Robust Image Geolocalization

Akhil Arularasu  
Emory University  
Atlanta, Georgia  
akhil.arularasu@emory.edu

Parth Parag Kulkarni  
The University of Central Florida  
Orlando, Florida  
parthparag.kulkarni@ucf.edu

Gaurav Kumar Nayak  
The University of Central Florida  
Orlando, Florida  
gauravkumar.nayak@ucf.edu

Mubarak Shah  
The University of Central Florida  
Orlando, Florida  
shah@crcv.ucf.edu

Abstract

Popular contemporary geolocalization methods, both classification-based as well as retrieval-based, do not account for noisy images which often occur in the real world. This is a common flaw in geolocalization. Existing geolocalization datasets all contain images that are high resolution and clean without any natural or artificial perturbations. Consequently, current geolocalization techniques like TransGeo also make the same assumptions implicitly, rendering their models vulnerable to perturbations, which often occur in real-world images. We propose an innovative approach that forces the model to incorporate a diverse range of artificial noise types to address the problem. Extensive experiments on the benchmark CVUSA dataset using our enhanced TransGeo model demonstrate the effectiveness of our proposed method, achieving significant improvements in matching accuracy and thus improving the model’s robustness. Our research presents a pioneering direction in robust image geolocalization by introducing noise-based augmentation to TransGeo and creating new avenues for geolocalization applications in real-world scenarios. Code is available here.

1. Introduction

Geolocalization is a fundamental task in computer vision with numerous practical applications in urban planning, disaster management, and geospatial analysis. While current state-of-the-art Transformer-based models in retrieval-based geolocalization such as TransGeo have been effective, they have a major weakness: processing and evaluating the model’s noisy images. In the real world, images are often perturbed and one cannot proceed with the expectation that they will be clean 100% of the time. Our innovative approach highlights this weakness as a model pre-trained on clean images degrades when evaluated on images with the following torchvision noises: gaussian noise, gaussian blur, hue, saturate, brightness, and contrast of various levels of severity (One through Five) for each.

However, after being trained on a combination of noisy and clean images, the model shows significant improvement when evaluated on noisy images. In particular, when the model is trained on a single noise (Gaussian Noise), its evaluation accuracy for Gaussian Noise increased significantly as opposed to when the model was trained on a combination of all of the noises, its evaluation accuracies for the other noises increased significantly. This is because a model trained on multiple noises is more general as opposed to a model trained on just a single noise which is why it performs better. A more general conclusion though, is that training the TransGeo model on noisy images makes it more robust to noise.

Future work in this direction involves using a binary classifier to determine if an image is noisy or not. If it is noisy, we must ‘denoise’ the image using various techniques and then feed it into the pre-trained model, thus reducing the need for the time-consuming process of training on perturbed images.

2. Related Work

Prior to TransGeo, existing works in the field of cross-view geo-localization [3, 25, 11, 12, 19, 21, 22] generally adopted a two-stream CNN framework to extract different features for two views, then learn an embedding space where images from the same GPS location are close to each other. However, they failed to model the significant appearance gap between the two views, resulting in poor retrieval performance. VIGOR[27] proposed a new urban dataset assuming that the query can occur at arbitrary loca-
tions in a given area, so the street-view image is not spatially aligned at the center of the aerial image. TransGeo showed that a vision transformer can tackle this challenging scenario with learnable position embedding on each input patch. It was seen that L2LTR\cite{23} adopts vanilla ViT\cite{5} on top of ResNet\cite{6}, resulting in a hybrid CNN+transformer approach. Since it adopts CNN as a feature extractor, the self-attention and position embedding are only used on the high-level CNN features, which does not fully exploit the global modeling ability and position information from the first layer. Furthermore, it requires significantly larger GPU memory\cite{23} and pre-training dataset than CNN-based methods, while the TransGeo approach is GPU memory-efficient and uses the same pre-training dataset as CNN-based methods, e.g. SAFA\cite{17}.

Additionally, the paper "Benchmarking Neural Network Robustness to Common Corruptions and Perturbations" \cite{7} is an existing work that the robustness of neural networks when exposed to various types of corruptions and perturbations. The authors conducted comprehensive experiments using different neural network architectures and benchmark datasets to evaluate the models’ performance under challenging conditions.

The key findings indicated that while neural networks demonstrate impressive accuracy on clean data, they are susceptible to a wide range of corruptions and perturbations commonly encountered in real-world scenarios. This highlights the need for improved robustness in machine learning models to ensure consistent and reliable performance in practical applications.

This paper relates to our work in evaluating neural network robustness after introducing noise to training images in the TransGeo model. The research serves as a valuable reference to underscore the importance of addressing robustness concerns and enhancing the performance of TransGeo under various noisy conditions.

TransGeo made big strides in improving Geolocalization techniques. The Robustness paper highlighted the importance of making machine learning models robust to image corruptions and perturbation. However, an important question still has to be addressed: how do these models perform when tested on perturbed images (as is often the case in the real world)?

3. TransGeo

The TransGeo\cite{24} model changed the approach discussed in section\cite{2} becoming the first pure Transformer-based method for cross-view geo-localization without relying on polar transform or data augmentation. TransGeo leverages the transformer’s self-attention mechanism to capture contextual relationships across the entire image, making it well-suited for cross-domain matching tasks like geolocalization.

<table>
<thead>
<tr>
<th>Method</th>
<th>R@1</th>
<th>R@5</th>
<th>R@10</th>
<th>R@1</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVM-Net\cite{9}</td>
<td>22.47</td>
<td>49.98</td>
<td>63.18</td>
<td>93.62</td>
</tr>
<tr>
<td>Liu\cite{12}</td>
<td>40.79</td>
<td>66.82</td>
<td>76.36</td>
<td>96.12</td>
</tr>
<tr>
<td>Reweight\cite{4}</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>98.30</td>
</tr>
<tr>
<td>Regmi\cite{16}</td>
<td>48.75</td>
<td>-</td>
<td>81.27</td>
<td>95.98</td>
</tr>
<tr>
<td>Revisit\cite{26}</td>
<td>70.40</td>
<td>-</td>
<td>-</td>
<td>99.10</td>
</tr>
<tr>
<td>SAFA\cite{17}</td>
<td>81.15</td>
<td>94.23</td>
<td>96.85</td>
<td>99.49</td>
</tr>
<tr>
<td>L2LTR\cite{23}</td>
<td>91.99</td>
<td>97.68</td>
<td>98.65</td>
<td>99.75</td>
</tr>
<tr>
<td>†SAFA\cite{17}</td>
<td>89.84</td>
<td>96.93</td>
<td>98.14</td>
<td>99.64</td>
</tr>
<tr>
<td>†Shi\cite{18}</td>
<td>91.96</td>
<td>97.50</td>
<td>98.54</td>
<td>99.67</td>
</tr>
<tr>
<td>†Toker\cite{20}</td>
<td>92.56</td>
<td>97.55</td>
<td>98.33</td>
<td>99.57</td>
</tr>
<tr>
<td>†L2LTR\cite{23}</td>
<td>94.05</td>
<td>98.27</td>
<td>98.99</td>
<td>99.67</td>
</tr>
</tbody>
</table>

Table 1: Comparison of previous works including TransGeo in terms of Recall R@k (%) on CVUSA. “†” indicates methods using polar transform.

In Fig.\cite{1} we see that the TransGeo model consists of two essential components: the query network and the reference network. These networks process street-view (query) and satellite (reference) images, respectively. The query network receives street-view images as input and processes them through a stack of transformer encoder layers. The transformer encoder uses self-attention to effectively learn feature representations from the query images. Similarly, the reference network processes satellite images using transformer encoder layers. This network learns feature representations from satellite images by leveraging the self-attention mechanism. TransGeo performs cross-domain matching between the query and reference features to determine the geographic location of the street-view images. This matching is achieved by aligning relevant features output by the two networks. By comparing the query and reference images in a shared feature space, TransGeo effectively learns to map street-view images to their corresponding aerial images. Training is done in a supervised manner using a large-scale dataset of paired street view and satellite images with annotated geographic coordinates. The model learns to minimize the geolocalization error between the predicted and ground-truth coordinates during training.

The model is trained using the soft-margin triplet loss function\cite{9} which is represented by the equation

\[ L_{triplet} = \log(1 + e^{\alpha(d_{pos} - d_{neg})}) \]  

(1)

TransGeo employs a unique ‘attend-and-zoom’ strategy where the model uses an attention-guided non-uniform cropping method such that uninformative image patches are removed with negligible drop in performance as a way to reduce computation cost. The saved computation can be reallocated to increase resolution only for informative patches,
Figure 1: **TransGeo** Model Architecture. An overview of the proposed method. Stage-1 uses regular training by employing Eq. 1. Stage-2 follows the “attend and zoom in” strategy by increasing the resolution of the important regions of reference aerial image, using attention-guided non-uniform cropping. The patch size remains unchanged. 

resulting in performance improvement with no additional computation cost. This strategy is highly similar to human behavior when observing images. 

As shown in Table 1, TransGeo beats all of the existing state-of-the-art models at the time with its novel approach to geolocalization.

### 4. Approach

Our approach involves first validating a pre-trained TransGeo model with noisy testing images to obtain a baseline performance as mentioned in section 4.1, then incorporating various noises to the training images as mentioned in section 4.2.1 while training the models, and finally validating the models with various noisy testing images. Section 4.2.2 discusses how the loss used in training is calculated.

#### 4.1. Validating TransGeo on Noisy Images

The first step in this process was to evaluate a pre-trained TransGeo model using noisy validation query images. To accomplish this goal, we first had to identify which noises to add. We chose to add the following torchvision noises to the validation images and vary the severity levels from One through Five: Gaussian Noise, Gaussian Blur, Hue, Contrast, Brightness, and Saturate. We generated noisy images 'on the fly' during training by adding noise in the test query section of the data loader and then.

#### 4.2. Introducing Noise into Training

Following the evaluation with noise, the next logical step was to introduce noise to images during training and then evaluate them to see if there was any improvement.

##### 4.2.1 Incorporating Noise

In order to train the images with noise, we made modifications to the data loader. The TransGeo model data loader plays a critical role in efficient training as it serves as the key interface responsible for loading and pre-processing the training and validation data during training. We set up the data loader to pass both noisy as well as clean images for each image in the training set to the model. After the data loader adds in noises, both noisy and clean images are passed into the model and features are obtained. Then, we calculate the loss as described in section 4.2.2.
4.2.2 Loss Calculations

After obtaining 3 sets of features: noisy ground, clean ground, and clean aerial, we calculate the loss between clean ground and clean aerial features along with the loss between noisy ground and clean aerial separately. This ensures that the model learns to map both noisy and clean ground images as closely as possible to the corresponding aerial images in each case. The average of these two losses is the final loss which is used to train the model. The loss function used is the soft-margin triplet loss (Eq. 1).

5. Experiments

In this section, our experiments will be discussed in more detail. Sec. 5.1 refers to the dataset used in the experiments, Sec. 5.2 elaborates on the types of perturbations added to the images, and Sec. 5.3 clarifies the training details.

5.1. Dataset

The dataset used in our method and experiments is cross-view USA (CVUSA) [24]. The Dataset consists of various images of street and aerial views from different regions of the US. The task helps to determine location without GPS coordinates for the street-view images. Google Street View [1] panoramas are used as ground images, and matching aerial images at zoom level 19 are obtained from Microsoft Bing Maps [2]. The dataset comprises 35,532 image pairs for training and 8,884 image pairs for testing, and recall is the primary metric for evaluation.

5.2. Perturbations

In this section, we describe the perturbations applied to the training and testing images in our experiments. Six different perturbations were incorporated, each implemented using the torchvision library. These perturbations were applied to introduce diversity and robustness to the model’s
training process. The following noises were used:

Contrast: Contrast adjustments, seen in Fig. 2, were made to alter the difference between light and dark areas in the images.

Hue: Hue variations, seen in Fig. 3, were introduced to simulate changes in the dominant color tone of the images.

Saturate: The saturation level, seen in Fig. 4, of the images was modified to create variations in color intensity, further enriching the dataset.

Gaussian Noise: Gaussian noise, seen in Fig. 5, was added to the images to simulate real-world variations and enhance the model’s ability to generalize to noisy inputs.

Brightness: Changes in brightness levels, seen in Fig. 6, were applied to emulate varying lighting conditions in real-world scenarios.

Gaussian Blur: Gaussian blur, seen in Fig. 7, was applied to the images to mimic the effect of out-of-focus or motion-blurred scenes.

To provide different degrees of perturbation, we utilized severity levels ranging from 1 to 5 for each type of noise. These severity levels represent increasing levels of perturbation, with level 1 being the mildest and level 5 the most intense. By incorporating these diverse perturbations during training and testing, our model is better equipped to handle real-world variations, improving its robustness and generalization capabilities.

5.3. Training Details

We followed the same training details as TransGeo [24] and keep the same hyperparameters. We use $\rho = 2.5$ for ASAM [10]. The weight decay of AdamW [13] is set to 0.03, with default epsilon and other parameters in PyTorch [14]. The sampling strategy is the same as [27].

6. Results

In this section, we will be discussing the results of our experiments.

6.1. Effect of Perturbations on TransGeo

The results achieved by adding perturbations to the testing query images of the TransGeo model served as a baseline result and are shown in Table 2. It is seen that when the model is pre-trained on clean images and evaluated on noisy images, the model performs worse regardless of severity level. As seen by the graph in Fig. 8, as the severity level increases, the model performs steadily worse.

6.2. Performance of Models Trained with Different Training Procedures

Table 3 shows the validation accuracies when the TransGeo model is trained on a combination of clean and noisy images and evaluated on noisy images. The first row is a continued baseline, showing TransGeo’s accuracy on noisy images of severity level four. The second row contains the evaluation accuracies for severity level four noisy data on a TransGeo model trained on a combination of clean and Gaussian noise-only severity level four images only, while the third row contains evaluation accuracies for severity level four noisy data on another TransGeo model trained on a combination of clean images and images each with a random noise (chosen out of the six aforementioned noises) severity level four. It can be seen that when TransGeo is trained on a combination of clean and Gaussian Noise level four images, the evaluation performance of Gaussian Noise and Saturation level four testing images is the best. On the other hand, when TransGeo is trained on a combination of clean images and images with random noises level four, the evaluation performance of the remaining noises is significantly improved and the model performs better on randomly noised testing images than when trained only on clean + Gaussian Noise. This is because a model trained on multiple noises is more general as opposed to a model trained on just a single noise (Gaussian Noise) which is why it performs better.
## 7. Conclusion and Future Work

In this section, our conclusions and future steps will be discussed.

### 7.1. Conclusion

Existing geolocalization models are created on the assumption the images are high-resolution and clean without any disturbances. They unfortunately do not account for real-world situations where images are perturbed. Our proposed approach shows that when the model is modified to incorporate a diverse range of noises in its images during training, its evaluation of noisy images also greatly improves. Additionally, the amount of noisy images that the TransGeo model is trained on also affects performance; when it was trained on a combination of clean and Gaussian Noise only, the model performed very well for Gaussian Noise and Saturation testing images but when it was trained on a combination of clean and random noises, the model performed best on the other types of noises which is logical because if the model is trained on a specific type of noise it will do better when tested on that particular noise.

### 7.2. Future Work

Our future work as seen in Fig. 9 will involve creating a new model that can handle perturbations in an image such that any pre-trained model can geolocalize effectively. The procedure for this entails a 3-step process:

1. **Identify Query Image** - Pass the image through a binary classifier to ascertain whether an image is clean or noisy. If clean, pass the image directly into the pre-trained model; otherwise, process the noisy image (below)

---

### Table 3: Evaluation accuracies when TransGeo is trained on noisy data.

<table>
<thead>
<tr>
<th>Model Trained on/Eval. Noise</th>
<th>Gaussian Noise</th>
<th>Gaussian Blur</th>
<th>Hue</th>
<th>Saturation</th>
<th>Brightness</th>
<th>Contrast</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clean Images</td>
<td>66.02</td>
<td>61.75</td>
<td>53.58</td>
<td>47.65</td>
<td>24.56</td>
<td>0.12</td>
</tr>
<tr>
<td>Clean Images + Noisy Images (Lvl 4 Gaussian Noise only)</td>
<td>89.01</td>
<td>64.23</td>
<td>74.55</td>
<td>88.73</td>
<td>79.02</td>
<td>15.23</td>
</tr>
<tr>
<td>Clean Images + Noisy Images (Random choice between Lvl 4 Gaussian Noise, Gaussian Blur, Hue, Saturation, Brightness, Contrast)</td>
<td>83.57</td>
<td>78.13</td>
<td>80.32</td>
<td>83.16</td>
<td>79.02</td>
<td>76.40</td>
</tr>
</tbody>
</table>

---

**Figure 9: Future Work**
2. **Obtain Condition Vector** - Identify the type of noise with CLIP [15] encoders, matching the text encoding of the type of noise with the image encoding

3. **Train Diffusion\[8\] Model** - Using the text encoding of noise type and the image encoding as a condition, train a diffusion model to generate a clean image (image denoising)

8. **Acknowledgements**

The authors would like to thank Dr. Tanvir Ahmed and Dr. Niels Lobo for their assistance and guidance in this project. The authors would also like to thank the CRCV for providing the adequate resources required to complete this project.

**References**


