ArcGeo Supplementary Materials

1. Implementation Details

Training is performed for 500 epochs with cosine warm-up scheduler. Experiments with ResNeXt-50 backbone are performed with RangerLars optimizer [1] (RAdam optimizer with Layer-wise Adaptive Rate Scaling (LARS) [2] and lookahead [3]) with the maximum learning rate of 10^{-3} for the encoder and MHSA blocks and 2×10^{-3} for the head. Experiments with ViT-B are performed with AdamW [4] optimizer with the maximum learning rate of 3×10^{-4} and the weight decay of 10^{-2}. The batch size is close to 192, unless a different value is specified (some experiments required small adjustment to batch size to meet VRAM limitations of our compute setup).

We also conduct a series of experiments evaluating the utility of sharpness aware optimization using ASAM [5] which was demonstrated as a useful optimizer in cross-view matching applications by Zhu *et al*. [6], however it requires 32-bit precision which limits batch-size. By default, our experiments do not incorporate ASAM unless indicated otherwise.

ResNeXt-50 Model Variant: For our CNN-based experiments we use ResNeXt-50 (32x4d) [7] which contains 25M parameters and has been demonstrated as a capable foundation model for many vision tasks. We conduct experiments using weights from semi-weakly supervised ImageNet pretraining [8] as well as large scale pretraining on a more extensive cross-view matching dataset [9] (see evaluation results in Section 4.6).

Following feature extraction, we perform an aggregation step using a multi-head self-attention (MHSA) module containing two transformer layers. Specifically, the MHSA contains a BN + conv layer for projection of the input feature map to 1024 channels, 2 sequential transformer layers, and a conv + BN + ReLU + conv layer to perform final transformation of the feature map. The MHSA module enables interaction between all parts of the feature map, beyond the receptive field of the convolutional backbone, to update the features before pooling. We do not add a positional encoding as some previous approaches suggest [10, 11] , and instead fully rely on the positional representation produced by convolutions, as suggested by Xie, et. al [12]. The network head consists of a GeM pooling [13] followed by a standard fast.ai [14] head setup with two linear fully connected layers, which output 1024 width global embedding vector $\{X_{g_i}, X_{a_i}\}$.

ViT-B Model Variant: For our transformer-based experiments we use a ViT-B model with BEiT-v2 [15] and DEiT-v3 [16] pretraining (see Figure 1). BEiT-v2 utilizes one of the most advanced at the moment self-supervised pretraining techniques, which supposes the majority of vision benchmarks with a single model, if training is performed at large scale [17]. Meanwhile, DEiT-v3 [16] provides one of the most advanced fully supervised ViT training setups. These models were chosen because they provide close to state-of-the-art performance on a number of vision tasks, given their small model size (ViT-B model size is 86M parameters, which is more than three times larger than ResNeXt-50).

We adopt common components of many transformer networks including patch-wise image embedding, position embedding and multi-head self-attention. The aerial image $I_a \in \mathbb{R}^{H \times W \times C}$ is converted into N $P \times P$ patches (our model uses $P = 16$) yielding $I_{p_i} \in \mathbb{R}^{N \times (P \times P \times C)}$. The N patches are then flattened to dimension $\mathbb{R}^{N \times P^2 \times C}$ as input to the linear projection layers to generate image tokens $I_{t_i} \in \mathbb{R}^{N \times D}$. Image tokens are concatenated with a learnable class token to form the embedding space of the transformer model. following the standard definition for ViT transformers [18].

Similar to the ResNeXt-50 configuration, we also have independent aerial and ground transformers, but the embedding is produced directly from the output of the classification token, without using any additional pooling layer and head. The value of the classification token is modified during propagation of the input through the transformer by interaction of this token with all image tokens. The classification token contains the image summary, similar to the pooled features in our MHSA module, however it is applied at each network layer and is not limited by selection of the average value over the feature map. Since transformers intrinsically have a receptive field equal to the image size, there is also no need to add extra MHSA layers. For experiments where FoV is not 90°, the positional encoding is linearly interpolated to match the input image dimensions.

Since use of ViT-B models increases the computational cost of training, which increases quadratically with the

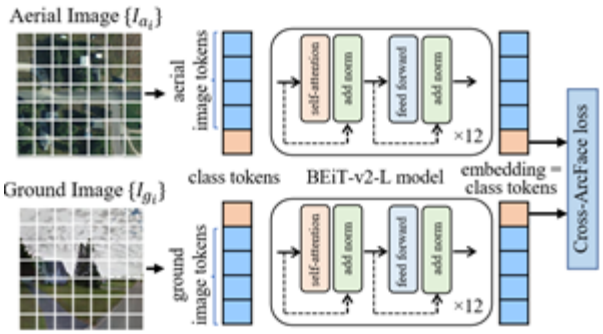

Figure 1. Visualization of query images under different test-time FoV (left) and top-5 retrieved aerial images predicted from our model. Ground truth aerial pairs are marked in green.

number of tokens, we consider only basic training without use of ASAM optimizer and global negative mining strategies. The increase of the computational cost for BEiT-B-v2 becomes apparent especially for unknown view direction or FoV = 360° cases. In these cases, the size of the considered input is large, and the GPU memory and computational cost requirements increase by approximately 16 times in comparison to the configuration using FoV = 90° and known view direction. To mitigate memory related overhead, we use gradient checkpointing which allows for large enough batch sizes for meaningful comparison.

Mixed precision is used for optimizing the training speed and reducing GPU memory requirements. Large FoV training with ViT-B model is performed with using gradient checkpointing. Training of ViT-B based setup takes approximately 2.5 days for FoV = 90° model with known view direction and approximately 8 days for FoV = 360° model at 2×V100 GPUs. In comparison, the ResNeXt50-based configuration can be trained 2-3 times faster.

2. FoV-based Data Augmentation

Some training experiments were conducted using square crop data augmentation, while evaluation is performed using horizontal cropping, following the precedent established by existing approaches. During training, when FoV-based augmentation is applied, we crop images according to a Gaussian distribution centered around the target FoV with $\sigma = 10$ similar to the approach described by Rodriques *et al*. [19]. Using this approach, ground images are cropped and resized from the original panorama corresponding to the desired FoV. Table 1 shows image sizes used for all considered FoV and cropping strategies.

3. Negative Mining Strategy

Our best performing model uses a 2-step negative mining strategy. In the first stage of training, we adopt an in-batch negative mining strategy [20] where we select the N hardest samples within the batch which are used in loss computation. The value of N is gradually reduced following an exponential decay strategy

$$N = \frac{2b}{1 + e^{3.5t}}$$

where t is the current training step expressed as a portion of the total training from 0-1, and b is the batch size. This results in hard negative pairs being gradually introduced to training, which resulted in better performance during our early experiments. The in-batch negative mining strategy is applied to the first 50% of training. Afterwards we switch to a global negative mining strategy as defined by Zhu *et al*. [20], which leverages a FIFO queue to efficiently approximate the hardest negative samples in the dataset. We maintain a queue size of roughly 8,000 image pairs and replace half of each batch with mined negatives.

FoV	Horizontal Crop	Square Crop
360	896×224	896×224
180	448×224	448×224
90	224×224	224×224
70	174×224	$174 \times 174 \rightarrow 224 \times 224$
60	149×224	$149 \times 149 \rightarrow 224 \times 224$
45	112×224	$112 \times 112 \rightarrow 224 \times 224$
30	75×224	$75 \times 75 \rightarrow 224 \times 224$

Table 1. Image sizes used for experiments with our two cropping strategies.

4. Formative Experiments

To assess the contribution of different components of our pipeline we conducted a set of initial experiments with our ResNeXt-50 setup. We consider the case of unknown view direction without polar transform and start with a dual ResNeXt-50 setup. The base setup is trained with using batch-all triplet loss with batch size of 64 for 48 epochs. As a starting option we use Adam optimizer. This setup reaches r@1 (top-1 recall) of 2.35%. Table 2 lists the change of the performance at each step of pipeline improvement.

As a first step we added learning rate warmup and image augmentation. Another addition to the model is incorporation of MHSA to enable interaction between different parts of the image beyond the model receptive field to generate features as described in Section 3.2. In the initial setup we used 512 channels and a single MHSA block. This addition improved r@1 from 2.35% to 5.59% over the baseline model.

Based on our previous experience, RangerLars optimizer [1] and cosine learning rate scheduler gives better performance than Adam for finetuning ResNeXt-50. Therefore, we switched to this optimizer in our further ResNeXt-50

	CVUSA, FoV = 90°, Unknown view-direction				
Incremental Improvements	mAP@5 (%)	r@1 (%)	r@5 (%)	r@1% (%)	r@5% (%)
base ResNeXt50	-	2.35	9.59	54.21	88.16
+ MHSA + augmentation + warmup	-	5.59	17.98	70.98	94.48
+ RangeLars optimizer + cosine LR scheduler + large bs with fp16	23.40	14.95	38.73	88.03	98.38
+ ArcGeo loss	41.47	31.24	58.45	92.96	98.74
+architecture optimization +longer training	**48.21**	**38.08**	**65.00**	**94.82**	**99.03**
DSM at FoV = 90° [24]	-	16.19	31.44	71.13	-
L2LTR at FoV = 90° [11]	-	26.92	50.49	86.88	-

Table 2. Summary of initial experiments conducted to derive our model architecture.

experiments. In addition, we enabled mixed precision fp16 training, which gave approximately x2 speedup, and allowed us to increase the batch size to 192 (larger batch size is favorable for batch-all losses). These changes improved top-1 recall to 14.95%, which is very close to the performance of 16.19% reported by Shi *et al*. [21].

The next big improvement of the model performance is achieved by switching from triplet to ArcGeo loss, which boosted r@1 nearly twice to 31.24%. Further model optimization includes use of two sequential multi-head self-attention blocks in MHSA module, increasing the embedding space and MHSA widths from 512 to 1024 as described by Shi *et al*. [22], as well as increasing the length of training to 500 epochs. These modifications boosted r@1 to 38.08%, which is significantly higher than the previously reported SOTA for cross-view matching for FoV = 90° with unknown view direction (no polar transform) of 26.92% [11].

Further improvement of the basic setup includes use of ASAM [23] optimizer, top-k in-batch mining and quasi-negative mining as well as pretraining on full CVUSA dataset, which is described in Section 3.4.

5. Effects of Negative Mining

We performed an additional set of experiments to quantify performance improvement due our two-step negative mining process. Table 3 shows results before and after application of global negative mining for both known and unknown view direction test cases. All models are trained using ArcGeo loss and received pretraining on the CVUSA-full dataset.

In both cases we observe improved performance after applying global negative mining. The improvement is more pronounced for the unknown view-direction test case where r@1 increases from 54.28% to 66.13%.

Method	View Direction	Negative Mining	r@1 (%)	r@5 (%)	r@10 (%))
Ours (ResNeXt-50) †*	Unknown	In-batch	54.28	80.36	87.47
Ours (ResNeXt-50) †◊*	Global	ArcGeo	66.13	87.51	91.90
Ours (ResNeXt-50) †*	Known	In-batch	91.00	97.53	98.48
Ours (ResNeXt-50) †◊*	Known	Global	93.49	97.93	98.75

Table 3. Quantitative results for FoV = 90° test case on CVUSA using our two-step negative mining process. Step 1 includes in-batch negative mining, followed by Step 2 which includes additional training with global negative mining. The ◊ symbol indicates models which were trained using global negative mining. The * symbol indicates our model which was pretrained on the larger CVUSA-full dataset.

6. Effects of FoV-based Data Augmentation

A key benefit of our approach is the capability to operate across a wide range of test FoVs, using a single model, requiring no knowledge about query image sensor characteristics. To further explore the relationship of train/test FoV we conducted a series of ablation studies designed to characterize the role of the FoV data augmentation described in Section 4.2. We trained several versions of our model using varied training FoVs sampled from a Gaussian distribution with mean (μ) which was shifted between experiments. For all experiments we use a fixed $\sigma = 10°$. We consider the case where view direction is known to properly isolate FoV as the variable of interest. We report performance for several evaluation conditions using our best performing ResNeXt-50 model after fine-tuning for 250 epochs in several configurations shown in Figure 2.

While our model is robust to inference in a variety of test-time FoV conditions, further finetuning using train-time FoV distribution that is representative of test-time FoV results in improved performance. For example, when test imagery has FoV = 45°, finetuning using $\mu = 45°$ yields an improved r@1 of 68.62% in contrast to 55.66% using $\mu = 90°$ training as shown in Figure 2.

In cases where test-time FoV is known, such a finetuning process would allow for more precise alignment of the model's embedding space to match the characteristics of the test domain. We observe that even when training using very small values of $\mu = 30°$, test accuracy for larger FoV does not drop dramatically. Qualitatively, we see our model is able to correctly rank images in a variety of test-time FoV as shown in Figure 2.

References

[1] Rangelars. https://github.com/mgrankin/over9000. 1, 2

[2] Y. You, I. Gitman, and B. Ginsburg. Large batch training of convolutional networks. *arXiv:1708.03888*, 2017. 1

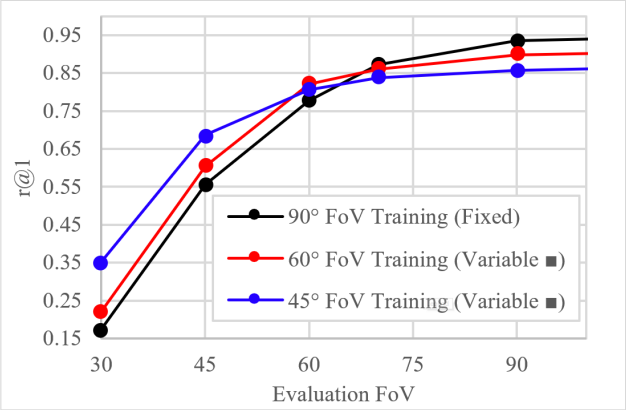

Figure 2. Performance of our ResNeXt-50 based model on limited FoV CVUSA test set after finetuning for several configurations of FoV data augmentation. The ■ symbol indicates models which were trained and evaluated using square cropping.

[3] M. R. Zhang, J. Lucas, G. Hinton, and J. Ba. Lookahead optimizer: k steps forward, 1 step back. *arXiv:1907.08610*, 2019. 1

[4] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv:1711.05101*, 2017. 1

[5] J. Kwon, J. Kim, H. Park, and I. Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. *arXiv:2102.11600*, 2021. 1

[6] S. Zhu, M. Shah, and C. Chen. Transgeo: Transformer is all you need for cross-view image geo-localization. *arXiv:2204.00097*, 2022. 1

[7] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. *Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[8] I. Z. Yalniz, H. Jégou, K. Chen, M. Paluri, and D. Mahajan. Billion-scale semi-supervised learning for image classification. *arXiv:1905.00546*, 2019. 1

[9] R. Souvenir S. Workman and N. Jacobs. Wide-area image geolocalization with aerial reference imagery. *IEEE International Conference on Computer Vision*, 2018. 1

[10] L. Liu and H. Li. Lending orientation to neural networks for cross-view geo-localization. *CVPR*, 2019. 1

[11] H. Yang, X. Lu, and Y. Zhu. Cross-view geo-localization with layer-to-layer transformer. *Neurips*, 2021. 1, 3

[12] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *arXiv:2105.15203*, 2021. 1

[13] F. Radenović, G. Tolias, and O. Chum. Fine-tuning cnn image retrieval with no human annotation. *arXiv:1711.02512*, 2017. 1

[14] The fastai deep learning library. https://github.com/fastai/fastai. 1

[15] Z. Peng, L. Dong, H. Bao, Q. Ye, and F. Wei. Beit v2: Masked image modeling with vector-quantized visual tokenizers. *arXiv:2208.06366*, 2022. 1

[16] H. Touvron, M. Cord, and H. Jégou. Deit iii: Revenge of the vit. *arXiv:2204.07118*, 2022. 1

[17] W. Wang, H. Bao, L. Dong, J. Bjorck, Z. Peng, Q. Liu, K. Aggarwal, O. Mohammed, S. Singhal, S. Som, and F. Wei. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv:2208.10442*, 2022. 1

[18] A. Dosovitskiy, L. Beyer, and A. Kolesnikov. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv:2010.11929*, 2020. 1

[19] R. Rodrigues and T. M. Global assists local: Effective aerial representations for field of view constrained image geo-localization. *WACV*, 2022. 2

[20] S. Zhu, T. Yang, and C. Chen. Revisiting street-to-aerial view image geo-localization. *WACV*, 2020. 2

[21] Y. Shi, X. Yu, D. Campbell, and H. Li. Where am i looking at? joint location and orientation estimation by cross-view matching. *arXiv:2005.03860*, 2020. 3

[22] Y. Shi, X. Yu, L. Liu, T. Zhang, and H. Li. Optimal feature transport for cross-view image geo-localization. *arXiv:1907.05021*, 2019. 3

[23] J. Kwon, J. Kim, H. Park, and I. Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. *arXiv:2102.11600*, 2021. 3

[24] R. M. Nguyen S. Hu, M. Feng and G. Hee Lee. Cvm-net: Cross-view matching network for image-based ground-to-aerial geo-localization. *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 3