DTU Compute

SELF-SUPERVISED LEARNING FOR IMAGE-BASED GEOLOCATION

Master Thesis

Ke Yang - s212495

Ke Yang

Signature

11.02.2024

Date

Acknowledgements

Ke Yang, MSc of Computer Science and Engineering, DTU

Contents

Abstract

This study delves into the utilization of self-supervised learning methodologies in the realm of image geolocation tasks, while also exploring the viability of addressing it through regression paradigms. Initially, we introduce a self-supervised learning model based on Masked Image Modeling and conduct a comprehensive analysis of its training procedure and efficacy during the pretraining phase. Our findings reveal that the MAE model boasts advantages in terms of training speed and exhibits proficiency in accurately reconstructing partially occluded images. Subsequently, leveraging the pretraining weights derived from the MAE model as initial parameters, we fine-tune a Visual Transformer (ViT) model tailored for the geolocation regression task, aiming to tackle the geolocation problem within a regression framework. Experimental results manifest that the integration of pretraining weights from the MAE model amplifies both the accuracy and convergence speed of the ViT model in the geolocation regression task. Furthermore, we scrutinize the ramifications of various pretraining and fine-tuning strategies on model performance, discovering that fine-tuning with a dataset related, albeit not entirely identical, to the pretraining data can yield further enhancements in model efficacy. In summary, this study undertakes a comprehensive examination of image geolocation tasks using self-supervised learning techniques, alongside an exploration of regression methodologies within this domain. You will be able to see the source code at Github: https://github.com/keyang1211/Image-geolocation

Keywords: Image, Geolocation, Pretraining, Self-supervised learning, ViT.

1 Introduction

The task of image geolocation stands as a pivotal challenge within the domains of computer vision and cartography, encapsulating multifaceted complexities. Various factors, such as geographical diversity and the heterogeneous nature of capture environments, exert profound influence on the intricacies of localization. For instance, images featuring prominent landmarks or architectural structures inherently entail richer geographical cues compared to those depicting natural landscapes like beaches, grasslands, or forests, thereby rendering their prediction of capture locations relatively more precise and tractable. In our case, utilizing the training set MP-16 and the test set IM2GPS3K as examples, approximately half of the urban photos, such as streetscapes and landmarks, are included. Additionally, about one-quarter consists of natural landscapes. The remaining portion comprises indoor images, foods, or humans with fewer geographical features included. Furthermore, even within identical geographical settings, temporal variations, seasonal dynamics, and divergent stylistic attributes of capture devices can engender substantial disparities, further amplifying the intricacy of the localization task.

The task of image geolocation has various potential applications in practical scenarios. One application is in social media analysis, where it can be used to analyze photos uploaded by users and determine their capture locations, thereby providing more accurate geographical tags and location-relevant content recommendations. Another application is in the tourism industry, where photo geolocation can assist tourists in discovering nearby attractions, restaurants, or activities, offering personalized travel suggestions. Additionally, photo geolocation can be utilized in urban planning and geographic information systems (GIS) to help determine population density, activity hot-spots, and environmental changes in specific areas. These applications demonstrate the significant role of photo geolocation in addressing various practical problems and enhancing the accuracy and intelligence of location awareness.

In previous studies, this particular problem has consistently been framed as a classification task. In works such as [1, 2, 3], similar approaches were employed, whereby the Earth's surface was partitioned into cells of varying sizes, each containing an equal number of images based on the training set. This transformation effectively converted the task into a multi-classification problem. However, we argue that for multi-class classification tasks, utilizing the distribution of the training set and employing cross-entropy as the loss function, regression tasks offer a more favorable solution. This assertion stems from the fact that in classification tasks, the loss becomes insensitive to the distance between the predicted class and the correct class in cases of classification errors. In contrast, regression tasks directly utilize the distance between the predicted coordinates and the true coordinates as the loss, providing a more intuitive measure that requires fewer computational resources.

Furthermore, considering the distribution of the training set, as illustrated in Figures 1.1 and 1.2, wherein one depicts the distribution of the training set MP-16 on a map, and the other shows the distribution of the test set IM2GPS3K, we observe that their distributions are roughly similar. Moreover, the majority of the test set data is concentrated in regions such as Europe, North America, and East Asia, which are densely populated with training set data. Consequently, for methods that segment the Earth's surface into categories based on the distribution of the training set [1, 2, 3], the categories occupied by the test set data tend to correspond to regions with smaller surface areas. Consequently, using classification-based approaches relying on the segmentation of the Earth's surface according to the training set's distribution may result in optimized cross-entropy loss, leading to comparatively small average distance errors and higher precision within smaller ranges. However, this classification approach also poses limitations, particularly concerning issues related to transferability and generalization.

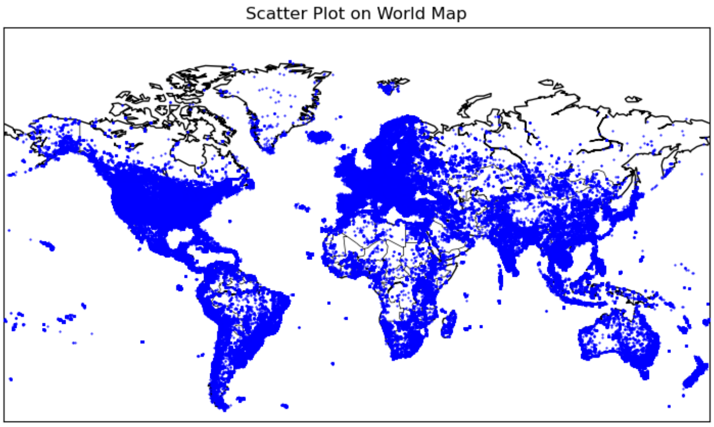


Figure 1.1: **Training set distribution**,This is the distribution of the 4.72 million data points from the training set on a world map, with each point representing an image.

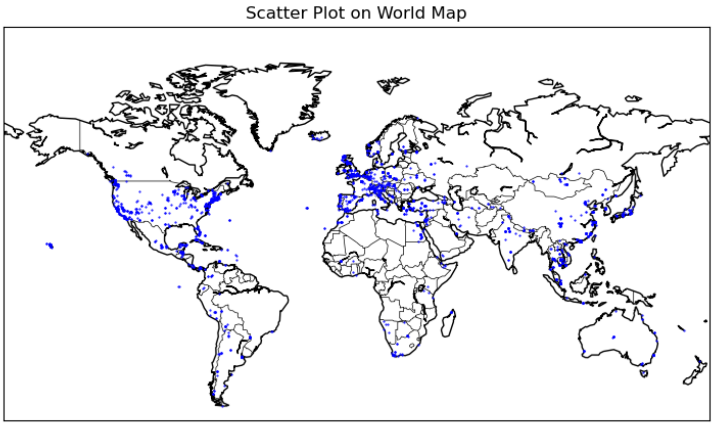


Figure 1.2: **Testing set distribution**This is the distribution of the 3k data points from the test set on a world map, with each point representing an image.

In this study, we attempted image geolocation using regression methods with the same dataset as the aforementioned approaches. We explored the utilization of self-supervised learning pre-trained models and observed their performance. To this end, we established two baseline models: one based on a regression model utilizing ResNet [4], and another based on a regression model utilizing Vision Transformer [5]. Additionally, we implemented a self-supervised pre-training model, Masked Autoencoder, which automatically masks and restores images. Furthermore, we incorporated a regression component into this model, leveraging a portion of its pre-trained weights (effectively mirroring the structure of the regression model based on Vision Transformer [5]). Subsequently, we trained these models for the task of image geolocation. Following the completion of training, we recorded the average distance errors of all regression models and compared their prediction accuracies within different ranges (regions, countries, and continents). We also compared these results with those obtained from classification models and conducted subsequent analyses accordingly.

2 Related Works

2.1 Image Geolocation

In the work by [6], the image geolocation task is categorized into two scales: Closed-Domain Image Geolocation (CDIG) and Open-Domain Image Geolocation (ODIG). CDIG involves predicting the geographical location of images within a specific region, such as a city, a country, or certain defined areas. Early approaches to similar tasks predominantly employed specific feature-based traditional methods for image localization within particular natural environments, such as beaches [7], mountains [8, 9], deserts [10], or urban settings [11, 12]. Later advancements in deep learning within computer vision expanded the scope of image geolocation beyond street-level ([13]), encompassing broader geographical scales ([14]). While such methods often demonstrate remarkable performance on specific datasets and tasks, their limitation lies in the diminished transfer learning value across other domains or tasks. Their targeted nature and relatively narrow applicability restrict their widespread real-world use. Therefore, despite their proficiency in specific environments or problems, their generalizability and transferability are comparatively weaker, limiting their practicality and adaptability in diverse scenarios.

ODIG focuses on a global scale, learning more diverse features, exhibiting enhanced versatility, and transferability. Initially, addressing such tasks involved retrieval methods. For instance, the pioneering attempt at planet-scale image geolocation is attributed to IM2GPS [15], followed by subsequent work by [16] and [17]. Most of these retrieval methods involve finding similar examples from a large reference dataset to predict the location, thereby consuming significant computational and storage resources.

However, in 2016, [1] proposed the use of convolutional neural networks for classification tasks to perform geospatial localization of images. This work can be considered as the first to transform this task into a classification problem. Additionally, [18] highlighted the difficulty in directly predicting coordinates for such tasks. Subsequent works primarily approached this task as a classification problem.

Drawing inspiration from planet's Adaptive Partitioning method in [2], the approach involved partitioning based on the density distribution of training data into different area segments. It also introduced scene classification, training distinct CNN-based geographical localization classifiers for various scenes. This work effectively subdivided the ODIG task into several CDIG tasks. However, in this paper, we attempted to approach this task as a regression problem. We directly predict the latitude and longitude coordinates of the image capture location from the model.

Furthermore, with the evolution of Transformers in the visual domain, there emerged classifiers based on Vision Transformers [5] for image geolocation tasks [3][6]. Some studies leveraged modalities beyond images, yielding impressive results. For instance, [19] combined temporal data with image data, effectively learning features related to time aspects like sunlight intensity and seasons, resulting in promising outcomes. Additionally, in [6], the creation of textual descriptions for images achieved impressive performance in zero-shot classification.

2.2 Self-supervised Learning

Self-supervised learning (SSL) is a paradigm in machine learning where the model is trained using labels generated from the data itself rather than relying on external annotations. This approach leverages the inherent structure or information within the data to formulate learning tasks. SSL has gained significant attention in recent years due to its potential to address the challenges associated with limited labeled data and the high cost of manual annotation.

In the context of image geolocation and related computer vision tasks, self-supervised learning methods have been employed to learn meaningful representations from images without the need for explicit geographic annotations. Several techniques fall under the umbrella of self-supervised learning, each introducing innovative ways to design pretext tasks that encourage the model to capture relevant features and contextual information.

In [20], self-supervised learning is broadly categorized into four types from the perspective of the overall structure. The first category is ”The Deep Metric Learning Family,” exemplified by methods like SimCLR[21]. It is centered around optimizing contrastive loss[22], aiming to minimize the distance between representations of positive samples and maximize the distance from negative samples. The second category is ”The Self-Distillation Family,” illustrated by SimSIAM[23] and DINO[24]. This family involves placing two different views of an image into two separate encoders, mapping the output of one student network to the output of another teacher network through a predictor. The optimization is based on minimizing the discrepancy between their outputs. The third category is ”The Canonical Correlation Analysis Family,” represented by VICReg[25], which focuses on optimizing the correlation between representations from two views. The fourth category includes types like Masked Image Modeling, which is also the primary focus of this paper.

2.2.1 Masked Image Modeling

”Masked Image Modeling” refers to a self-supervised learning method in which a model learns information about an image by processing masked or obscured portions of the image. In this approach, certain parts of the image are covered or masked, and the model’s task is to learn representations of the image content by processing these masked regions. In 2016, [26] introduced a pre-training approach where a significant portion of the image is initially masked, followed by a reconstruction process using an encoder-decoder structure. Meanwhile, [27] extended the masked language modeling method of BERT to the Vision Transformer architecture. Building on the use of pixelwise reconstruction loss, [28] optimized the workflow of self-supervised learning, achieving state-of-the-art ImageNet 1k performance among competitors that don’t utilize additional data. Inspired by the work of [28], this paper aims to implement pre-training of Masked Autoencoder in the field of image geolocation.

3 Architectures

3.1 Baseline models

To thoroughly evaluate the performance of models pre-trained through self-supervised learning in experiments, as well as to assess the impact of pre-training weights on the models, two baseline models are established for comparison in this study.

3.1.1 Resnet

ResNet-152[4] is a deep residual neural network that has achieved significant success in tasks such as image classification and other computer vision tasks. It is an extended version within the ResNet series, boasting 152 layers in depth and utilizing residual connections to address the challenges of gradient vanishing during training of deep networks.

In essence, ResNet-152 is structured with multiple residual blocks, they are all bottleneck blocks ,see in figure 3.1,3.2.In ResNets with fewer layers (fewer than 50), basic blocks are employed exclusively. The bottleneck residual block serves as the primary building block in ResNet-152, comprising a 1x1 convolutional layer for dimensionality reduction, followed by a 3x3 convolutional layer for feature extraction, and finally another 1x1 convolutional layer for dimensionality restoration. This design significantly reduces the number of parameters and computational complexity while enhancing the network's performance and generalization ability.

Table 3.1: ResNet Architecture

Layer Name	Output Size	Configuration
conv1	112×112	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2 1×1, 64 3×3, 64 1×1, 256 ×3
conv3	28×28	1×1, 128 3×3, 128 1×1, 512 ×8
conv4	14×14	1×1, 256 3×3, 256 1×1, 1024 ×36
conv5	7×7	1×1, 512 3×3, 512 1×1, 2048 ×3
Ave pooling	1×1	average pool, 1000-d fc, softmax

The overall architecture of ResNet-152 can be presented through a table3.1, detailing the name, output size, and configuration of each layer. For instance, in the conv1 layer, the output size is 112x112, configured with a 7x7 convolutional kernel, 64 output channels, and a stride of 2 for convolutional operation. Additionally, the structure of residual connections can be illustrated through an image, depicting the internal composition of the bottleneck residual block, including input, output, skip connections, and other components. Simultaneously, the first block of each layer is a downsampling block, as shown in Figure 3.2. Layers marked with '/2' indicate convolutional layers that reduce the data of each channel to 1/4 of its original size, achieved by appropriately setting padding (2) and stride to 2. Finally, the extracted features are fed into subsequent fully connected layers after applying average pooling layer.

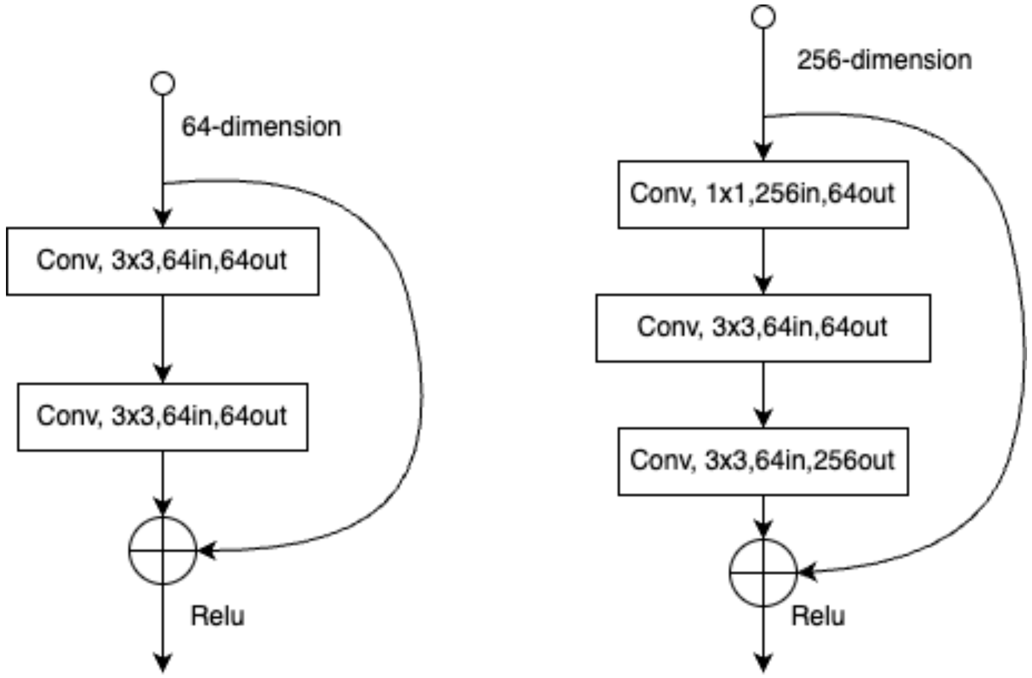


Figure 3.1: **Blocks in Resnet**,On the left are basic blocks, and on the right are bottleneck blocks.

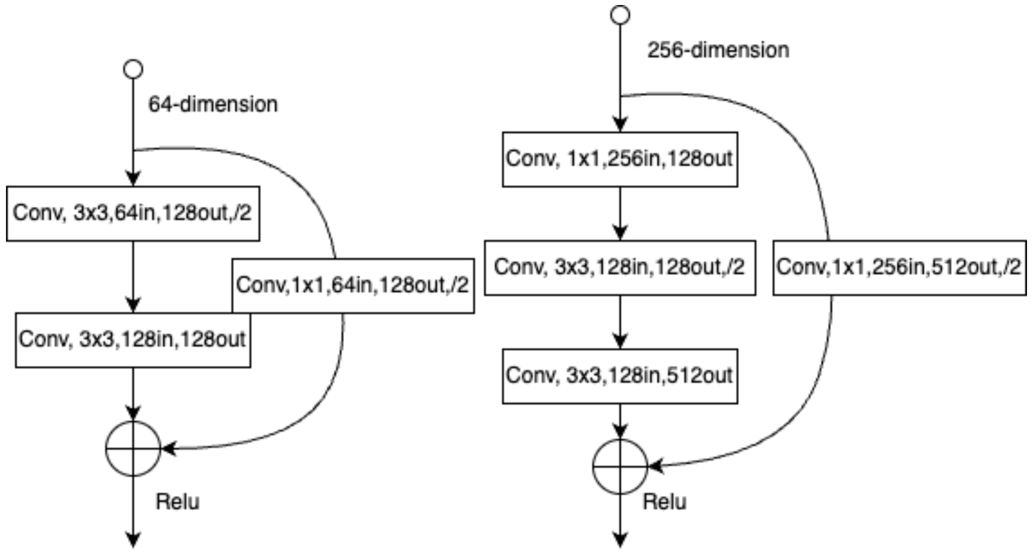


Figure 3.2: **Downsampling Blocks in Resnet**,On the left are basic blocks, and on the right are bottleneck blocks.

Within the three fully connected layers, the Rectified Linear Unit (ReLU) activation function is used to introduce non-linearity. In the final layer's output, two values are produced, and the Tanh function is applied to yield values within the range of [-1, 1]. This facilitates mapping to coordinate values during the training process, with latitude ranging from -90 to 90 degrees and longitude from -180 to 180 degrees.

3.1.2 Vison Transformer

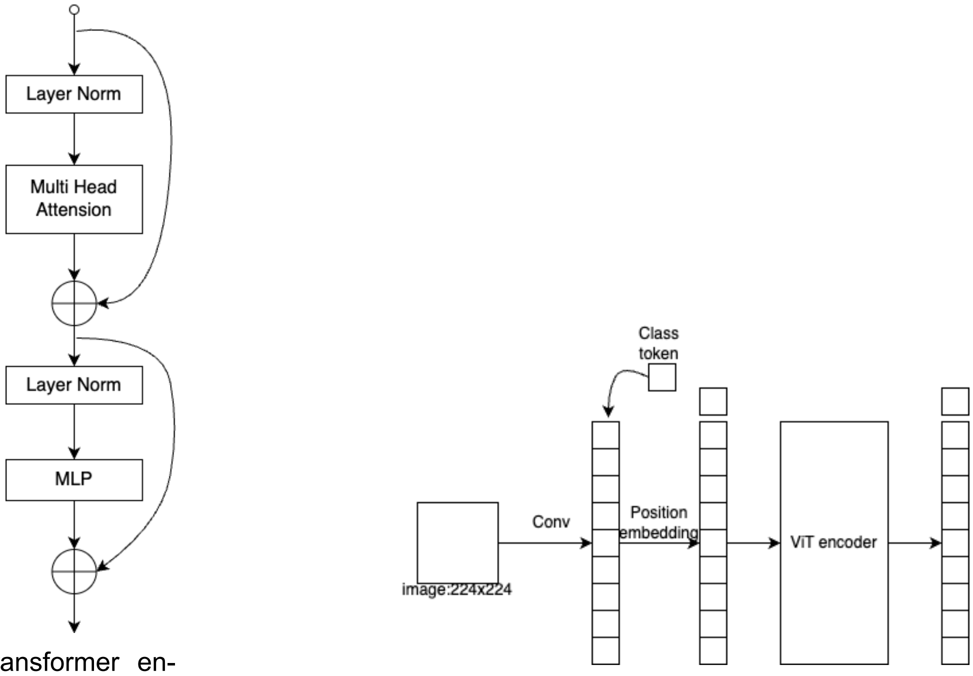
The baseline model ViT (Vision Transformer) is an image classification model based on the self-attention mechanism. It divides the image into small patches using convolution and maps these patches into a sequence.

The fundamental component of ViT is the Transformer block, comprising multiple attention heads, each responsible for learning different features. In the left image 3.3a, we observe the structure of a Transformer block, incorporating Layer Normalization, self-attention layers, and fully connected layers. The self-attention layers facilitate interactions across different positions in the sequence, directing attention to significant image patches to capture global information. Additionally, two residual connections are included. Their purpose aligns with that of residual connections in ResNet, aiming to mitigate gradient explosion and vanishing during training of deep networks.

The entire feature extraction process in ViT, illustrated in the image on the right (see Figure 3.3b), unfolds as follows. The ViT encoder comprises 12 blocks (see Figure 3.3a), each consisting of multiple multi-head self-attention layers with 12 heads. Additionally, the size of each patch is 16×16, and the embedding dimension of each patch is 768. The maximum dimension in the fully connected layers is 3072.

Initially, the input image is segmented into uniformly sized image patches, which are then linearly transformed and mapped into a sequence. Subsequently, the sequence of image patches undergoes a series of Transformer blocks, each containing multiple attention heads to model relationships between image patches. Ultimately, features from the sequence are integrated through average pooling or similar methods to obtain the image representation.

Subsequently, the class token is extracted from the output, serving as input to fully connected layers with a structure similar to the ResNet architecture described earlier. This final layer produces two values within the range of [-1, 1]. This facilitates mapping to coordinate values during the training process, with latitude ranging from -90 to 90 degrees and longitude from -180 to 180 degrees.



(a) Vision Transformer encoder block

(b) Vision transformer

Figure 3.3: **Vison Transformer Architecture**On the left is a block representing the components of the Vision Transformer. On the right is the entire process of feature extraction by the Vision Transformer.

3.2 Masked autoencoder

3.2.1 Pretraining

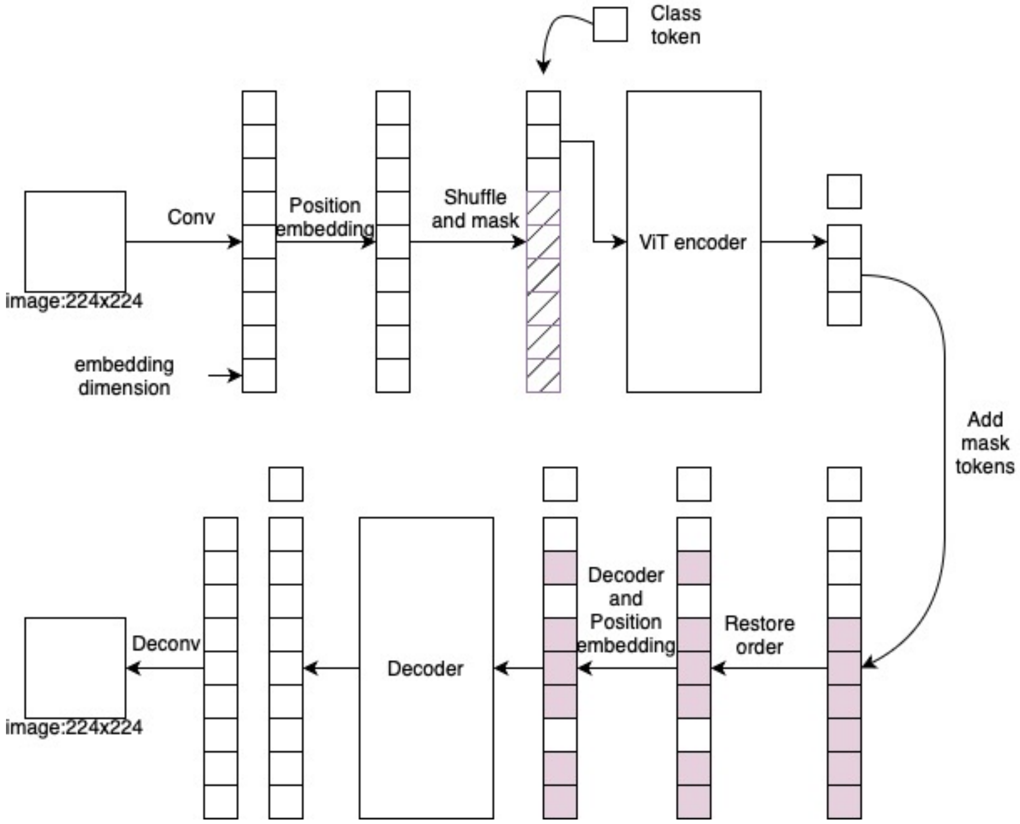


Figure 3.4: **MAE**,This is the specific structure employed for MAE (Masked Autoencoder) pre-training in the study. Upon input of an image into the model, a random masking process is applied to part of the input. The remaining unmasked portion is then fed into the encoder. The output from the encoder is padded with blank tokens to complete the masked regions and is then input into the decoder. The decoder produces an output, which is further processed through deconvolution to reconstruct an image tensor of shape [3, 224, 224].

The specific architecture used for pre-training in this article is inspired by the MAE model. Inspired by [5], we employ convolutional layers for making patches and embedding the hidden dimensions (768). Compared to manually partitioning and concatenating channels of image tensors into patches as described in [28], employing convolutional layers for making patches and embedding the hidden dimensions would offer several advantages. Firstly, it is more convenient, to condense two steps into one. Secondly, utilizing convolutional layers reduces the parameter count. Additionally, employing convolutional layers allows for a more comprehensive learning of the interdependencies among different channels.Complete positional embedding is performed at this stage. Subsequently, all patches are randomly shuffled, and a certain proportion of the last-ranked patches are removed to achieve random masking. Class tokens are added to the remaining patches, and position embedding is applied to the class token. They are then input into the ViT encoder.By doing so, the input of pruned image data into a more complex and deep ViT encoder can significantly reduce the time and computational costs required for pre-training.

The ViT encoder used in this context closely resembles the ViT model in the baseline, consisting of 12 blocks (refer to Figure 3.3a). Each block comprises multiple multi-head self-attention layers with 12 heads. Additionally, the size of each patch is 16×16, and the embedding dimension of each patch is 768. The maximum dimension in the fully connected layers is 3072. However, it lacks the final fully connected layer and omits the extraction of the class token, with all outputs retained. A blank mask token is appended after the encoder's output, and the original order is restored. A linear layer is utilized to reduce the hidden dimensions to 384, followed by the addition of position embedding. This processed data is then fed into the decoder.

The number of blocks in the decoder is one-third of the encoder, and the hidden dimensions of the MLP in each block are half of those in the encoder. As a result, inputting data with complete length (number of blocks) into a relatively lightweight decoder compared to the encoder can reduce training time and computational costs.The class token is removed from the decoder's output, and a transposed convolutional layer is used to restore it to an image tensor of shape [3, 224, 224].

3.2.2 Vison Transformer and regression

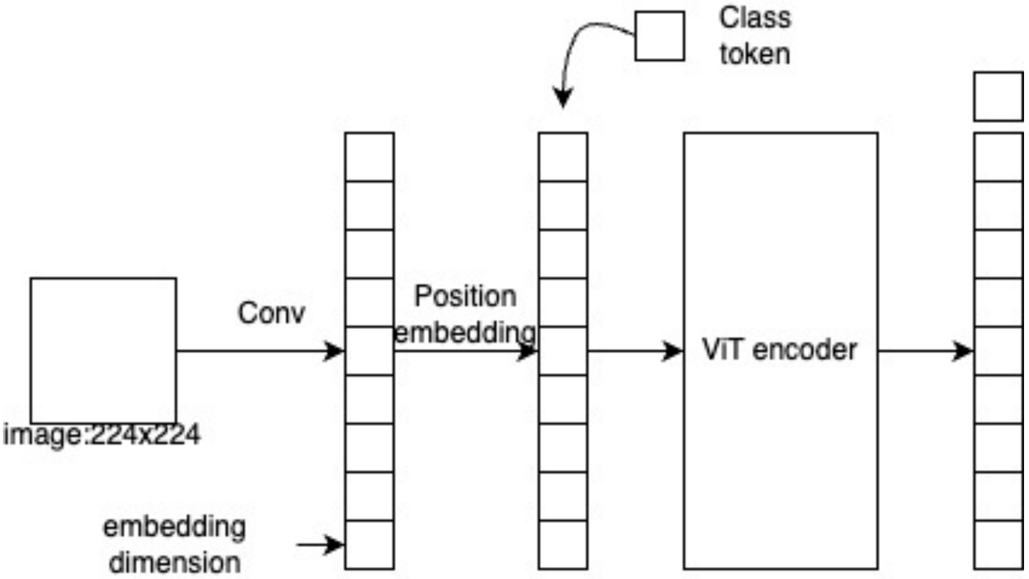


Figure 3.5: **Featrue extractor**,This is the feature extraction component to be used in downstream tasks after pre-training. Unlike the complete process of self-supervised learning pre-training, after patchification, embedding, and adding position embedding and class tokens, the input is fed directly into the ViT encoder without performing shuffle and mask operations.

Moreover, due to the structure of the ViT encoder, which mainly consists of multi-head self-attention layers, as long as the embedding dimensions remain the same, we can reuse the pre-trained parameters in downstream tasks without concerning the number of patches or blocks. Therefore, when utilizing the upper part of the model for feature extraction as depicted in Figure 3.5, one can simply set the mask ratio to 0 to perform feature extraction on the entire image.

After completing pre-training with the MAE model, the pre-trained weights are utilized for an image geolocation task. The employed structure corresponds to the portion of the MAE pre-training architecture discussed earlier (see Figure 3.5), up to the stage just before adding the mask token. It is essential to set the mask ratio to 0 at this point. Subsequent to the encoder's output, the class token is extracted and fed into the final fully connected layer, mirroring the two baseline models. Within the three fully connected layers, the Rectified Linear Unit (ReLU) activation function is applied to introduce non-linearity. In the output of the final layer, two values are generated, and the Tanh function is employed to confine the values within the range of [-1, 1].

4 Expriments settings

Here are the details of the datasets used in the experimental process and some settings for model training:

4.1 Datasets

The datasets used in our study consist of real-world images paired with their corresponding geographical coordinates as labels. The training set, MediaEval Placing Task 2016 dataset(MP-16) [29], and the validation set, YFCC25600 [30], are both sourced from Google Flickr.

4.1.1 Training set

The MediaEval Placing Task 2016 dataset(MP-16)[29] training set,which is a subset of the Yahoo Flickr Creative Commons 100 Million dataset (YFCC100M), comprises 4.72 million images, each associated with its respective coordinates. This includes images of nature, urban scenes, and even some indoor settings, as well as images featuring objects such as food and portraits, with relatively limited geographic information. The specific distribution is presented in Table 4.1.As depicted in Fig. 4.1, the first three images represent landmark buildings from various locations, situated in the area of highest image density, making them highly representative and containing the most geographic information. The images in the second row showcase natural landscapes, which, despite some differences between locations, contain significantly less geographic information. Finally, the images in the third row contain almost no geographic information, making it nearly impossible to predict their geographical locations based solely on image features.

Category	mp-16	Im2GPS3k
urban	1969939	1607
natural	1127576	845
indoor/human/...	1626180	545

Table 4.1: Sample Counts of Different Categories in the Datasets

4.1.2 Validation set

The YFCC25600 validation set consists of 25,600 images along with their coordinates. As these datasets originate from Google Flickr, the images have no restrictions on scenes or targets, encompassing valuable information such as landscapes and architecture. However, they also contain a small amount of noise, including portraits and food images.

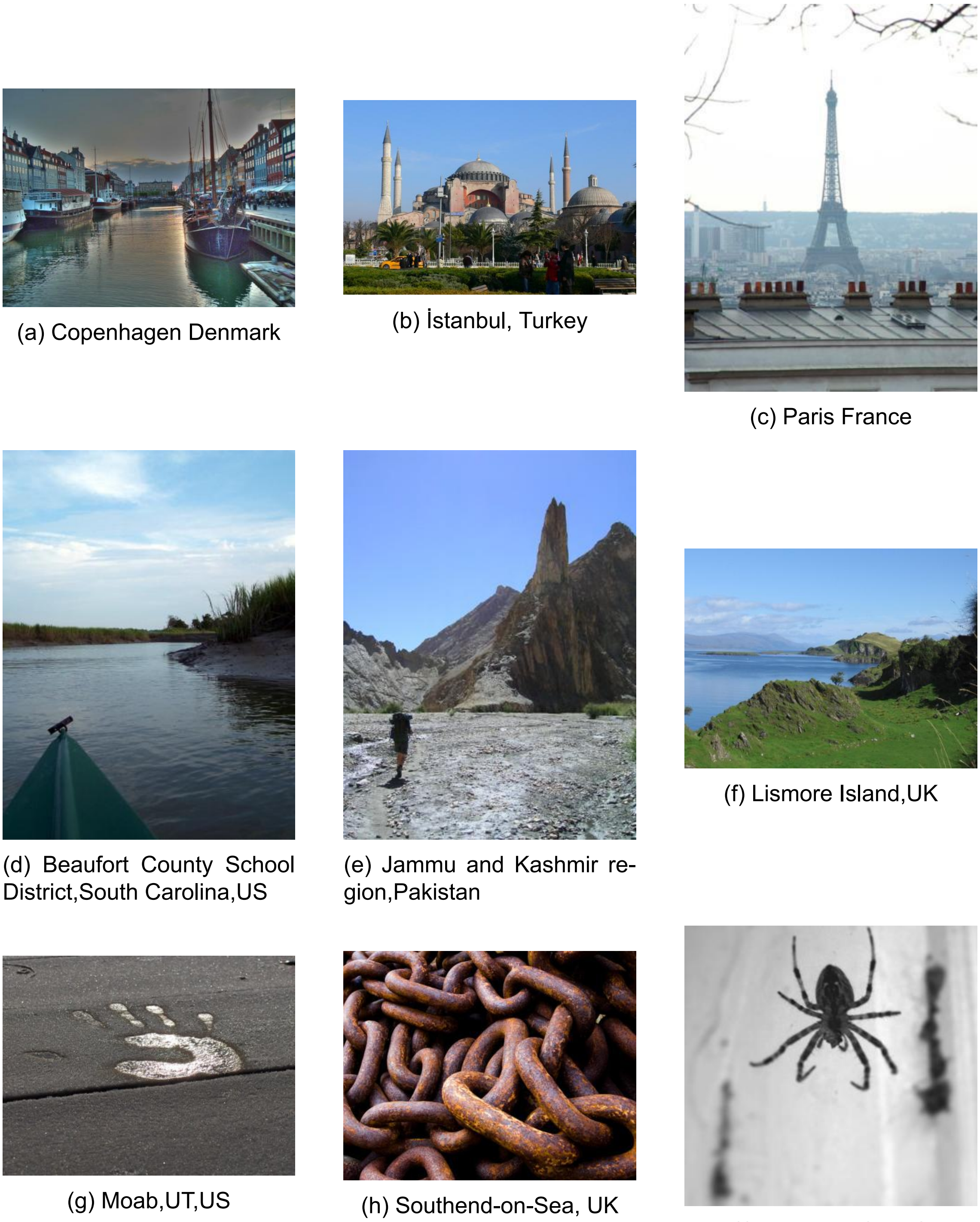
4.1.3 Test set

Furthermore, the performance evaluation of all models is based on their performance on the Im2GPS[31] and Im2GPS3k[32] test sets. Im2GPS3k includes the content of Im2GPS, where Im2GPS consists of carefully selected high-value targets such as landmarks and scenic spots.For example, the images in the first row of Fig. 4.1 have been selected for their higher density of geographic information.

4.2 Evaluation Metrics

4.2.1 SSL pretraining

Due to the fact that both the input and output of our self-supervised learning pretraining model are images with tensor shapes of [3, 224, 224], the loss function for the self-



(a) Copenhagen Denmark

(b) İstanbul, Turkey

(c) Paris France

(d) Beaufort County School District,South Carolina,US

(e) Jammu and Kashmir region,Pakistan

(f) Lismore Island,UK

(g) Moab,UT,US

(h) Southend-on-Sea, UK

(i) Los Altos,CA,US

Figure 4.1: Nine images from mp-16

supervised learning pretraining model is defined as the Mean Squared Error (MSE) at each position. Subsequently, the average loss for each patch is computed, followed by summation of the losses across all patches. This aggregated value serves as the total loss.

4.2.2 Image geolocation regression

For the image geolocation regression task, considering the presence of specific latitude-longitude coordinates in the original training, validation, and test datasets, as well as the model's final output predictions also being specific latitude-longitude coordinates, the loss function for this regression task is defined as the mean error distance. In this context, the distance computation adopts the Great Circle Distance (GCD) method.

The spherical distance between two latitude-longitude coordinate points is computed as follows:

$$\text{GCD} = R \cdot c$$

where R is the Earth's radius (in kilometers), and c is calculated using the Haversine formula:

$$c = 2 \cdot \text{atan2}\left(\sqrt{a}, \sqrt{1-a}\right)$$

with a defined as:

$$a = \text{sin}^2\left(\frac{\Delta\text{lat}}{2}\right) + \text{cos}(\text{lat1}) \cdot \text{cos}(\text{lat2}) \cdot \text{sin}^2\left(\frac{\Delta\text{lon}}{2}\right)$$

Here, Δlat and Δlon represent the differences in latitude and longitude, lat1 and lat2 represent the latitude values of the two coordinate points we want to calculate the distance, respectively, converted from degrees to radians.

$$\text{Loss} = \frac{1}{N}\sum_{i=1}^{N} \text{GCD}(y_i, \hat{y}_i)$$

where N represents the batch size, y_i denote the true latitude and longitude of the i-th sample, and $\hat{\text{y}}_i$ represent the predicted latitude and longitude by the model, respectively.

This formulation reflects the objective of minimizing the average error distance between the predicted and ground truth coordinates, thereby optimizing the model's performance in accurately estimating geographical locations from images.

4.3 Baseline settings

To compare the performance of various deep learning methods, we conducted multiple random guesses based on the distribution of the training set. Specifically, we clustered the coordinates of the training set, randomly generated points according to the mean and variance of each cluster, and compared them with the coordinates of the test set to obtain the random guess values as references.

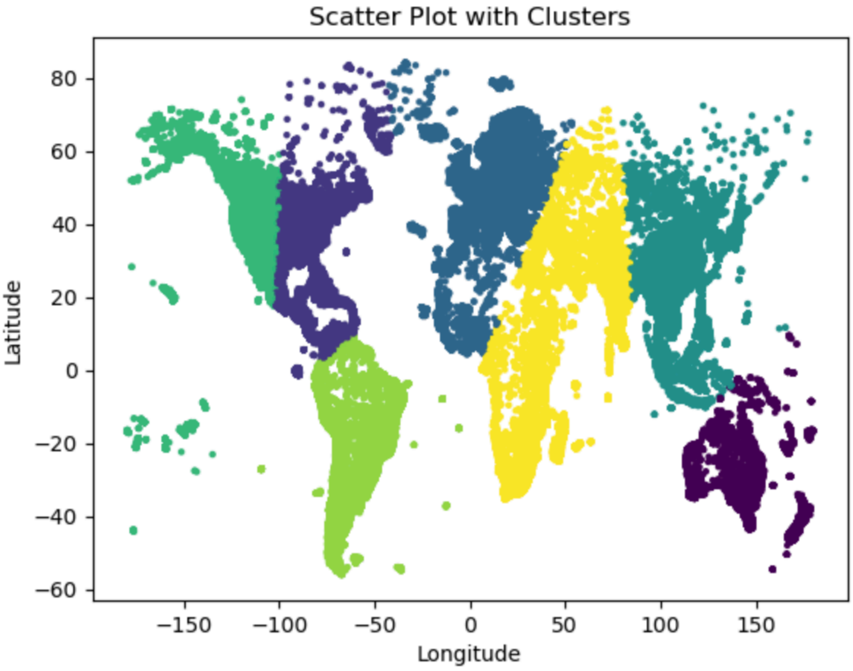


Figure 4.2: **Clustering of training set**,This is clustering based on the training set, and through the clustering results, a class is randomly selected. Then, a point is randomly generated based on the mean and variance of this class. This process is repeated to generate random guesses equal to the number of the test set.

For both ResNet and ViT, the two baseline models, we used similar training parameters. Neither model had pre-trained weights, and default PyTorch initialization was applied. The batch size was set to 64. We utilized the AdamW optimizer with an initial learning rate of 0.0001. A StepLR scheduler was employed, reducing the learning rate by a factor of 0.1 every three epochs. Additionally, we set the weight decay to 0.01 and applied gradient clipping of 0.1 to prevent overfitting and mitigate gradient explosions.

4.4 Masked autoencoder

During the pre-training phase of MAE, a batch size of 64 was also employed. The AdamW optimizer was utilized with an initial learning rate of 0.0001. However, a MultiStepLR scheduler was used with a gamma parameter set to 0.5, aiming for a slower reduction in the learning rate. Similar to the baseline models, weight decay was set to 0.01, and gradient clipping of 0.1 was applied to prevent overfitting and mitigate gradient explosions. Additionally, each image has a random mask ratio of 75

For the downstream task of image geolocation, we maintained consistency with the ViT configuration outlined above for the sake of comparison. The batch size was fixed at 64. We employed the AdamW optimizer with an initial learning rate of 0.0001. To regulate the learning rate, we utilized a StepLR scheduler, decrementing the learning rate by a factor of 0.1 every three epochs. Moreover, we set the weight decay to 0.01 and implemented gradient clipping of 0.1 to counteract overfitting and alleviate gradient explosions.

All the aforementioned models were trained using the training dataset. During the training process, the performance of the models on the validation dataset, primarily measured by the average loss, was monitored every 8000 steps. Upon completion of training, the model checkpoint with the best performance on the validation dataset was selected. This checkpoint was then loaded to evaluate the model using the test dataset, yielding the final test results.

We tested all geographical location prediction models using two separate test sets, obtaining average error distances for each test set. Additionally, we evaluated prediction accuracy on different geographical scales: 200 kilometers, 750 kilometers, and 2500 kilometers, representing city/region, country, and continent, respectively. The experimental data presented in the following sections were derived from these evaluations. Furthermore, we recorded various training-related metrics for each model to facilitate analysis.

5 Expriments results and analysis

5.1 SSL Model

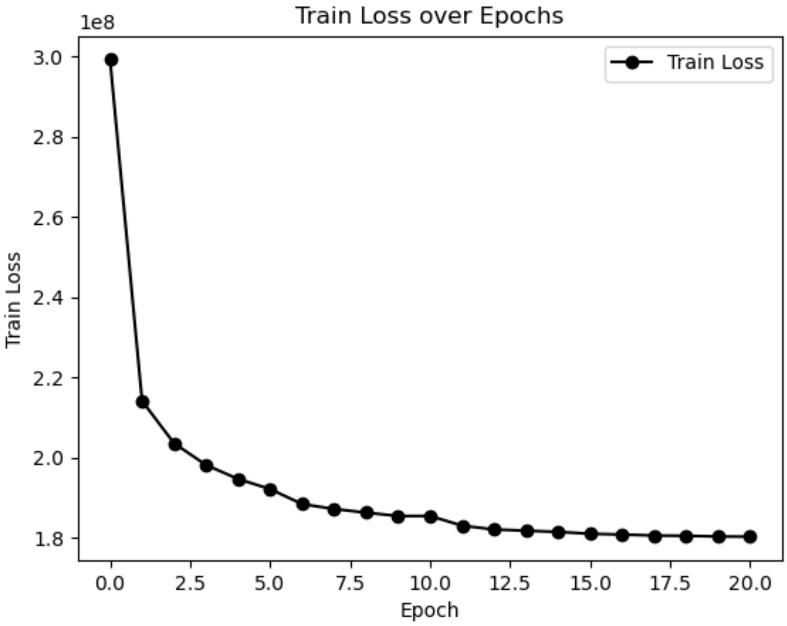
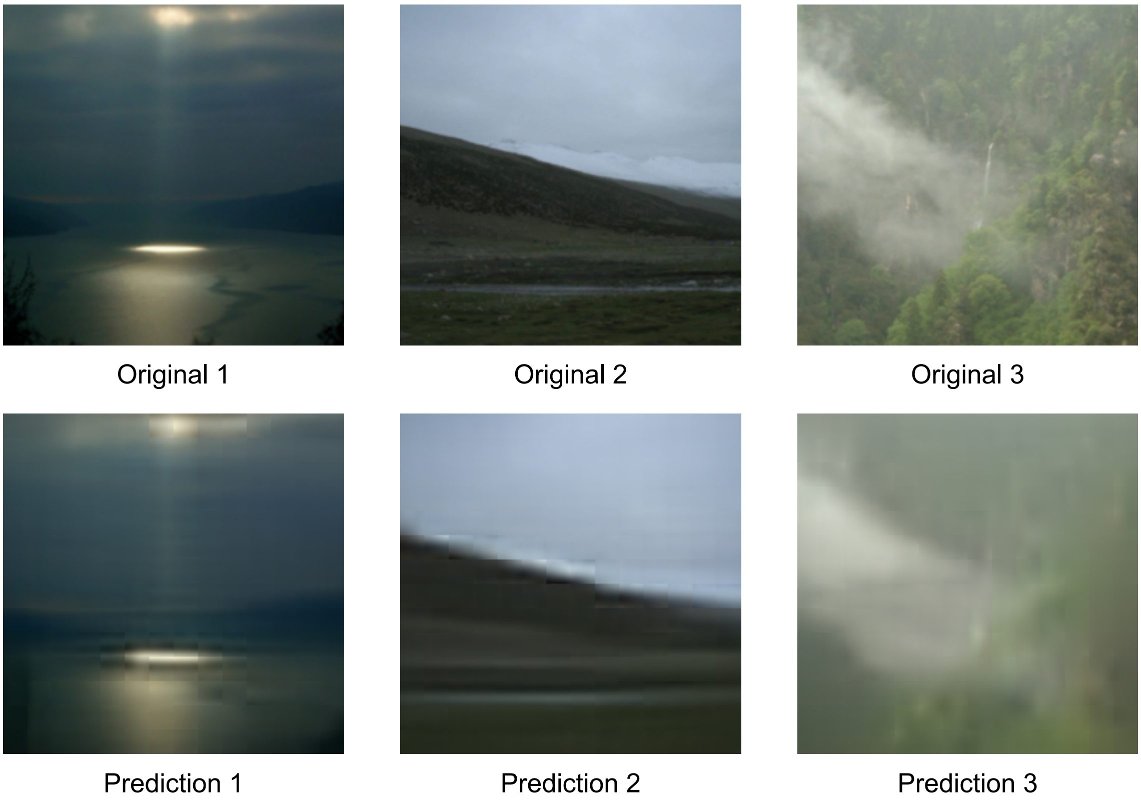


Figure 5.1: **MAE pretraining**,This is the training loss curve for the MAE pre-training, where the vertical axis represents the sum of the losses over the entire training set, and the horizontal axis represents the training epoch.

As shown in Figure 5.1, the training process of the self-supervised learning model MAE is demonstrated. Due to time and resource constraints, we ran 20 epochs, and the training loss has stabilized. As mentioned earlier, we input the partially masked data into an encoder of the same size as the ViT model, and then pass the complete data with filled-in masks into a decoder one-third the size of the ViT model. Overall, the MAE pre-training model has approximately 1.3 times the total parameter count of the ViT model, yet trains faster. On an NVIDIA A100 GPU, with the same settings, the MAE pre-training model can train 10 epochs within 24 hours, while the ViT model can only train 7 epochs. However, it's worth considering that the loss function for training the ViT model is slightly more complex. Overall, the training speed has met our expectations.



Original 1 Original 2 Original 3

Prediction 1 Prediction 2 Prediction 3

Figure 5.2: **MAE restored images**,Here is a comparison between the original images and the images restored by the MAE pre-training. The top row shows the original images, while the bottom row shows the images restored by the model using masked images.

In Figure 5.2, we present three sets of original images alongside their corresponding images restored by the MAE model. It can be observed that the MAE model can almost completely restore geometric shapes, large color areas, and some gradual color changes. However, as seen in the third set of images, the model does not fully restore some details. This is partly related to the masking ratio and the volume of the dataset. When the masking ratio is low, the restoration of details will be better, but it may not necessarily benefit downstream tasks. The model may learn to achieve better restoration results by simply copying known areas. Higher masking ratios help the model learn features of known areas and the connections between blocks in known areas. As for the volume of the dataset, naturally, a larger and more comprehensive dataset will better enhance the model's performance in restoring images and will also be more advantageous for downstream tasks.

5.2 Training Loss Analysis

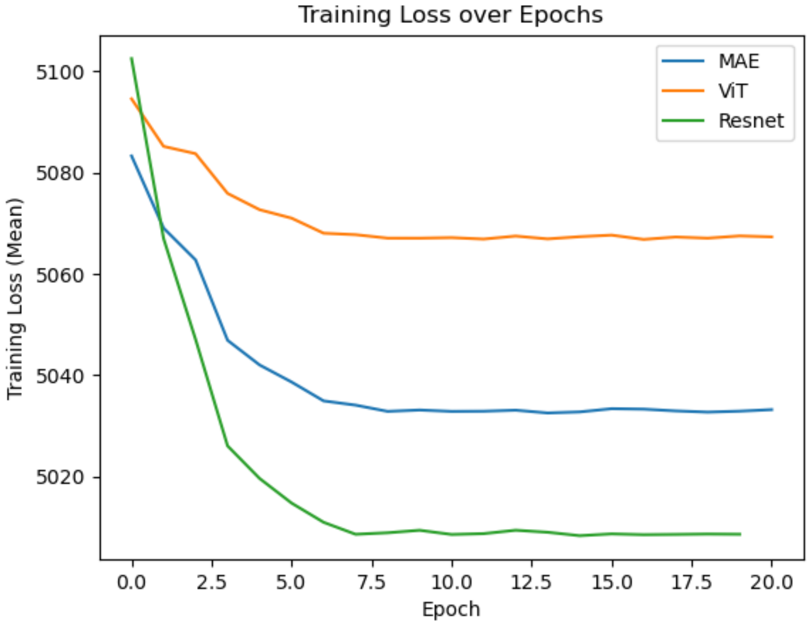


Figure 5.3: **Training Loss Curve**,Here are the training loss curves for three different models on the image geolocation regression task, where ”MAE” refers to the ViT model trained with pre-training weights obtained from the MAE model.

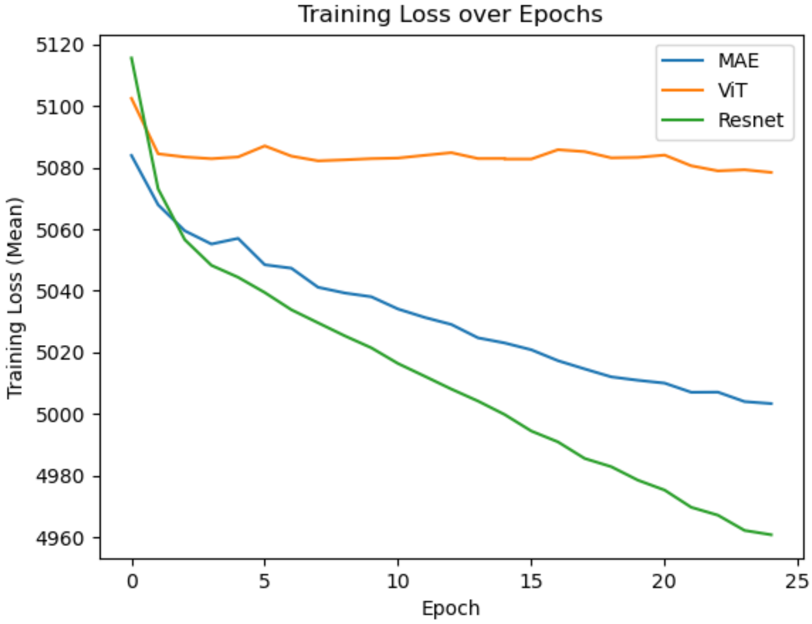


Figure 5.4: **Training Loss Curve**,Here are the training loss curves for three different models on the image geolocation regression task. In this task, the training set comprises half of the entire dataset. ”MAE” refers to the ViT model pre-trained using the other half of the training set and fine-tuned for the regression task along with the same half used for ViT and Resnet.

The two figures, Figure 5.3 and Figure 5.4, respectively depict the training loss curves for the three models on the image geolocation task. The first figure shows the training loss curves when using the entire training dataset for both pre-training and downstream regression tasks. The second figure illustrates the training loss curves when using half of the training dataset for pre-training and the remaining half for downstream regression tasks. In both cases, ”MAE” represents the model pre-trained using the entire dataset and fine-tuned on the entire dataset for regression tasks, as well as the model pre-trained using half of the dataset and fine-tuned on the other half for regression tasks.

As mentioned in Section 3.2.2, the structure of MAE for training downstream regression tasks is identical to that of ViT. Therefore, the training loss curves of ViT and MAE presented in the previous two figures are comparably valuable. In both scenarios, models with pre-training weights from MAE exhibit a superior starting point, with their loss consistently lower than that of ViT regression models without pre-training weights. In Figure 5.3, models with pre-training weights from MAE achieve stability at a lower loss level.The reason lies in the pre-training of MAE on the entire training dataset, during which the model has learned some useful representations of images, and certain parameters have been optimized. Therefore, when training downstream tasks, there is a better initialization, leading to consistently superior training loss compared to models without pre-training weights.However, ultimately, both pre-training and downstream tasks utilize the same training dataset, resulting in almost identical convergence speeds.

In Figure 5.4, the advantage of models with pre-training weights from MAE becomes more pronounced; they not only start at a better point and maintain lower loss throughout training but also demonstrate faster convergence compared to ViT. Even when the loss of the ViT model has stabilized and is no longer decreasing, models with pre-training weights from MAE continue to optimize. The reasons for these phenomena are similar to those mentioned in the preceding paragraph. The key difference is that this time, the training dataset for pre-training is different from that of the downstream regression task. This difference also accounts for the ability of models with pre-training weights to continue optimization even when the ViT model tends to converge. This also underscores the importance of pre-training, where a pre-training dataset that is different but relevant to the training dataset may yield greater benefits. This observation is consistent with real-world scenarios, where most datasets lack ideal and readily available labels. Leveraging these abundant datasets for pre-training can enhance the effectiveness of downstream target tasks.

In both of the above figures, it can be observed that the training loss of the ResNet model consistently outperforms the other two models. This phenomenon may be attributed to the overly complex structures of the other two models, which lack sufficient datasets for training.

5.3 Paired t-test

We conducted a Paired t-test on the results of the ViT model and the ViT model pretrained with MAE on the Im2GPS3k test dataset. The paired t-test is a statistical method used to determine if there is a significant difference between the means of two paired groups. In our analysis, we aimed to evaluate whether the pretraining with MAE had a statistically significant impact on the performance of the ViT model in terms of geolocation prediction accuracy. We conducted a Paired t-test on the test results of the two models on the Im2GPS3k dataset, which measured the distance between their predicted coordinates and the actual coordinates. The resulting t-statistic was -4.38624571, with a p-value of 0.00001193. The results revealed a significant difference between the two models, with a p-value of less than 0.05, indicating that the impact of MAE pretraining on performance is evident, but the mean error of the results from models pretrained with MAE is higher compared to those without pretraining. Moreover, we observe distinct differences in accuracy between the two models across specific ranges. After pretraining, there is a slight improvement in performance at larger ranges, but a deterioration is observed at smaller ranges.

The negative value of the t-statistic indicates that the mean of the second dataset is smaller than that of the first dataset. However, with a p-value greater than the typical significance level of 0.05, there is insufficient statistical evidence to reject the null hypothesis, suggesting that there is no significant difference in the means of the two datasets.

5.4 Accuracy Analysis for Various Regression Models

Model	im2gps	im2gps3k
Random guess	9017.88	8975.40
ResNet	5669.71	5451.12
ViT	5651.33	5215.02
MAE(pretrain:Full ,train:Full)	5582.45	5277.12

Table 5.1: Comparison of Model Accuracy for imgps

Model	Acc200	Acc750	Acc2500
Random guess	1.2%	3.9%	9.6%
Resnet	1.1%	11.4%	31.0%
ViT	2.1%	13.8%	33.9%
MAE	0.7%	12.2%	34.9%
PlaNet[1]	34.3%	48.4%	64.6%

Table 5.2: Comparison of Accuracy on Im2GPS3k

Model	Acc200	Acc750	Acc2500
Random guess	2.1%	4.6%	8.9%
Resnet	1.6%	7.8%	34.4%
ViT	4.7%	7.8%	35.9%
MAE	3.1%	7.8%	34.4%
PlaNet[1]	37.6%	53.6%	71.3%

Table 5.3: Comparison of Accuracy on Im2GPS

In the information displayed in Tables 5.1, 5.2, and 5.3, we observe that all three regression models contribute positively to the geolocation of images, as their average errors and accuracies in larger intervals are superior to random guessing based on clustering. However, compared to the more mature classification tasks, which classify categories based on the distribution of the training set, there is a significant gap. It can be observed that the accuracy of regression tasks varies greatly with the range. The accuracy within 2500 kilometers is much higher than that within 200 kilometers. This is due to the characteristics of regression tasks, where models attempt to balance the errors of all samples while optimizing the average error, resulting in a distribution of errors that is densely populated near the average error. This also indicates that to use regression models for this task, a larger dataset and model structure may be required to reduce the average error and achieve better accuracy. However, in our training dataset, firstly, the data volume of 4.72 million is not sufficient for the entire Earth's range. Moreover, as illustrated by the data distribution plot in Section 1.1, the continuity of the training set data is poor, with significant biases, which greatly affects regression tasks.Correspondingly, when a sufficiently large training dataset is obtained, the number of classes for classification tasks will sharply increase, leading to decreased efficiency.

Furthermore, when using the entire training dataset for self-supervised learning pretraining and downstream task training, the actual effect did not show significant improvement. Instead, there was a slight decrease in accuracy within smaller ranges. However, the average error remained unchanged, which may be attributed to the concentration of error distribution towards the average error set by training with pretraining weights.

Experiment Setting	**Acc200**	**Acc750**	**Acc2500**
Pretrain All + Regression All	0.7%	12.2%	34.9%
Pretrain first Half + Regression last Half	2.3%	11.2%	33.2%
ViT last Half	1.8%	11.9%	32.7%
Resnet last Half	2.4%	11.7%	32.0%

Table 5.4: Comparison of Model Accuracy with Different Pretraining and Regression Settings on Im2GPS3k

Experiment Setting	**Acc200**	**Acc750**	**Acc2500**
Pretrain All + Regression All	3.1%	7.8%	34.4%
Pretrain first Half + Regression last Half	3.1%	7.8%	37.5%
ViT last Half	0%	6.3%	34.4%
Resnet last Half	1.5%	4.6%	37.5%

Table 5.5: Comparison of Model Accuracy with Different Pretraining and Regression Settings on Im2GPS

In the above Figures 5.4 and 5.5, we observe a comparison between ”Pretrain with all training set + Regression with All training set” and ”Pretrain with first Half + Regression with last Half”. Despite halving the training set for the downstream task training, there is no significant impact on accuracy. In fact, there is even a slight improvement in accuracy within 200 kilometers on the Im2GPS3k test set. Additionally, on the Im2GPS test set with more distinct features, there is also a slight improvement in accuracy within 2500 kilometers. Moreover, ”Pretrain with first Half + Regression with last Half” exhibits a slight decrease in average error on both test sets compared to ”Pretrain with all training set + Regression with All training set”. Thus, it can be concluded that when using self-supervised learning pretraining, employing a dataset that is related to but not entirely identical to the downstream task dataset may lead to more significant improvements in the downstream task performance.

Furthermore, comparing the results obtained by using half of the training set with those of the two baseline models also reflects the positive impact of employing a dataset similar but not identical to the downstream task’s dataset. Relative to Table 5.2 and Table 5.3, where the same dataset was used for both pretraining and downstream task training, this positive impact becomes more pronounced.Using only half of the labeled data achieves results comparable to training with the entire labeled dataset. This underscores the significance of self-supervised learning pretraining, which leverages the abundance of unlabeled data available in real-world scenarios.

6 Conclusion

Based on the experiments and results presented above, we can draw the following conclusions:

1. **Effectiveness of Self-Supervised Learning Pretraining**: The experiments demonstrate the effectiveness of self-supervised learning pretraining, particularly with the MAE model, in enhancing downstream regression tasks. Models pretrained with MAE weights consistently outperform those without pretraining, exhibiting faster convergence and achieving lower training losses.

2. **Impact of Dataset Similarity on Model Performance**: Utilizing a dataset similar but not identical to the downstream task dataset for pretraining can lead to a significant improvement in downstream task performance. This is evidenced by the comparison between models pretrained with different subsets of the training dataset and those pretrained on the entire dataset.

3. **Potentials of Regression Models**: Despite the challenges posed by regression tasks, such as the need for larger datasets and model structures to reduce average error, regression models still demonstrate positive contributions to image geolocation tasks compared to random guessing based on clustering. Unlike classification tasks where the loss is unrelated to the distance between predicted and true classes, regression tasks directly measure loss based on the distance between predicted coordinates and true coordinates. This direct measurement is more intuitive and requires fewer computational resources. Additionally, regression tasks are less reliant on the distribution of class labels in the training set, potentially leading to better transferability and generalization compared to classification tasks.

In conclusion, the experiments underscore the importance of self-supervised learning pretraining and the careful selection of training strategies, highlighting the potential for significant improvements in downstream task performance through thoughtful dataset selection and model training procedures.

7 Future works

The potential of regression models in image geolocation tasks remains to be further explored, with several areas offering avenues for research and improvement. Firstly, investigating how to better leverage prior knowledge or other auxiliary information to enhance the performance of regression models could be fruitful. For instance, pre-segmenting images based on scene recognition into specific scenes and then applying regression models to different scenes could be explored. This is because features within uniform scenes tend to be more continuous. For example, combining classification and regression to address the image geolocation problem could be another promising approach. As illustrated in Figure 1.1, the distribution of the mp-16 dataset relies heavily on the population and development level of countries and regions. We could first conduct a classification task at the country or even larger regional level, and then train separate regression tasks for each category. This approach could significantly improve the geolocation accuracy of images in densely populated regions of the dataset. Regional tasks might indirectly demonstrate the superiority of regression tasks.

Furthermore, there is significant potential for further research into self-supervised learning pretraining. Exploring and utilizing more similar unlabeled datasets could be beneficial. This is because finding unlabeled or inadequately labeled datasets is often much simpler than finding datasets with readily available labels.

Overall, these directions could lead to advancements in the field, improving the robustness and accuracy of regression models in diverse geographic environments and data distributions, thereby broadening their applicability and effectiveness.

Bibliography

[1] Tobias Weyand, Ilya Kostrikov, and James Philbin. “PlaNet - Photo Geolocation with Convolutional Neural Networks”. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2016, pp. 37–55. ISBN: 9783319464848. DOI: 10.1007/978-3-319-46484-8_3. URL: http://dx.doi.org/10.1007/978-3-319-46484-8_3.

[2] Eric Müller-Budack, Kader Pustu-Iren, and Ralph Ewerth. “Geolocation Estimation of Photos Using a Hierarchical Model and Scene Classification”. In: *European Conference on Computer Vision*. 2018. URL: https://api.semanticscholar.org/CorpusID:52954003.

[3] Shraman Pramanick et al. *Where in the World is this Image? Transformer-based Geo-localization in the Wild*. 2022. arXiv: 2204.13861 [cs.CV].

[4] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[5] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV].

[6] Lukas Haas, Silas Alberti, and Michal Skreta. *Learning Generalized Zero-Shot Learners for Open-Domain Image Geolocalization*. 2023. arXiv: 2302.00275 [cs.CV].

[7] Liangliang Cao et al. “BlueFinder: Estimate Where a Beach Photo Was Taken”. In: *Proceedings of the 21st International Conference on World Wide Web*. WWW ’12 Companion. Lyon, France: Association for Computing Machinery, 2012, pp. 469–470. ISBN: 9781450312301. DOI: 10.1145/2187980.2188081. URL: https://doi.org/10.1145/2187980.2188081.

[8] Georges Baatz et al. “Large Scale Visual Geo-Localization of Images in Mountainous Terrain”. In: *Computer Vision – ECCV 2012*. Ed. by Andrew Fitzgibbon et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 517–530. ISBN: 978-3-642-33709-3.

[9] Olivier Saurer et al. “Image Based Geo-Localization in the Alps”. In: *Int. J. Comput. Vision* 116.3 (Feb. 2016), pp. 213–225. ISSN: 0920-5691. DOI: 10.1007/s11263-015-0830-0. URL: https://doi.org/10.1007/s11263-015-0830-0.

[10] Eric Tzeng et al. “User-Driven Geolocation of Untagged Desert Imagery Using Digital Elevation Models”. In: *2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2013, pp. 237–244. DOI: 10.1109/CVPRW.2013.42.

[11] Amir Roshan Zamir and Mubarak Shah. “Accurate Image Localization Based on Google Maps Street View”. In: *European Conference on Computer Vision*. 2010. URL: https://api.semanticscholar.org/CorpusID:7581570.

[12] Bolei Zhou et al. “Recognizing City Identity via Attribute Analysis of Geo-tagged Images”. In: *European Conference on Computer Vision*. 2014. URL: https://api.semanticscholar.org/CorpusID:2735548.

[13] Gabriele Berton, Carlo Masone, and Barbara Caputo. *Rethinking Visual Geo-localization for Large-Scale Applications*. 2022. arXiv: 2204.02287 [cs.CV].

[14] Sudharshan Suresh, Nathaniel Chodosh, and Montiel Abello. *DeepGeo: Photo Localization with Deep Neural Network*. 2018. arXiv: 1810.03077 [cs.CV].

[15] James Hays and Alexei Efros. “IM2GPS: Estimating geographic information from a single image”. In: July 2008, pp. 1–8. DOI: 10.1109/CVPR.2008.4587784.

[16] Amir Roshan Zamir and Mubarak Shah. “Image Geo-Localization Based on MultipleNearest Neighbor Feature Matching UsingGeneralized Graphs”. In: *IEEE Trans-*

actions on Pattern Analysis and Machine Intelligence 36.8 (2014), pp. 1546–1558. DOI: 10.1109/TPAMI.2014.2299799.

[17] Nam Vo, Nathan Jacobs, and James Hays. *Revisiting IM2GPS in the Deep Learning Era*. 2017. arXiv: 1705.04838 [cs.CV].

[18] Alexandre de Brébisson et al. *Artificial Neural Networks Applied to Taxi Destination Prediction*. 2015. arXiv: 1508.00021 [cs.LG].

[19] Menghua Zhai et al. *Learning Geo-Temporal Image Features*. 2019. arXiv: 1909.07499 [cs.CV].

[20] Randall Balestriero et al. *A Cookbook of Self-Supervised Learning*. 2023. arXiv: 2304.12210 [cs.LG].

[21] Ting Chen et al. *A Simple Framework for Contrastive Learning of Visual Representations*. 2020. arXiv: 2002.05709 [cs.LG].

[22] JANE BROMLEY et al. "SIGNATURE VERIFICATION USING A "SIAMESE" TIME DELAY NEURAL NETWORK". In: *International Journal of Pattern Recognition and Artificial Intelligence* 07.04 (1993), pp. 669–688. DOI: 10.1142/S0218001493000339. eprint: https://doi.org/10.1142/S0218001493000339. URL: https://doi.org/10.1142/S0218001493000339.

[23] Xinlei Chen and Kaiming He. *Exploring Simple Siamese Representation Learning*. 2020. arXiv: 2011.10566 [cs.CV].

[24] Mathilde Caron et al. *Emerging Properties in Self-Supervised Vision Transformers*. 2021. arXiv: 2104.14294 [cs.CV].

[25] Adrien Bardes, Jean Ponce, and Yann LeCun. *VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning*. 2022. arXiv: 2105.04906 [cs.CV].

[26] Deepak Pathak et al. *Context Encoders: Feature Learning by Inpainting*. 2016. arXiv: 1604.07379 [cs.CV].

[27] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV].

[28] Kaiming He et al. *Masked Autoencoders Are Scalable Vision Learners*. 2021. arXiv: 2111.06377 [cs.CV].

[29] M. Larson et al. "The Benchmarking Initiative for Multimedia Evaluation: MediaEval 2016". In: *IEEE MultiMedia* 24.01 (Jan. 2017), pp. 93–96. ISSN: 1941-0166. DOI: 10.1109/MMUL.2017.9.

[30] Jonas Theiner, Eric Mueller-Budack, and Ralph Ewerth. *Interpretable Semantic Photo Geolocation*. 2021. arXiv: 2104.14995 [cs.CV].

[31] James Hays and Alexei A. Efros. "im2gps: estimating geographic information from a single image". In: *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2008.

[32] James Hays and Alexei A. Efros. "Large-Scale Image Geolocalization". In: *Multimodal Location Estimation of Videos and Images*. 2015. URL: https://api.semanticscholar.org/CorpusID:22758644.

Technical
University of
Denmark

2800 Kgs. Lyngby
Tlf. 4525 1700