. Anoosheh, T. Sattler, R. Timofte, M. Pollefeys, and L. V. Gool. Night-to-day image translation for retrieval-based localization. *Proceedings - IEEE International Conference on Robotics and Automation*, 2019-May:5958–5964, 9 2018. ISSN 10504729. doi: 10.1109/ICRA.2019.8794387. URL `https://arxiv.org/abs/1809.09767v2`.

[2] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:1437–1451, 11 2015. ISSN 01628828. doi: 10.1109/TPAMI.2017.2711011. URL `https://arxiv.org/abs/1511.07247v3`.

[3] Z. Chen, A. Jacobson, N. Sunderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford. Deep learning features at scale for visual place recognition. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3223–3230, 1 2017. ISSN 10504729. doi: 10.1109/ICRA.2017.7989366. URL `https://arxiv.org/abs/1701.05105v1`.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. pages 248–255, 3 2010. doi: 10.1109/CVPR.2009.5206848.

[5] S. Hausler, S. Garg, M. Xu, M. Milford, and T. Fischer. Patch-netvlad: Multi-scale fusion of locally-global descriptors for place recognition. 3 2021. doi: 10.1109/cvpr46437.2021.01392. URL `https://arxiv.org/abs/2103.01486v1`.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:770–778, 12 2015. ISSN 10636919. doi: 10.1109/CVPR.2016.90. URL `https://arxiv.org/abs/1512.03385v1`.

[7] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *7th International Conference on Learning Representations, ICLR 2019*, 3 2019. URL `https://arxiv.org/abs/1903.12261v1`.

[8] T. Jenicek and O. Chum. No fear of the dark: Image retrieval under varying illumination conditions. *Proceedings of the IEEE International Conference on Computer Vision*, 2019-October:9695–9703, 8 2019. ISSN 15505499. doi: 10.1109/ICCV.2019.00979. URL `https://arxiv.org/abs/1908.08999v1`.

[9] A. Krizhevsky. Learning multiple layers of features from tiny images. pages 32–33, 2009. URL `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60:84–90, 5 2017. ISSN 15577317. doi: 10.1145/3065386. URL `https://dl.acm.org/doi/abs/10.1145/3065386`.

[11] M. Leyva-Vallina, N. Strisciuglio, and N. Petkov. Generalized contrastive optimization of siamese networks for place recognition. 3 2021. URL `https://arxiv.org/abs/2103.06638v1`.

[12] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten. Exploring the limits of weakly supervised pretraining. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11206 LNCS:185–201, 5 2018. ISSN 16113349. doi: 10.1007/978-3-030-01216-8_12. URL `https://arxiv.org/abs/1805.00932v1`.

[13] E. Mintun, A. Kirillov, and S. Xie. On interaction between augmentations and corruptions in natural corruption robustness. 2 2021. URL `https://arxiv.org/abs/2102.11273v2`.

[14] T. Naseer, W. Burgard, and C. Stachniss. Robust visual localization across seasons. *IEEE Transactions on Robotics*, 34:289–302, 4 2018. ISSN 15523098. doi: 10.1109/TRO.2017.2788045.

[15] H. Porav, W. Maddern, and P. Newman. Adversarial training for adverse conditions: Robust metric localisation using appearance transfer. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1011–1018, 3 2018. ISSN 10504729. doi: 10.1109/ICRA.2018.8462894. URL `https://arxiv.org/abs/1803.03341v1`.

[16] F. Radenovic, G. Tolias, and O. Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:1655–1668, 7 2019. ISSN 19393539. doi: 10.1109/TPAMI.2018.2846566.

[17] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9351:234–241, 5 2015. ISSN 16113349. URL `https://arxiv.org/abs/1505.04597v1`.

[18] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:257–271, 2 2018. ISSN 01628828. doi: 10.1109/TPAMI.2017.2667665.

[19] F. Warburg, S. Hauberg, M. López-Antequera, P. Gargallo, Y. Kuang, and J. Civera. Mapillary street-level sequences: A dataset for lifelong place recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2623–2632, June 2020. doi: 10.1109/CVPR42600.2020.00270.

[20] M. Zaffar, A. Khaliq, S. Ehsan, M. Milford, and K. McDonald-Maier. Levelling the playing field: A comprehensive comparison of visual place recognition approaches under changing conditions. 3 2019. URL `https://arxiv.org/abs/1903.09107v2`.

[21] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *Proceedings of the IEEE International Conference on Computer Vision*, 2017-October:2242–2251, 3 2017. ISSN 15505499. doi: 10.1109/ICCV.2017.244. URL `https://arxiv.org/abs/1703.10593v7`.

APPENDIX

A. EXAMPLE OF SOME CORRUPTIONS' SEVERITIES

In fig. 5 we outline some examples of each severity of a few corruptions. There corruptions are from top to bottom: *Zoom blur*, *Snow*, *Fog* and *Crop*.

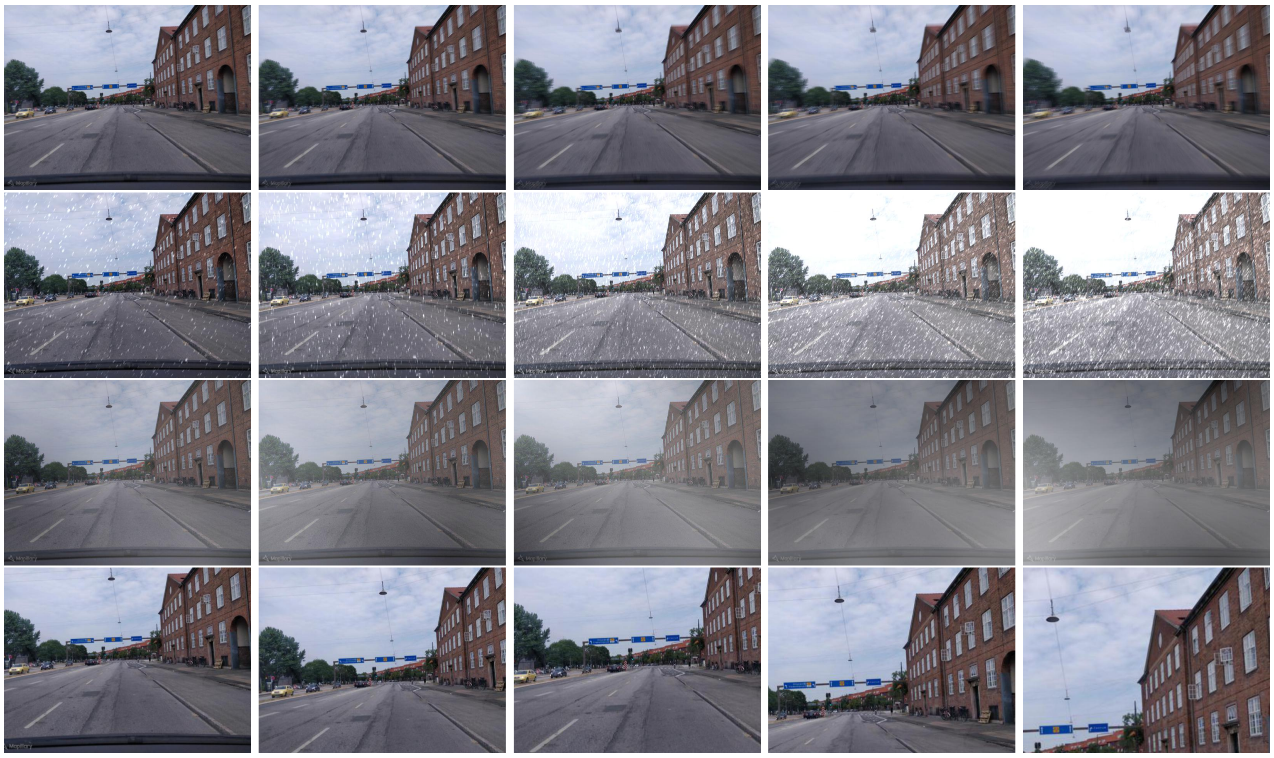

Figure 5. Each of the five severity levels of the corruptions *Zoom blur*, *Snow*, *Fog* and *Crop* from top to bottom. Severity levels are 1 to 5 from left to right for each corruption.

B. FULL CORRUPT RECALL RESULTS

As all previous results are standardized to the performance of ResNet-50 avg CL, we have detailed the non-standardized results in table 3. The Relative top-1 Corrupt Recall for each corruption and the mean is laid out in table 4.

R@1	**mCR@1**	Shot	Defocus	Motion	Zoom	Snow	Frost	Fog	Bright.	Elastic	JPEG	Rotate	Crop
44.3	27.5	24.8	16.4	24.7	28.1	13.6	25.7	31.9	31.4	38.2	29.4	34.9	30.9

Table 3. Non-standardized results for ResNet-50 avg CL, all in percentages. Each corruption's result is the Corrupt Recall@1 for that corruption.

Model	**Rel. mCR@1**	Shot	Defocus	Motion	Zoom	Snow	Frost	Fog	Bright.	Elastic	JPEG	Rotate	Crop
ResNet-50 avg CL	100	100	100	100	100	100	100	100	100	100	100	100	100
ResNet-50 GeM CL	103	105	114	115	105	101	80	98	103	109	111	93	107
ResNet-50 avg GCL	121	131	105	126	113	124	103	104	121	140	116	124	140
ResNet-50 GeM GCL	89	92	95	94	83	103	65	70	84	91	81	92	119
ResNet-152 avg CL	110	102	109	109	105	108	102	113	124	128	108	93	113
ResNet-152 GeM CL	103	98	121	100	98	106	97	93	117	103	90	105	112
ResNet-152 avg GCL	84	85	88	78	71	107	73	63	84	75	89	82	117
ResNet-152 GeM GCL	80	80	90	72	66	98	68	57	92	62	82	73	121
ResNeXt avg CL	94	82	103	96	83	87	85	50	70	124	107	82	163
ResNeXt GeM CL	82	85	96	79	74	89	64	43	77	87	99	65	130
ResNeXt avg GCL	57	48	62	42	45	88	49	24	53	42	57	56	114
ResNeXt GeM GCL	57	36	58	49	40	74	44	18	51	71	45	71	131

Table 4. Relative mean Corrupt Recall@1 and Recall@1 for each corruption type in percentages, tested on the MSLS validation set. All models were trained on the MSLS test set.