Sweden or Switzerland – GeoLocation on Noisy Datasets

Björn Engdahl, Matthias Heubi
Project mentor: Wenlong Huang (wenlongh@stanford.edu)

Department of Computer Science
Stanford University
engdahl@stanford.edu, mheubi@stanford.edu

Abstract

GeoLocation aims to predict the location where an image was taken without the use of metadata such as GPS-coordinates. We create a 180k-sized dataset for Sweden and Switzerland based on World-Wide Scale Geotagged Image Dataset and attempt to predict the country as well as the GPS coordinate for the images. We compare three different models, the first two of which are based on our own ideas and the third one inspired by GeoCLIP. We introduce a new paradigm for dividing the geographic area into cells – a crucial part of many geolocation models. Finally, we show that our multi-task regression model gives the overall best results for country-level and lower resolution predictions (200km and 750km range), whereas the newly proposed cell model gives overall best results for higher resolution predictions (1km and 25km range).

1. Introduction

The task of determining the latitude and longitude where a photo was taken is commonly known as *geolocation* or *reverse geotagging*. Performing it at a world-wide scale purely based on the image itself, without relying on metadata, is a challenging task. Apart from being interesting from a research perspective, it has gained substantially in importance and even public mind share due to recent geopolitical developments, such as the war in Ukraine: Images appearing in social media must be confirmed to be authentic both in terms of content and location to avoid falling prey to disinformation.

The problem becomes increasingly difficult as you scale up. From determining the location in urban areas with distinctive landmarks, to the harder task at country and world-level where disjoint regions may share common terrain features or not be covered by images in the training set at all.

For this project we limit the scope to our two home countries, Switzerland and Sweden. Given an image, we try to estimate the location within these countries as accurately as possible.

Starting out simple for the baseline, we first aim at determining in which of the two countries an image was taken.

From there we make the problem progressively harder, by trying to narrow down the location within each country. For the baseline we use an image encoder and feed the features through an MLP with a classifier head. To predict a more accurate position, we evaluate three different architectures, the first two based on our own original ideas and the third inspired by GeoCLIP [11]. **1)** A multi-task approach where we add a second head to the baseline model and estimate coordinates directly using regression. **2)** Dividing each country into a finer network of cells and classifying the image into one of the cells, and **3)** A retrieval based approach where we use contrastive learning to align image and GPS features. We demonstrate that each model has advantages and disadvantages depending on the resolution of the prediction accuracy.

2. Related Work

Widely considered the first seminal work was *Im2gps* [15] in 2008. This *retrieval based* approach was matching a query image against a database of reference images based on hand-tuned features (this was before the advent of deep learning and convolutional neural networks). The authors compiled a dataset from Flickr of more than 6 million GPS-tagged images and represented the matching result as a probabilistic distribution (heat map) over earth's surface. Being comparison-based this approach does not scale well.

With the breakthrough of deep learning and CNNs, Google presented *PlaNet* [27] in 2016. This system approached the task as a *classification problem*, dividing earth's surface into adaptively sized cells (thereby balancing the number of training examples per cell). By using an *Inception* [23] network as backbone, the scaling problem

was solved thanks to constant inference time. The system required a massive 91 million images for training, however.

In 2018, *Müller-Budack et al.* [17] improved upon PlaNet by introducing the idea that some features have a wider geographic span than others, and that relevant features may depend upon the scene type (indoors, outdoors, etc.). The authors used multitask learning with cells at different spatial resolutions to model hierarchical features, using a *Resnet-152* [16] as backbone. This system outperformed PlaNet while requiring only 4.7 million training images.

In 2022, Pramanick et al. created *TransLocator* [18], a dual-branch transformer-based model taking in features from the RGB image as well as its semantic segmentation map in the two branches. The transformer layers perform cross-attention between the branches. The model produced state-of-the-art results for continent-level accuracy.

In 2023, *Pigeon* [14] used multi-task contrastive pretraining, a new loss function, and downstream guess refinement, among other tweaks, to achieve state of the art accuracy. Based on *CLIP ViT-L/14 336* [19] as backbone, the system was the first to outperform the best humans. It still approached the problem as a classification task, mapping an image onto a cluster of geocells which is then refined into a location guess. The CLIP transfomer was finetuned to the geolocation task using only 400,000 images (however it came pretrained on 400 million images and captions).

Finally, the same year *GeoCLIP* [11] basically went back to a retrieval based approach. However, instead of comparing against millions of images, they proposed to model features as a learned continuous function around the globe, then using this function to generate a gallery of reference features for all locations, including those for which no training image was provided. They project images and GPS coordinates into a shared feature space and align them using the contrastive learning approach introduced by *SimCLR* [12]. The inference part then matches the query image against this gallery. The hierarchical nature of features is supported by positional encoding based on random fourier features (RFF).

3. Data

We base our training and evaluation on the *World-Wide Scale Geotagged Image Dataset* [22]. The authors compiled metadata of more than 14 million images from Flickr [1], tagged with GPS position, of which we only use images from Switzerland and Sweden.

This dataset is challenging in that it is not restricted to landscape, outdoor pictures. Rather, it consists of whatever the users have uploaded, a mixture between indoors, outdoors, pets, family and landscape images, etc.

We filter the metadata based on longitude/latitude to isolate Switzerland and Sweden, using publicly available geographical country shapes from GADM[1]. This results in 131k and 101k image references, respectively. While downloading the actual image data from Flickr, we find that more than 20% are no longer accessible, resulting in a net image stock of 101k and 80k. Figure 1 shows the distribution of images from Sweden. Switzerland is more balanced across the country.

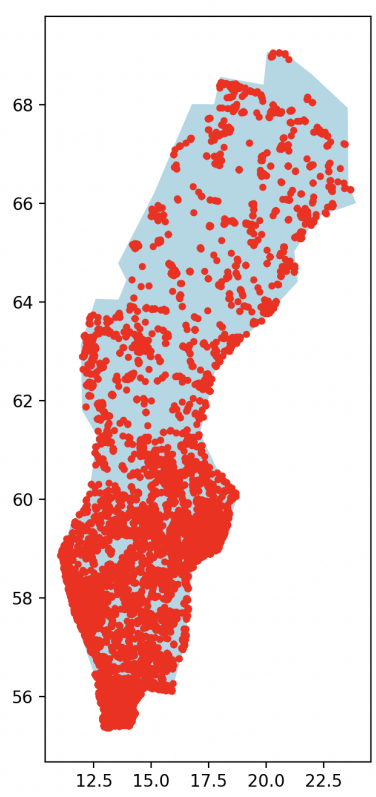

Figure 1. Geographic distribution of Swedish images in the dataset. Urban areas in the South are much more densely covered than the rural northern part.

We divide the resulting dataset into 80/10/10 partitions for train/val/test. Furthermore we create several smaller versions of the dataset (1K/10K/20K/100K) to be used in our experiments. These datasets are drawn from the original train partition, so they can all be evaluated using the large test set as well, ensuring more stable comparison.

We resize to 224x224 and normalize each channel to mean=[0.485, 0.456, 0.406] and standard deviation=[0.229, 0.224, 0.225].

[1]https://gadm.org

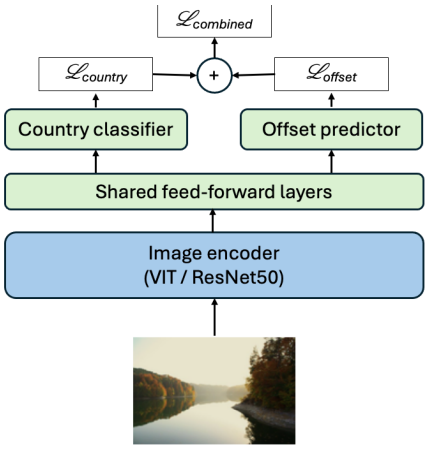

Figure 2. Architecture of our Multi-task offset predictor model.

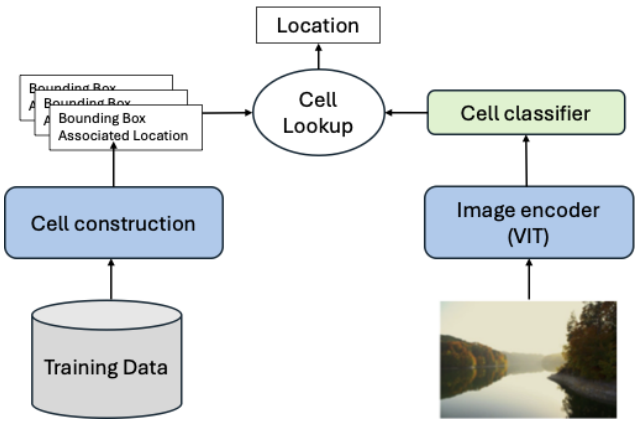

Figure 3. Process of cell-based position estimation.

4. Method

4.1. Baseline - Predict Country

For the baseline, we formulate the problem as predicting the country where the photo was taken. We create a pipeline of pretrained versions of the ResNet50 [16] and ViT_B_16 [13] encoders from torchvision [5] with a three layer MLP on top. Although *Müller-Budack et al.* relied on a larger ResNet152 [17], *Theiner et al.* later demonstrated [25] that ResNet50 can achieve the same or better accuracy. We use cross entropy loss to train our model.

4.2. Predicting the location within the country

To improve the resolution, we try different approaches to predict the precise GPS position within each country. We have categorized our approaches into three categories:

- **Regression**: Use regression to directly predict the target coordinates, based on the image's features
- **Classification**: Split the country into cells which provide a single representative position each, then classify the image into one of these cells
- **Retrieval**: Use a retrieval based approach to obtain coordinates based on the closest match from a pool of coordinates associated with reference features

4.3. Regression - "Multi task offset predictor"

We first devise a new method by adding another head to our baseline model. In addition to classifying the country, it now predicts a normalized latitude and longitude offset o_{lat} and o_{lon} within the country as measured from the lower left corner. Rather than predicting the raw GPS position directly, this limits the prediction to the range 0-1, which will avoid generating vastly wrong locations, but makes the two tasks somewhat co-dependent. Other works [18, 20] have shown that training on multiple tasks simultaneously can improve the results of a main task. We theorize that adding the new regression task might improve the accuracy for the country classification task as well.

We define our multi-task loss:

$$\mathscr{L}_{\text{comb}} = \alpha\mathscr{L}_{\text{country}} + (1-\alpha)\mathscr{L}_{\text{offset}}$$

where $\mathscr{L}_{\text{country}}$ is our cross entropy loss from the baseline and $\mathscr{L}_{\text{offset}}$ is the MSE loss of the offset. α is a constant to balance losses, we use $\alpha = 0.5$.

The architecture is shown in figure 2.

4.4. Classification - "Cell predictor"

Improving prediction resolution using cells is fairly straightforward: Instead of classifying just into two countries, we split each country into a number of cells, then use the same classifier architecture as in the baseline, but with c classes, where c is the total number of cells. After classifying an image into a cell, we use the GPS coordinate that was associated with that cell during cell construction as the image's location guess. The full process is shown in figure 3. We use cross entropy loss to train the model.

Final accuracy will depend on how suitable the cells are constructed and how well the GPS coordinate associated with the cell statistically reflects the coordinates of the images contained in the cell.

The estimate could be further improved by intelligently weighting the top x cells produced by the classifier, as done in [14] (where x might be specified as a fixed number, a percentage, or a probability cutoff threshold), maintaining multiple cell resolutions [17], maintaining overlapping cells at the same resolution [21] and/or switching to a retrieval approach inside the cell [14].

For the purpose of this paper, we focus on cell construction only and base our results directly on the associated GPS coordinate without any further downstream refinement.

4.4.1 Cell construction

With cells being used as labels for the classifier, there are multiple factors that make for "good" cell construction.

Balance: The number of training images per cell should be somewhat balanced to create a solid classifier.

Cell size: Since a cell ultimately places all its images at a single GPS coordinate, smaller cells make for more accurate guesses. However, smaller cells means more cells overall, which means the likelihood of image misclassification increases, thereby again lowering overall prediction accuracy.

Feature variance: The more cohesively the training image features for one cell are clustering, the sharper the classifier will become.

Minimum count: Finally, in addition to being balanced, each cell should contain a certain minimum number of training images.

Past research mostly used rectangular S2 cells [6] that were adaptively divided to obtain a balanced number of images per cell [27, 17]. Some used administrative boundaries like roads and city limits, arguing it would yield better feature cohesion and interpretability [14, 25] and one used it's own combinatorial partitioning algorithm as a method to improve resolution [21] .

Choosing $n = 200$ as the minimum number of images per cell, we first implement a simple top-down approach: We start with a single cell covering the country bounds. We then recursively split this cell into 4 subcells as long as any of the subcells contain at least n images. The recursion is ended prematurely if a minimum cell size of 0.01 degrees (approximately 1km latitude) is reached.

We then modify this method by first recursively splitting, then recursively consolidating images of cells with less than n images back up to their parent cells until the minimum number of images is reached. This approach may result in a small cell placed inside a big cell, whereby the big cell no longer includes the images from the area of the small cell. The intuition being that accuracy should improve, mainly due to having less cells overall and by ensuring that all cells contain the minimum number of images and are well balanced. While the overall number of cells gets reduced by about 30% and classification accuracy improves slightly, we don't see a statistically significant change in location accuracy. This is most likely due to the fact that former misclassifications to a wrong nearby small cell are now substituted by correct classifications to a much larger cell.

Finally, we introduce a new bottom-up cell construction algorithm, which considers all four given criteria while constructing cells. To the best of our knowledge this is a completely new approach.

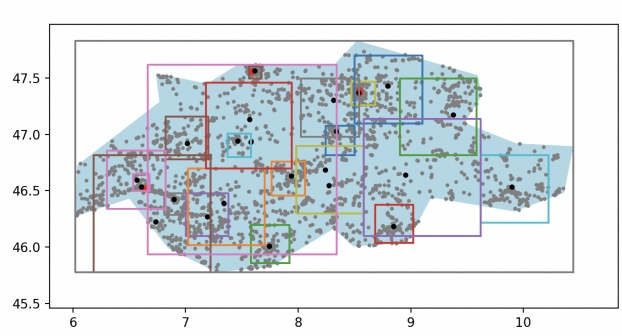

Figure 4. Bottom-up constructed cells for Switzerland (10K set). Assigned GPS locations are shown in black, red dots signify locations where the cell was too small to be displayed properly

4.4.2 Bottom-up cell creation

Our cell construction approach was inspired by the realization that cells are just groupings of images which will ultimately translate to the same single associated GPS coordinate. While human intuition would expect cells to form a non-overlapping mesh, there is no such requirement from a classification perspective (and we already broke that assumption with our recursive consolidation approach above).

We thus propose to build cells bottom-up by creating density heat maps at ever decreasing resolution, grouping more and more images into one heat map cell. At each resolution we check whether the heat map contains spots with more than the required minimum number of images for cell construction. We then create a candidate cell for each hot spot to claim those images. Finally we compute the *within-cluster sum of squares (WCSS)* for each candidate cell's images and pick the one cell with the highest score (based on features extracted by the encoder). Those images then get removed from the heat map computation and the process repeats.

We picked *WCSS* as a simple measure for feature cohesion to cluster the data bottom-up not only spatially but also visually. The hypothesis being, that such algorithmic clustering might ultimately yield better results than other heuristics, such as the assumption that administrative boundaries also contain visually similar features, for example.

For a cluster (or cell in our case) C_i, WCSS is defined as the sum of the squared Euclidean distances between each feature vector and the centroid of the cluster:

$$\text{WCSS}_i = \sum_{x_j \in C_i} \|x_j - \mu_i\|^2$$

where $x_j \in R^d$ is the feature vector for image j in cluster i, and $\mu_i \in R^d$ is the mean vector of all image feature vectors of the cluster. d is the number of features.

The resulting cell allocation for Switzerland (using the 10K dataset) is shown in figure 4. While it looks non-intuitive and somewhat chaotic to the human eye, it does actually improve prediction accuracy over the top-down approach.

Note that the resulting image allocation may still be quite unbalanced (in all cell-based approaches) because we constrain cell size to a minimum of 0.01 degrees. But only cells in urban hotspots contain a disproportionally large number of images. While not ideal for practical applications, this could actually be beneficial from a benchmark perspective, because it puts more emphasis on these "high precision, high volume" centers while training the classifier – at the expense of a few low volume countryside guesses.

4.4.3 Coordinate association

The first idea that comes to mind when mapping a cell to GPS coordinates would be to use the cell's center point. However, as [17] has pointed out, it is more beneficial to use the mean of all image coordinates in the cell as a location guess, because it will be closer to the majority of images occurring in that cell. We thus implement the mean.

4.5. Retrieval: "Contrastive gallery predictor"

Taking inspiration from the contrastive learning approach proposed by SimCLR [12] and GeoCLIP's application of SimCLR [11, 2] to a GPS position retrieval problem, we create the contrastive gallery predictor. Traditionally, geolocation retrieval problems construct a gallery of images with known GPS-positions and then try to match a test image against the gallery of training images [15]. Such a gallery can become very large. Therefore GeoCLIP [11] instead proposes a gallery of GPS positions, $\mathcal{G}$. We base our method on their solution and formulate the problem as a contrastive learning problem similarly to SimCLR.

We use an image encoder $\mathcal{V}(\cdot)$ to project the image into a 512d feature space. We also project the 2d GPS-position into a 512d feature space using a location encoder $\mathcal{L}(\cdot)$

During training we encode images $(I_1, \ldots, I_n)$ and locations $(G_1, \ldots, G_n)$ from the training dataset to form features $F_{I,i} = \mathcal{V}(I_i)$ and $F_{G,i} = \mathcal{L}(G_i)$. For a minibatch of positive examples, $\mathcal{M}_p$, we augment each image in the mini-batch with negative samples, i.e. false GPS positions, $\tilde{F}_{G,i}$, $\mathcal{M}_n$. To this extent we try two variations 1) sample negative positions from the training set, and 2) like GeoCLIP use GPS positions from previous mini batches.

We try to improve results by applying random transformation to the image.

Our training objective is formulated like SimCLR - to minimize the distance (cosine similarity) between F_{I_i} and F_{G_i} for positive pair in $\mathcal{M}_p$, and at the same time maximizing the distance between each pair in $\mathcal{M}_n$. We use the normalized temperature-scaled cross-entropy used in [12] and explored in [26]. For the i'th sample from a batch B, the loss $\mathscr{L}_i$ becomes:

$$\mathscr{L}_i = -\log \frac{\exp(\text{sim}(F_{I,i}, F_{G,i})/\tau}{\text{sim}(F_{I,i}, F_{G,i}) + \sum_k^{|\mathcal{M}_\mathcal{N}|} \text{sim}(F_{I,i}, \tilde{F}_{G,k})}$$

During inference, we compute the similarity between the encoded image vector, $\mathcal{V}(I)$ and the encoded positions in the gallery, $\mathcal{L}(G_i)$ for each $G_i \in \mathcal{G}$. We pick the gallery position with the highest similarity as our predicted location.

For the location encoder we try two variants: A 3-layer fully connected network as well as a random fourier series layer (see below) inserted before the fully-connected network.

[11] also found that using the equal earth [10] projected coordinate, G_{ee} instead of the raw GPS coordinate slightly improved the results. We use that as well.

The architecture is depicted in figure 5.

Random Fourier Series: Previous work has shown that MLPs have inherent spectral bias, failing to capture higher frequencies [9]. Specifically, we use random fourier series and employ the method proposed by [24] also used by [11] and add a fourier feature mapping of our 2d GPS position, $\gamma(\cdot)$, before our MLP layer in an attempt capture high frequency details when projecting our 2d position into higher dimensions, in order to improve the accuracy at finer resolutions. Given an input feature vector G, $\gamma(\cdot)$ encodes $G' = \gamma(G) = [\cos(2\pi \mathbf{R}G), \sin(2\pi \mathbf{R}G)]^T$, where $\mathbf{R}$ is a fixed matrix of random frequencies whose entries $r_{i,j} \sim \mathcal{N}(0, \sigma)$. A larger σ will result in higher frequencies.

Given a location G and an MLP layer $f_{mlp}(\cdot)$, our location encoder now performs $\mathcal{L}(G) = f_{mlp}(\gamma(G_{ee}))$. Similarly to GeoClip, we add results from C different sigmas to capture a wider range of frequencies and add the results, so that the full location encoder becomes $\mathcal{L} = \mathcal{L}_{\sigma 1} + \cdots + \mathcal{L}_{\sigma_C}$. Unless otherwise stated we use C=3 and $\sigma = 2^0, 2^8, 2^{16}$.

4.6. Dataset size

For a subset of our models we perform experiments with different dataset sizes to determine the sensitivity of the model to variations in dataset size.

4.7. Ablations

We perform several ablations to determine the effects of the different alternations made to our model. We try to isolate the effect of the following components.

- Regression: Effect of multi-task learning for country classification over single task learning
- Classification: Additional effect of WCSS ranking over heatmap based bottom-up candidate cell creation

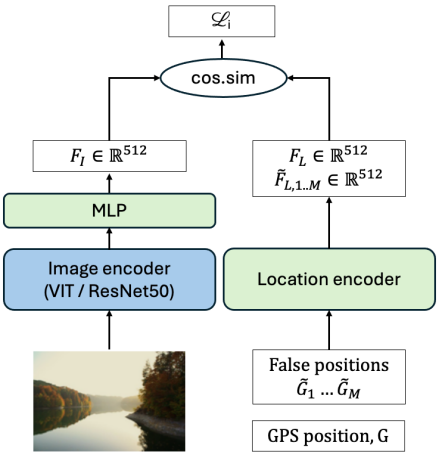

Figure 5. Contrastive gallery predictor model architecture. The image shows the training for a single pair of image/gps position.

- Classification: Difference between using location or features as a source for WCSS ranking
- Retrieval: Effect of contrastive learning over vanilla learning method
- Retrieval: Effect of random fourier series layer over vanilla MLP layer
- Retrieval: Effect of augmenting training set with positive samples through random transformations

5. Results

5.1. Evaluation metrics

For evaluation we use two metrics:

- We determine accuracy as percentage of test images correctly classified by their *country of origin*
- For exact positioning, we use the percentage of test images correctly placed *within a radius of* r from the ground truth, a metric first established by *Im2gps* [15] and used as a reference ever since. In literature, it is common to evaluate accuracy within the following radii: 1km (street), 25km (city), 200km (region), 750km (country), 2500km (continent). We omit the continent level in this project. To compute the distance we use the haversine distance, where r = radius of the earth, ϕ_1, ϕ_2 = latitude positions and λ_1, λ_2 = longitude positions [2].

[2] $d = 2r \sin^{-1}(\sqrt{\frac{1-\cos(\phi_2-\phi_1)+\cos\phi_1 \cdot \cos\phi_2 \cdot (1-\cos(\lambda_2-\lambda_1))}{2}})$

Model	Country	1km	25km	200km	750km
Baseline	77	n/a	n/a	n/a	n/a
Regr	**78.1**	0.1	4.8	**63.4**	**77.2**
Cells-down	75.4	14.8	31.4	61.8	74.7
Cells-up (ours)	75.9	**15.5**	**31.8**	61.2	75.0
Retr-MLP	76.7	0.1	6.7	53.3	73.4
Retr-RFF	77	15.1	30	61.9	76

Table 1. Overall best results per model. The table displays test set accuracies for the different evaluation metrics for our Baseline, Regression, Cell-TopDown, Cell-BottomUp, Retrieval with MLP and Retrieval with RFF models.

5.2. Experiments

We evaluate our three models and compare to the baseline. Overall results are shown in table 1. Below we elaborate on the results for the different models.

Implementation details: We use PyTorch [5] for building our model. Pandas [4] to handle csv-data, Scikit to create t-sne clusters [7], GeoPandas [3] to process spatial data.

Training details: We train our models using the Adam optimizer. We use a learning-rate of 1e-5 and train on a single GPU. Due to limitations of compute we train our models for 3-5 epochs where the validation accuracies generally plateau, although a slightly better result might have been reached after additional epochs. We do not employ other tweaks such as learning rate schedules or gradual unfreezing, again due to limited time and compute. Also due to compute limits, we only do single seed runs, although ideally results would have been averaged across several runs with different seeds. Apart from the baseline we run all experiments on the ViT encoder as it performed slightly better than ResNet on the full dataset for the baseline.

5.3. Regression: Multi-task offset predictor

We run the model on our full dataset and compare results with the baseline for the overall country classification task as well as the prediction accuracy on the various resolutions. As can be seen in table 1, the country level accuracy is slightly better compared to the baseline and also our best performing of overall for country prediction. This is likely due to the multi-task objective where the country classification task is aided by the coordinate prediction.

The coordinate prediction task does not perform well on higher resolutions 1km and 25km. To investigate further, we try to overfit the prediction on a smaller dataset, and manage to do so only after >100 epochs for the small 1K dataset. This indicates that this model architecture is sub-optimal for predicting on higher resolutions. The feed-forward network with a regression head is unable to adapt to the noisy dataset.

5.4. Cell predictor

We train the model twice on the full dataset, once using the top-down and once using the bottom-up approach, applying identical hyperparameters each time: Minumum number of images per cell $n = 200$ and minmal size of cells $s = 0.01$. For visual clustering during bottom-up (WCSS computation upon cell construction), we use the image features already extracted from the training images by the fine-tuned encoder during the top-down approach.

The data shows a small but significant improvement on high resolution predictions, improving 1km and 25km accuracy by 0.7% and 0.4% respectively, in line with our hypothesis. To separate the effect of bottom up heatmap based cell proposal and WCSS ranking for cell construction, we perform additional ablation studies, which show that the WCSS ranking is indeed crucial for the approach to work. In fact, doing bottom-up with heatmap alone reduces 1km accuracy by 0.9% compared to top-down. Another experiment, evaluating image coordinates vs. image features for WCSS computation, shows similar performance, indicating a high correlation between features and location, which was to be expected.

5.5. Retrieval: Contrastive gallery predictor

We augment each image in a mini-batch with false coordinates creating a mini-batch of 1 positive pair and M negative pairs. We find the M=1024 produces good results. We also experiment with introducing several positive variations with random transformations of the image, but find that the model now needs many more epochs to converge, although the result in the end is better. Hence, due to lack of compute, we decide to only have the single positive pair going forward. Refer to 5.6.3 for a limited experiment on an augmented dataset.

For the contrastive learning model, we run with and without the RFF-layer to be able to isolate the effect of the contrastive learning approach separately from the RFF-layer. We see that the model without the RFF-head produces slightly worse results than our Multi-task offset predictor except for the 25km resolution. The results could possibly improve further with an augmented dataset, to leverage the full potential of the contrastive learning setup. When adding the RFF layer though, we see a dramatic increase in accuracy for the finer resolutions. Indeed, the spectral bias in MLPs when projecting from the 2d coordinate to the 512d feature space is mitigated by the RFF-layer. We are able to improve the finer resolutions even further by tuning the frequency range in the matrix $\mathbf{R}$, but at the expense of accuracy for the country-level resolution. The big improvement from adding the RFF layer before the MLP is in line with our findings in section 5.3, that a feedforward network is unable to accurately predict raw GPS positions.

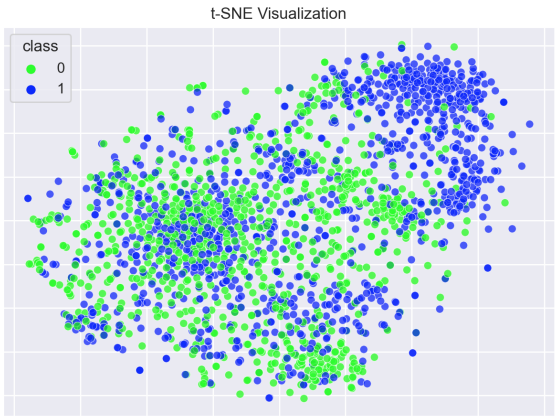

Figure 6. t-SNE clustering of image features. Blue dots are Swiss images and green dots Swedish images.

Figure 7. Examples of correctly / incorrectly classified images. **A)** Correctly classified SE, **B)** Incorrectly classified as CH, **C)** Correctly classified as CH, **D)** Incorrectly classified as SE.

5.6. Other findings

5.6.1 Analyzing classified images

We analyze the results of correctly vs incorrectly classified images to see if any patterns could be detected. Given the broad category of images ranging from pets, family to outdoor, it's difficult to find patterns. Figure 6 shows samples of correctly and incorrectly classified images. It is interesting to see that Swedish image (A) containing Swedish text / logotype is correctly classified. (B) The general image is incorrectly classified as Swiss. With Switzerland and Sweden both being mountainous, snowy countries, (D) is incorrectly classified as Sweden. A human performing the task could easily have arrived at the same conclusions.

Looking at a t-SNE visualization of the image features in

Figure 8. Model dependence on dataset size, showing test accuracy for the 180K and the 20K dataset. Cell creation is top-down.

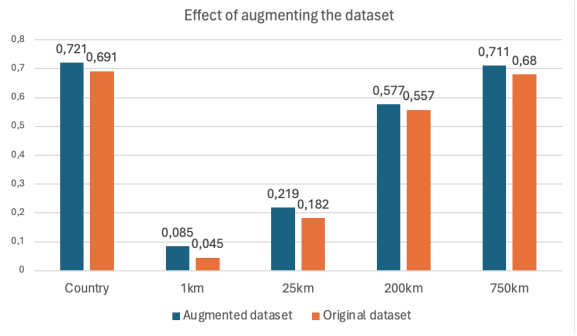

Figure 9. Effects of training on augmented dataset. Results for the contrastive RFF-model trained on a 20K dataset.

figure 7, we can see a blue cluster of Swiss images. Looking at a number of samples from this cluster they mostly stem from outdoor Alps images (there are no Alps in Sweden). Apart from this collection of dots, there is no discernible clear cluster, which indicates that many images are not easily or clearly attributable to a specific feature of a country.

5.6.2 Effect of small training sets

Figure 8 shows an experiment for the two highest prediction resolutions, when running the models on a smaller dataset. We can see that the cell based approach degrades more favorably than the retrieval approach. It requires new hyperparameter tuning to achieve this result, though, as – all else being equal – lowering the number of images would increase the size of cells.

5.6.3 Augmenting the dataset

Evaluating augmenting the dataset with random transformations, we run a limited experiment adding the same transformations as [12]. We run the RFF model on a dataset of 20K images and notice that accuracy improves by a few percent, but training requires many more epochs to converge. The original dataset converged after 5 epochs whereas the augmented dataset requires 18 epochs. This is likely due to the noise introduced by the transformations requiring the model to see many variations before reaching good results.

6. Conclusion

We addressed the problem of geolocation - determining the country and the GPS position where an image was taken, with the scope limited to images in Sweden and Switzerland. We created a dataset of 180K images and tried three different strategies to tackle the problem **1)** regression-based using multi-task learning, **2)** classification-based using bottom-up cell creation, and **3)** retrieval-based using contrastive learning. The first two methods are novel ideas, and the third method is inspired by the GeoCLIP method.

The best performing model on country-level and low-resolution predictions was the the regression model. We demonstrated that multi-task learning was beneficial – the accuracy of the country classifier head improved when adding a second head to predict GPS location, using a multi-task loss function. Future work could explore further improving the country classification by enhancing image features with scene data.

The performance of the retrieval approach increased significantly for finer resolutions when adding a layer of random fourier series before the MLP.

Furthermore, we designed a new bottom-up cell creation method that clusters images based on both location and visual appearance, improving the accuracy of high-resolution predictions.

To obtain the best results for several resolutions, you would have to combine the approaches and use separate models for different resolutions and potentially scene types.

Overall, we conclude that MLPs are not well suited for converting features to GPS coordinates (regression) or coordinates to features (retrieval). We are able to confirm GeoClip's finding that adding the fourier series is crucial for higher resolutions. The noisy data makes it hard for the MLP to learn a good approximation of GPS coordinates from the features of the image encoder. Others have tackled this problem by augmenting the features with scene classification or semantic maps, which would probably improve our results as well.

Our bottom-up cell construction algorithm showed that clustering with feature cohesion in mind improves results. While we only implemented WCSS ranking with greedy cell selection, a full clustering algorithm optimizing overall allocation should obtain even better results. As future work, we propose to project both location and visual features into an N-dimensional space, using a new hyperparameter α to balance the influence of features vs. location, then employing an established clustering algorithm like OPTICS [8], as used for example in [14] to perform the clustering.

A. Appendix

B. Contributions and Acknowledgements

Both participants contributed an equal amount regarding work, report wring, design of the approaches and the experiments. Björn focused on the development of the regression and retrieval methods and Matthias focused on preparing the data sets and the development of the cell-based methods. For scoping of the project we discussed with the project mentor.

References

[1] Flickr: https://www.flickr.com/.

[2] Geoclip: https://github.com/vicentevivan/geo-clip.

[3] Geopandas: https://geopandas.org/en/stable/.

[4] Pandas: https://pandas.pydata.org/.

[5] Pytorch: https://pytorch.org/.

[6] S2 geometry library: http://s2geometry.io.

[7] Scikit: https://scikit-learn.org/stable/.

[8] M. Ankerst, M. Breunig, P. Kröger, and J. Sander. Optics: Ordering points to identify the clustering structure. volume 28, pages 49–60, 06 1999.

[9] R. Basri, M. Galun, A. Geifman, D. Jacobs, Y. Kasten, and S. Kritchman. Frequency bias in neural networks for input of non-uniform density, 2020.

[10] T. P. Bojan Šavrič and B. Jenny. The equal earth map projection. *International Journal of Geographical Information Science*, 33(3):454–465, 2019.

[11] V. V. Cepeda, G. K. Nayak, and M. Shah. Geoclip: Clip-inspired alignment between locations and images for effective worldwide geo-localization, 2023.

[12] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations, 2020.

[13] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[14] L. Haas, M. Skreta, S. Alberti, and C. Finn. Pigeon: Predicting image geolocations, 2024.

[15] J. Hays and A. Efros. Im2gps: Estimating geographic information from a single image. pages 1 – 8, 07 2008.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[17] E. Müller-Budack, K. Pustu-Iren, and R. Ewerth. Geolocation estimation of photos using a hierarchical model and scene classification. In *European Conference on Computer Vision*, 2018.

[18] S. Pramanick, E. M. Nowara, J. Gleason, C. D. Castillo, and R. Chellappa. Where in the world is this image? transformer-based geo-localization in the wild, 2022.

[19] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021.

[20] R. Ranjan, V. M. Patel, and R. Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition, 2017.

[21] P. H. Seo, T. Weyand, J. Sim, and B. Han. Cplanet: Enhancing image geolocalization by combinatorial partitioning of maps, 2018.

[22] H. M. Sergieh, D. Watzinger, B. Huber, M. Döller, E. Egyed-Zsigmond, and H. Kosch. World-wide scale geotagged image dataset for automatic image annotation and reverse geotagging. In *ACM SIGMM Conference on Multimedia Systems*, 2014.

[23] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions, 2014.

[24] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, and R. Ng. Fourier features let networks learn high frequency functions in low dimensional domains, 2020.

[25] J. Theiner, E. Mueller-Budack, and R. Ewerth. Interpretable semantic photo geolocation, 2021.

[26] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding, 2019.

[27] T. Weyand, I. Kostrikov, and J. Philbin. *PlaNet - Photo Geolocation with Convolutional Neural Networks*, page 37–55. Springer International Publishing, 2016.