Extending Absolute Pose Regression to Multiple Scenes

Hunter Blanton[1] Connor Greenwell[1] Scott Workman[2] Nathan Jacobs[1]
[1]University of Kentucky [2]DZYNE Technologies

Abstract

Direct pose regression using deep convolutional neural networks has become a highly active research area. However, even with significant improvements in performance in recent years, the best performance comes from training distinct, scene-specific networks. We propose a novel architecture, Multi-Scene PoseNet (MSPN), that allows for a single network to be used on an arbitrary number of scenes with only a small scene-specific component. Using our approach, we achieve competitive performance for two benchmark 6DOF datasets, Microsoft 7Scenes and Cambridge Landmarks, while reducing the total number of network parameters significantly. Additionally, we demonstrate that our trained model serves as a better initialization for fine-tuning on new scenes compared to the standard ImageNet initialization, converging to lower error solutions within only a few epochs.

1. Introduction

Humans are extremely adept at localizing a camera from image content alone. If the general location is known, this process typically involves identifying recognizable geographic or architectural features to estimate how and where one would have had to position themselves to produce the image. In computer vision and robotics this task is often referred to as image-based localization, or camera localization, and represents the problem of estimating the position and orientation of the camera from which an image was taken, i.e., its corresponding camera pose, with respect to some underlying scene representation.

The predominant approach to solving this problem involves constructing a 3D model of the scene and estimating the camera pose using feature-based localization, or identifying 2D-3D correspondences between the image and the known scene model. Though this approach tends to be extremely accurate, it is slow and resource intensive. Recently, learning-based approaches [6, 11, 13] have used convolutional neural networks (CNNs) to regress camera pose in an end-to-end fashion. In such approaches, the weights of the network implicitly model the underlying scene layout, as opposed to feature-based methods which explicitly take advantage of a 3D model. In these absolute pose regression techniques, each model is specialized to a single area or scene. In other words, for each scene, a network is trained using a scene-specific set of training images and corresponding ground-truth camera poses.

The major benefit of an end-to-end approach is that estimating the pose of an image requires only a single forward pass through the network, which is much faster than structure-based methods. For example, PoseNet [12] takes less than $10ms$ per image, while DSAC++ [4] takes over $100ms$ per image. However, there are several drawbacks. First, adding new data requires completely retraining the model. Second, existing datasets are limited and provide very few training examples for each scene. In the case of Cambridge Landmarks [13], for example, there are as few as 250 training samples for a given scene. Finally, recent work has shown that learning-based approaches are still significantly less accurate than structure-based methods [16].

Despite these issues, we believe absolute pose regression has many practical benefits over more accurate methods. Pose regression using a CNN can perform spontaneous relocalization in a way that (1) is deterministic, (2) is fast, and (3) runs in constant time regardless of scene size. No other method has all three of these properties simultaneously. These properties make CNN-based pose regression ideal for many tasks ranging from autonomous navigation to mixed reality where real-time, reliable performance is crucial. However, the use case is limited to the single location in which the network was trained. In this work, we propose a method to regress camera pose across several different scenes without loss of accuracy or the need to store several large networks.

We propose a variant of PoseNet which we refer to as Multi-Scene PoseNet (MSPN). Our key insight is that we can make scene identification explicit in the network architecture. We implement this as a two stage network, shown in Figure 1. The first part of the network is shared across scenes and learns a general camera localization feature. This feature is then used for scene prediction, which indexes a database of scene-specific weights, as well as final pose regression using the indexed weights. Since each scene

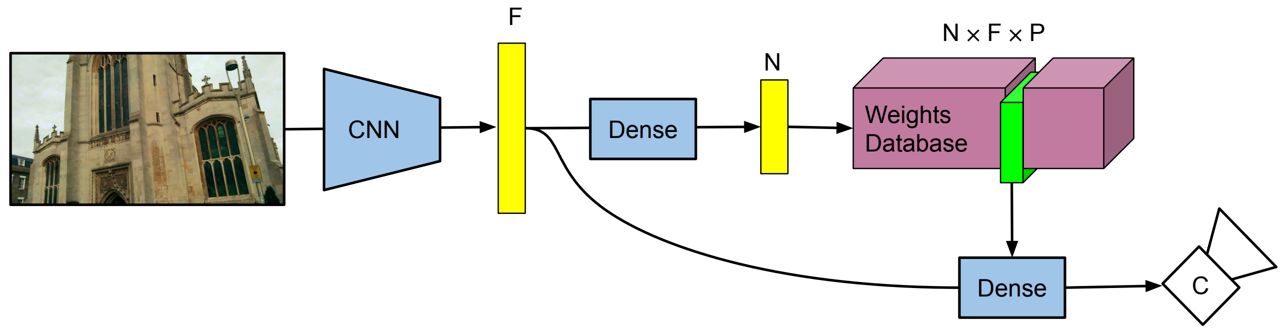

Figure 1: Our proposed Multi-Scene PoseNet architecture for multi-scene pose estimation. A convolutional neural network regresses an F dimensional image feature. This feature then goes through two branches. The first branch uses the image feature to predict a probability distribution over N possible scenes which is then used to index into a scene-specific weights database. The second branch uses the weights extracted from the weights database to construct a dense layer to transform the image features into the P dimensional pose parameters. The pose parameters are then used to construct the camera pose C.

effectively has a unique head in the network, our approach can be thought of as multi-task learning where each scene is a unique task. This allows us to move beyond memorizing each scene individually to learning a common model for camera localization across scenes. Further, MSPN enables training of the feature extractor using an order of magnitude more images, which we hypothesize may lead to more robust localization performance and improved accuracy.

We evaluate our proposed approach extensively on common benchmark datasets. Our key contributions include:

- A general method that allows for training pose regression networks on multiple scenes without sacrificing performance.
- A thorough evaluation on two benchmark datasets for absolute pose regression across multiple scenes.
- Demonstrating the value of our approach as pretraining for learning to localize in novel scenes.

2. Related Work

Estimating camera pose, i.e., the position and orientation of a camera relative to a scene, from a captured image is a fundamental problem in computer vision. Applications include 3D Reconstruction, mixed reality, and autonomous driving. Unsurprisingly, there are a wide variety of methods. For a thorough overview of work in this area, see Sattler et al. [16].

Image retrieval has been used extensively as part of localization systems. For example, a standard approach for image localization is to build a large database of images with known location, and then infer the location of a query image from the closest match (or some combination of the set of closest matches). Hays and Efros [9] demonstrated the success of image retrieval for image geolocalization using a large dataset of 6 million images. Other methods [1, 20] learn a low-level image feature and perform nearest-neighbors search to predict the location and pose of a query image. Image retrieval has been successfully applied for problems in structure-based pose regression (matching 2D to 3D) [21] and relative pose regression (estimating pose relative to a set of training images) [7]. In this work we focus on direct absolute pose regression with no retrieval step.

A large body of work has explored methods for directly estimating camera pose from an image using convolutional neural networks [6, 11, 13]. Though not as accurate as structure-based methods [16], these methods are attractive as they have fast, constant time inference regardless of scene complexity. The standard approach for such methods is to learn a feature embedding using a CNN, which implicitly captures details of the scene in the weights of the network, and then use features from this embedding to regress the camera pose. This requires only a single forward pass of the network. However, each scene typically has its own specialized model.

Methods for direct pose regression largely differ in choice of architecture or loss function. The simplest approach, PoseNet [13], trains a feed forward neural network using absolute error in orientation and position as a loss function. MapNet [6] enforces a relative pose correctness constraint that makes the final predictions more spatially consistent. LSTM PoseNets [19] replace a fully connected layer with an LSTM for feature reduction before final pose regression. Hourglass PoseNets [14] use an encoder-decoder network with skip connections. Despite the differences in these approaches, they tend to have similar performance overall. In our work, we focus on the PoseNet framework for absolute pose regression and explore how to take advantage of multiple scenes during model training. To our knowledge, our work is the first to explore methods for multi-scene absolute pose regression.

AnchorNet [15] explicitly partitions space into anchor points and regresses relative camera positions to every point. This work is similar to ours in that they take the first step towards a unified network that operates on distinct regions. However, AnchorNet still operates on a single scene at a time. The fundamental difference with our approach is that the final pose prediction is computed using a weighted average of relative anchor point poses. We instead directly regress a single absolute pose without averaging across many predictions. Our approach can be implemented in tandem with AnchorNet and we view them as complementary.

A separate branch of CNN-based pose regression is scene coordinate regression. DSAC [2] and the follow up DSAC++ [3] train a CNN to predict the 3D locations of pixels to establish 2D-3D correspondences. Next, the Perspective-3-Point algorithm and RANSAC [8] are used to compute the final camera pose. These methods are extremely accurate and compete with traditional structure-based approaches. However, they are much harder to implement and training takes much longer to converge. Also, due to the inherent randomness of RANSAC, these methods are not deterministic and have longer run-time. While some work has been done on performing pose regression across multiple scenes using these approaches [5, 18], they still require a separately trained expert network for each scene. Instead, we use a largely shared network with only a small scene-specific component.

3. Background

We describe the current paradigm of learning-based methods for absolute pose regression in a specified scene. Given a single image, the model output is a vector representing camera pose p, composed of location t and orientation q:

$$p = [t, q]. \tag{1}$$

Camera location t is a triple, $[x, y, z]$, which defines the spatial location of the camera center relative to the origin of the scene in 3D Cartesian coordinates. Camera orientation q defines a relative rotation to a canonical pose origin. Typically, for absolute pose, this is represented as a quaternion [13] instead of more traditional angle parameterization due to the inherent non-uniqueness of Euler angles. Recent approaches have had success using the log-quaternions [6]. The log-quaternion is desirable because it has fewer parameters (3 vs. 4) and naturally defines a unit quaternion without the need for explicit normalization.

During training the mean absolute error of each pose component, weighted by a hyperparameter β, is minimized:

$$L(\hat{p}, p) = \left\|\hat{t} - t\right\|_1 + \beta \left\|\hat{q} - q\right\|_1, \tag{2}$$

where $\hat{p}$ is the ground truth pose and p is the output of the network. The value of β is specific to each scene. Some approaches [13] set β before training begins, typically between 300 and 500 for indoor scenes and between 500 and 1000 for outdoor scenes. More recent works learn this weighting as part of the training process with a Laplace likelihood [11]:

$$L(\hat{p}, p) = \left\|\hat{t} - t\right\|_1 e^{-s_t} + s_t + \left\|\hat{q} - q\right\|_1 e^{-s_q} + s_q, \tag{3}$$

where s_t and s_q are optimized during training. By allowing the loss to be weighted dynamically, training can be performed without the need for hyperparameter tuning.

3.1. Theory of Absolute Pose Regression

The standard deep CNN method for absolute pose regression can be segmented into three stages [16]:

1. A function that extracts a localization feature from the image.
2. A non-linear embedding of this feature into high dimensional pose components.
3. A linear mapping from pose components to final pose using the learned basis poses.

In many learning-based approaches, the entire process is represented as a single CNN and a separate model is then trained end-to-end for each scene. For example in recent work on PoseNet [12], stage 1) is a standard ResNet-34 network, stage 2) is a fully connected layer with a ReLU activation, and stage 3) is another fully connected layer which outputs the final pose parameters. The fact that these methods use separate networks for each scene makes them very costly in terms of training time and number of stored parameters.

We propose an alternative, in which feature extraction is shared entirely by all scenes and only the final pose regression layer is trained to be scene-dependent. Additionally, We show that stage 1) and 2) above can be merged to reduce the network complexity without losing accuracy. In fact, our modification boosts performance in many cases. Our proposed alternative is more efficient in terms of learned parameters and can quickly learn to localize cameras in new scenes.

4. Multi-Scene Absolute Pose Regression

A naïve approach to estimating pose across multiple scenes is to train a single PoseNet with a training set containing images from all scenes, with no further modification. While this approach is simple, it often suffers a loss in performance over a single network per scene. We show that this decrease in performance can be overcome in many cases without a significant increase in model complexity.

We propose Multi-Scene PoseNet to decouple scene and pose using a database of scene-specific weights, shown in Figure 1. First, we use a CNN to extract an F dimensional image feature. This image feature is then used to regress an N dimensional probability distribution over each of the N possible scenes. The most likely scene (maximum probability) is used to index into a database of scene-specific weights producing an $F \times P$ fully connected layer, where P is the dimension of the pose parameters. Finally, the image feature is passed through the selected fully connected layer for final pose prediction. This is similar to ESAC [5], which uses a set of scene-specific expert networks and a gating network for expert selection. However, in our method a majority of the network is shared between all scenes and only the final layer is specific to a given scene.

The network is trained to predict the image scene and pose parameters jointly in an end-to-end manner. Pose regression is optimized using (3) and scene prediction is optimized using cross entropy. This results in features that are explicitly discriminative by scene as well as being useful for pose regression. The final loss function for pose prediction p and scene prediction s, with corresponding labels $\hat{p}$ and $\hat{s}$ becomes

$$Loss = L(\hat{p}, p) - \frac{1}{n}\sum_{i=1}^{N} \hat{s}_i log(s_i), \tag{4}$$

where N is the number of scenes.

With the exception of the fully connected layers in the weights database, all other parts of the network are shared by all scenes. This results in an extreme reduction in required parameters per scene. For example, using the approach of [12] each new scene requires its own model with over 22 million parameters. In our approach, the base network contains 22 million parameters and each new scene adds approximately 14 thousand parameters. For 10 scenes, our method uses 10% of the parameters. For 100 scenes, our method uses only 1% as many parameters. Having such a significant portion of the network shared across scenes allows for the use of deep network based localization over diverse areas, even in memory starved environments such as embedded vision systems on small autonomous drones.

4.1. Improving Network Efficiency

As described in Section 3.1, the theoretical model of absolute pose regression has three stages: feature extraction, non-linear projection of the feature vector to contravariant components in the learned pose basis, and finally a linear mapping from the learned pose basis to the canonical basis of the dataset. These components are represented explicitly in PoseNet as ResNet-34 feature extraction and global average pooling, a fully connected layer with ReLU activation, and a final fully connected layer with no activation, respectively. The ReLU activation in the second stage is troublesome as it removes the possibility of negative components to be present in the final linear mapping. Since negative pose values are not only possible, but as likely as positive values, this means that the network must learn two effective copies of any useful basis pose to be able to produce both positive and negative poses. Methods have been explored to solve this redundancy with different activations [17], but we found removing this fully connected layer altogether to be effective. Using only a single fully connected layer also results in a more compact representation for our weight database.

4.2. Implementation Details

For our experiments we use a ResNet-34 [10] feature extractor initialized from pretrained ImageNet weights. We do this to be consistent with PoseNet and MapNet for fair comparison. We train for 100 epochs with a batch size of 64 using the Adam optimizer with a learning rate of 10^{-4}. During training, we scale all input images such that their smaller dimension is 256 and then train on random 224×224 crops. At test time, we use a 224×224 square center crop. As in [12], each scene has 2 learned loss weighting parameters that are specific to the scene. For example, this means there are 14 unique loss weighting parameters learned when training with 7 unique scenes. We opt to use the log-quaternion representation of orientation due to the fact that it requires no explicit normalization for conversion to a rotation [6]. During training, the true scene is used for determining which final layer to use for pose regression. During inference, we use the most likely scene as predicted from the scene prediction branch.

5. Evaluation

We show comparisons with our method to the current state-of-the-art methods, PoseNet and MapNet. As is convention in these works, we provide median position and orientation error. All networks, including PoseNet and MapNet, use ResNet-34 as the backbone CNN.

5.1. Datasets

We train and evaluate using two common 6DOF camera pose datasets, Microsoft 7Scenes [18] and Cambridge Landmarks [13]. These two datasets have become the standard benchmarks for learning-based direct pose regression. 7Scenes is composed of 7 small indoor scenes ranging in difficulty and size, but all scenes are less than 10 meters wide. Cambridge Landmarks on the other hand is composed of 6 large-scale outdoor scenes on the order of 100s of meters wide. Both datasets were collected with hand-held cameras and ground truth data was computed using highly accurate structure-from-motion algorithms.

Method	Chess	Fire	Heads	Office	Pumpkin	Red Kitchen	Stairs
PoseNet (Single)	0.14/4.50	0.27/11.8	0.18/12.10	0.20/5.77	0.25/4.82	0.24/5.52	0.37/**10.60**
MapNet (Single)	**0.08/3.25**	0.27/11.7	0.18/13.3	0.17/**5.15**	0.22/**4.02**	0.23/**4.02**	0.30/12.10
Naïve (Multiple)	0.15/4.85	0.28/13.13	0.30/**11.54**	0.23/6.34	0.29/5.34	0.29/6.98	0.35/10.63
MSPN (Multiple)	0.09/4.76	0.29/**10.50**	0.16/13.10	**0.16**/6.80	**0.19**/5.50	**0.21**/6.61	0.31/11.63

Table 1: Results on 7Scenes dataset. Both PoseNet and MapNet are single scene approaches. The naïve approach is a single PoseNet with no modification trained on each of the scenes simultaneously. MSPN is our method. We report median translation / median orientation error. Values are highlighted red when they are significantly worse than what we are able to achieve with our modified network.

Method	Kings College	Old Hospital	Shop Facade	St. Mary's Church	Street	Great Court
PoseNet (Single)	**0.99/1.06**	2.17/**2.94**	**1.05/3.97**	**1.49/3.43**	**20.7/25.7**	-
MapNet (Single)	1.07/1.89	**1.94**/3.91	1.49/4.22	2.00/4.53	-	-
Naïve (Multiple)	1.72/3.12	2.58/5.96	1.97/9.25	2.55/6.92	48.37/45.46	12.04/10.99
MSPN (Multiple)	1.73/3.65	2.55/4.05	2.02/7.49	2.67/6.18	26.78/54.22	9.2/10.42

Table 2: Similar to Table 1 for Cambridge Landmarks. The difference between our method and the naïve approach are less significant here. We postulate that due to the capacity of the network and the small size of the training sets, our method provides less benefit for these scenes.

5.2. Effect of Joint Training

We first test the effects of the scene-specific component in the case of a scene oracle. That is to say, the true scene index is used to query the weights database instead of the output of the scene prediction head. By jointly training several scenes together with our method, we not only match, but in many cases improve upon the accuracy for each scene. By training on multiple scenes, the shared feature extraction network gets orders of magnitude more examples to learn from. Although different scenes can be seen as different domains in the context of localization, increasing the number of training examples may help increase the generality of the feature extraction, perhaps reducing over-fitting to the training sequences in each scene. Note that the batch size for an individual scene is on average $\frac{batch\ size}{N}$ for N scenes, so the effective batch size for the scene-specific weights is small for a large number of scenes.

A comparison with other methods is shown in Table 1 and Table 2. To verify that the network capacity is not simply large enough to learn all scenes, we also train a single PoseNet without any scene-specific layers on all scenes simultaneously as a baseline. Although it performs remarkably well, performance is typically much worse than our method. These results show that simply training a single model does not work well in general. Note that one method is not dominant over any other for either dataset. Keep in mind that the fundamental challenge we are trying to solve is parameter reduction for scaling to multiple scenes, but it tends to perform as well as or better than other methods. We believe forcing only the feature extraction to be generic is imposing implicit regularization, reducing the tendency of the network to overfit to the training data. Note that some scenes Cambridge Landmarks do not show the metric gain from our method compared to the baseline that is observed for 7Scenes. Due the the high capacity of the network, low number of training examples per scene, and low number of individual scenes, the naïve baseline can still perform reasonably well on this dataset. However, our methods exhibits a clear benefit for the 7Scenes dataset.

5.3. Stability With Different Scenes

As the key component of our method is training on a set of scenes at once, it is of interest to see if it performs similarly regardless of the scenes used for training. To test this, we train multiple networks where each network is trained using a different subset of all scenes in the dataset and each subset is missing a single scene. Table 3 and Table 4 show the general stability of our method to different scenes. With a few outliers, the error for a given scene remains similar regardless of which scenes we trained on. However, some scenes do seem to make learning difficult. For example, training the Landmarks dataset together without the Street scene results in uniformly improved performance by 10's of centimeters, which is a sizeable improvement. Upon inspection of the dataset, this is somewhat unsurprising as the Street training set was collected only along cardinal directions. While the strangeness of this specific scene is apparent to us, it is not trivial to determine that this scene is

	Great Court	Kings College	Old Hospital	Shop Facade	St. Mary's Church	Street
w/o Great Court	-	1.13/7.04	2.16/9.41	1.33/13.49	1.59/10.69	15.20/61.03
w/o Kings College	8.16/20.33	-	2.33/9.54	1.56/14.34	1.56/11.75	15.09/58.52
w/o Old Hospital	7.51/18.51	1.01/7.71	-	1.68/16.08	1.61/10.39	14.63/60.02
w/o Shop Facade	6.85/17.08	0.98/6.81	2.15/9.01	-	1.51/10.55	18.63/55.41
w/o St. Mary's Church	6.92/16.04	1.07/5.84	1.79/8.27	1.36/12.74	-	14.28/51.75
w/o Street	6.18/13.33	0.91/5.31	1.96/8.54	1.07/11.52	1.29/8.13	-

Table 3: We experiment with leaving one scene out during training of the Cambridge Landmarks dataset. We observe relatively stable performance with respect to the left out scene. Note that in some cases, we can see that a particular scene hinders performance for other scenes, most notably the model trained without the Street dataset performs much better on all other scenes than the other training setups. We report median translation / median orientation error.

	Fire	Heads	Stairs	Office	Red Kitchen	Chess	Pumpkin
w/o Fire	-	0.13/13.86	0.34/12.38	0.16/6.78	0.21/7.43	0.09/4.66	0.20/4.97
w/o Heads	0.27/9.70	-	0.29/11.04	0.16/6.75	0.22/7.92	0.10/4.90	0.21/5.53
w/o Stairs	0.26/10.94	0.15/13.62	-	0.16/6.35	0.23/7.51	0.10/4.90	0.20/5.19
w/o Office	0.26/10.95	0.14/12.95	0.30/11.70	-	0.22/7.47	0.10/5.14	0.19/5.12
w/o Red Kitchen	0.26/11.01	0.13/14.08	0.30/12.31	0.16/6.55	-	0.13/5.67	0.21/5.00
w/o Chess	0.27/10.92	0.14/12.72	0.28/13.00	0.16/6.54	0.22/7.65	-	0.21/5.66
w/o Pumpkin	0.28/10.69	0.14/12.93	0.34/11.88	0.16/6.52	0.22/7.38	0.11/4.95	-

Table 4: Similar to Table 3 for the 7Scenes dataset. We train a network using only 6 of the scenes to show the stability of our method.

Initialization	Fire	Heads	Stairs	Office	Red Kitchen	Chess	Pumpkin
ImageNet	0.94/82.19	1.19/148.11	0.99/92.36	0.79/44.37	0.86/19.54	0.56/30.91	0.79/48.19
MPSN	0.57/19.35	0.41/20.12	0.70/15.38	0.51/16.95	0.43/13.86	0.54/16.34	0.48/16.6

Table 5: We trained each 7Scenes scene independently for 1 epoch. Training of a new scene typically converges much faster when initializing from a MSPN trained on several other scenes as opposed to ImageNet pretraining. We report median translation / median orientation error.

challenging for a network to train on. Being able to quantitatively show how a scene effects training is a beneficial and unique ability of the proposed multi-scene pose regression framework.

5.4. Quickly Training New Scenes

The backbone of our method is a generalized localization feature extractor. Though the extracted features may have learned some amount of scene-specific information due to the high capacity of the network, the feature extractor acts as a good starting point for training localization of new scenes. There are two ways to approach this, freezing the feature extractor and training only the final scene-specific layer, or fine-tuning the whole network for the new scene. Unfortunately, fine-tuning only the final layer results in very poor performance. As mentioned above, we believe that this is because the network has the capacity to learn scene-specific features for the small number of scenes used in our experiments, meaning the resulting feature extractor is not completely independent of the scenes it was trained on. However, we do find that the resulting models serve as much better initialization for fine-tuning than ImageNet pretraining. Table 5 and Table 6 shows the final performance of networks trained for one epoch on specific scenes. We chose to train for only one epoch to highlight the improvement in performance on the new scene after just seconds of training. In all cases, one or both of position and orientation has much lower error after just one epoch when initialized from a MSPN. Figure 2 shows the full loss curve of 10 epochs for the Fire scene using both initialization methods. Overall we see that the model initialized using our method has nearly converged within only 10 epochs

Initialization	Great Court	Kings College	Old Hospital	Shop Facade	St. Mary's Church	Street
ImageNet	72.58/140.71	30.82/102.77	31.43/119.15	9.81/229.48	19.65/96.58	119.69/73.29
MSPN	73.16/45.71	33.75/11.05	30.23/92.15	10.19/47.54	23.45/29.46	123.81/33.79

Table 6: Similar to Table 5 for Cambridge Landmarks.

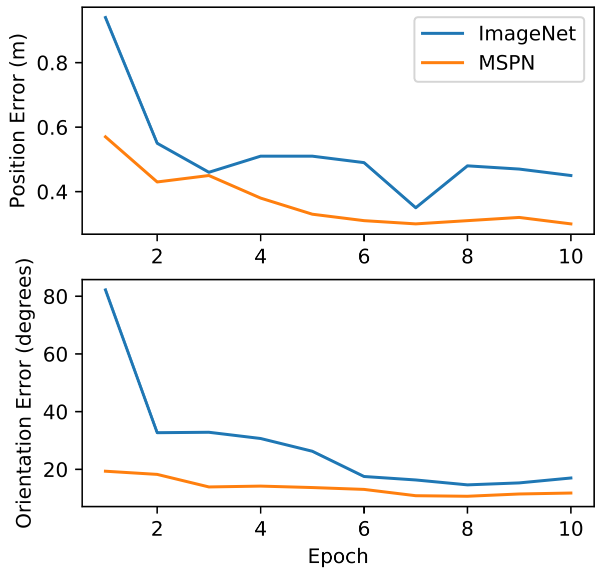

Figure 2: Error in position and orientation on Fire test set as training continues. "ImageNet" signifies the network was initialized from ImageNet pretrained weights. "MSPN" signifies that the network was pretrained for pose regression on other scenes. Fine-tuning a new scene from a network pretrained for pose regression converges much faster than from ImageNet pretraining even if the original scenes are very different from the new scene.

compared to still high error with the ImageNet pretrained models. The faster convergence allows for fine-tuning a new scene with significantly less computation. This means new scenes can be trained in a couple of minutes instead of nearly an hour (5 minutes vs 50 minutes for our experiments).

5.5. Stability Across Datasets

While many applications require accurate localization of the same camera across many different scenes, it is also possible that a method should be agnostic to camera and work across drastically different scenes. To evaluate our method on this scenario, we use all 13 scenes collectively between both datasets to train a single model. Results are shown in Figure 3 for the 7Scenes dataset. We show performance for single scene training, training on all scenes from the 7Scenes datasets, and training with all scenes from both 7Scenes and Cambridge Landmarks using our method. Performance is roughly the same in all cases, with no significant increase in error even when training with all scenes from both datasets. These results are surprising considering the vast difference in appearance and spatial extents between the two datasets. We believe this offers more evidence that the features extracted by our approach are better for the localization task in general.

5.6. Scene Detection

We also verify the ability of the network to infer in which scene the image is located for end-to-end inference. During training, scene prediction is trained along with pose regression using (4). At inference time, the maximum probability scene from the network is used. This allows for completely end-to-end pose regression framework for an arbitrary number of scenes. We compare this to the scene oracle approach in which the scene id is known at inference time. The results are shown in Table 7 and Table 8. We found that scene prediction accuracy is very good in general. Even in the case of the Pumpkin scene where accuracy is only around 85%, the resulting change in pose prediction error compared to the oracle approach is not particularly significant. This is likely because images that are hard to correctly classify are also hard to localize, so the resulting pose will be an outlier in each case. Note that there is a difference in error even when the scene accuracy is 100% because one network is trained with the scene detector and the other without. The additional loss provided by the scene classification results in a different intermediate feature vector.

6. Conclusion

We proposed a multi-scene approach to camera pose regression, Multi-Scene PoseNet, that enables accurate localization across multiple scenes using a single network. Our approach works by learning a set of scene-specific fully connected layers for final pose regression while maintaining a shared feature extractor that is applied across all scenes. This allows a majority of the network to be shared across scenes, and drastically increases the amount of examples that can be used for training. Compared with existing techniques, our approach can support multiple scenes at a fraction of the computational cost of existing meth-

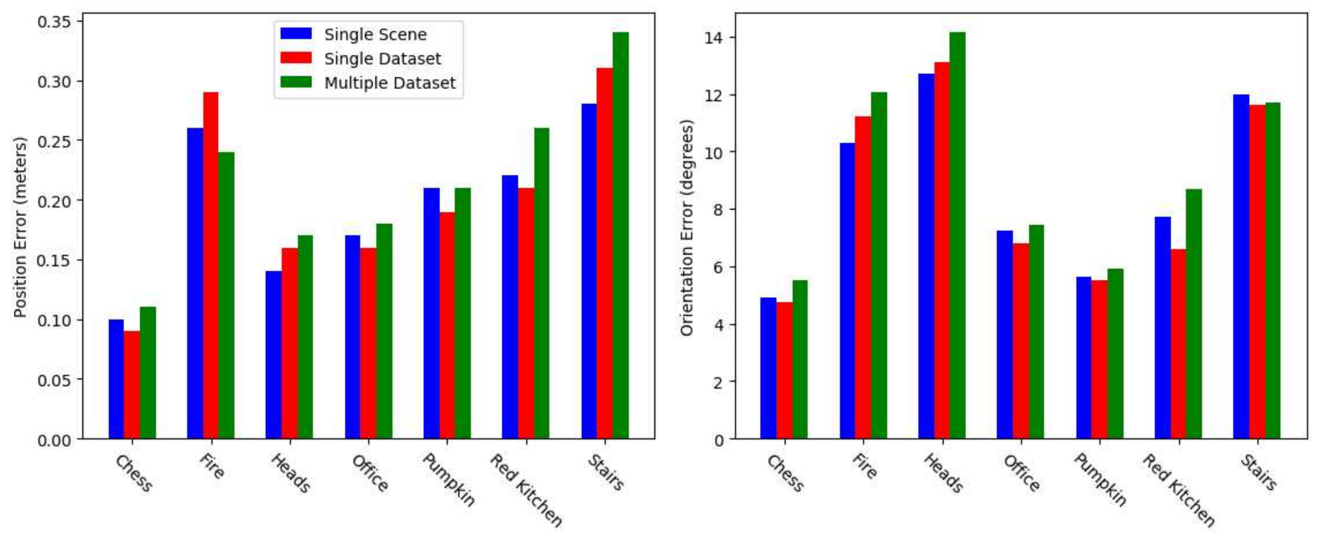

Figure 3: Comparison of final position and orientation error for 7Scenes scenes with various training methods. Single scene means that the network was trained with only images from the specified scene. Note that this is trained with our modified network without the intermediate fully connected layer. Single dataset training means all scenes from 7Scenes were used for training. Similarly, multiple dataset means all scenes from both 7Scenes and Cambridge Landmarks were used. Training across both datasets does not significantly degrade performance even in the worse cases.

	Fire	Heads	Stairs	Office	Red Kitchen	Chess	Pumpkin
Scene Oracle	0.29/11.20	0.16/13.10	0.31/11.63	0.16/6.80	0.21/6.61	0.09/4.76	0.19/5.50
End-to-end	0.30/10.88	0.16/13.33	0.30/13.78	0.17/6.89	0.22/7.19	0.12/6.10	0.23/6.95
Scene Accuracy	0.9895	0.9990	1.000	1.000	1.000	0.977	0.847

Table 7: End-to-end inference for 7Scenes scenes. End-to-end pose error is comparable to the scene oracle case where the scene id is known at inference time.

	Great Court	Kings College	Old Hospital	Shop Facade	St. Mary's Church	Street
Scene Oracle	9.2/10.42	1.73/3.65	2.55/4.05	2.02/7.49	2.67/6.18	26.78/54.22
End-to-end	7.0/11.05	1.22/5.82	2.31/6.19	1.17/11.27	2.2/8.45	25.04/53.25
Scene Accuracy	0.9987	0.9971	1.000	0.9709	0.9981	0.6743

Table 8: End-to-end inference for Cambridge Landmarks scenes. End-to-end pose error is comparable to the scene oracle case where the scene id is known at inference time.

ods, and yet still achieves competitive results on standard benchmark datasets, even achieving state-of-the-art performance on some scenes. We showed that this method is stable across various datasets and number of unique scenes. Finally, we showed how our approach captures a general camera localization feature that can be used to quickly understand new scenes, allowing for faster training of unseen scenes than typical ImageNet pretraining. We believe our method can act as a foundation for many useful extensions to absolute pose regression.

Acknowledgements We gratefully acknowledge the support of the US Air Force Research Laboratory, Sensors Directorate (FA8650-13-D-1547) and the National Science Foundation (IIS-1553116).

References

[1] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *IEEE Conference on Com-*

puter Vision and Pattern Recognition, 2016. 2

[2] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother. Dsac-differentiable ransac for camera localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3

[3] E. Brachmann and C. Rother. Learning less is more - 6d camera localization via 3d surface regression. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4654–4662, June 2018. 3

[4] E. Brachmann and C. Rother. Learning less is more-6d camera localization via 3d surface regression. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1

[5] E. Brachmann and C. Rother. Expert sample consensus applied to camera re-localization. In *International Conference on Computer Vision*, 2019. 3, 4

[6] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz. Geometry-aware learning of maps for camera localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2, 3, 4

[7] M. Ding, Z. Wang, J. Sun, J. Shi, and P. Luo. Camnet: Coarse-to-fine retrieval for camera re-localization. In *International Conference on Computer Vision*, 2019. 2

[8] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381395, 1981. 3

[9] J. Hays and A. A. Efros. Im2gps: estimating geographic information from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 2

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 4

[11] A. Kendall and R. Cipolla. Modelling uncertainty in deep learning for camera relocalization. In *IEEE International Conference on Robotics and Automation*. IEEE, 2016. 1, 2, 3

[12] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1, 3, 4

[13] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *International Conference on Computer Vision*, 2015. 1, 2, 3, 4

[14] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu. Image-based localization using hourglass networks. *CoRR*, abs/1703.07971, 2017. 2

[15] S. Saha, G. Varma, and C. V. Jawahar. Improved visual relocalization by discovering anchor points. In *British Machine Vision Conference 2018, BMVC 2018, Northumbria University, Newcastle, UK, September 3-6, 2018*, page 164, 2018. 3

[16] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2, 3

[17] W. Shang, K. Sohn, D. Almeida, and H. Lee. Understanding and improving convolutional neural networks via concatenated rectified linear units. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2217–2225, New York, New York, USA, 20–22 Jun 2016. PMLR. 4

[18] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 3, 4

[19] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-based localization using lstms for structured feature correlation. In *International Conference on Computer Vision*, 2017. 2

[20] S. Workman, R. Souvenir, and N. Jacobs. Wide-area image geolocalization with aerial reference imagery. In *International Conference on Computer Vision*, 2015. 2

[21] L. Yang, Z. Bai, C. Tang, H. Li, Y. Furukawa, and P. Tan. Sanet: Scene agnostic network for camera localization. In *International Conference on Computer Vision*, 2019. 2