Exploiting the Earth's Spherical Geometry to Geolocate Images

Mike Izbicki[1], Evangelos E. Papalexakis[2], and Vassilis J. Tsotras[2]

[1] Claremont McKenna College, Claremont, CA, USA
mike@izbicki.me
[2] University of California Riverside, Riverside, CA, USA
{epapalex,tsotras}@cs.ucr.edu

Abstract. Existing methods for geolocating images use standard classification or image retrieval techniques. These methods have poor theoretical properties because they do not take advantage of the earth's spherical geometry. In some cases, they require training data sets that grow exponentially with the number of feature dimensions. This paper introduces the *Mixture of von-Mises Fisher* (MvMF) loss function, which is the first loss function that exploits the earth's spherical geometry to improve geolocation accuracy. We prove that this loss requires only a dataset of size linear in the number of feature dimensions, and empirical results show that our method outperforms previous methods with orders of magnitude less training data and computation.

Keywords: Geolocation; Flickr; Deep Learning; von Mises-Fisher

1 Introduction

Consider the two images below:

Most people recognize that the left image is of the Eiffel Tower, located in Paris, France. A trained expert can further recognize that the right image is a replica of the Eiffel Tower. The expert uses clues in the image's background (e.g. replicas of other famous landmarks, tall cement skyscrapers) to determine that this image was taken in Shenzhen, China. We call these images *strongly localizable* because

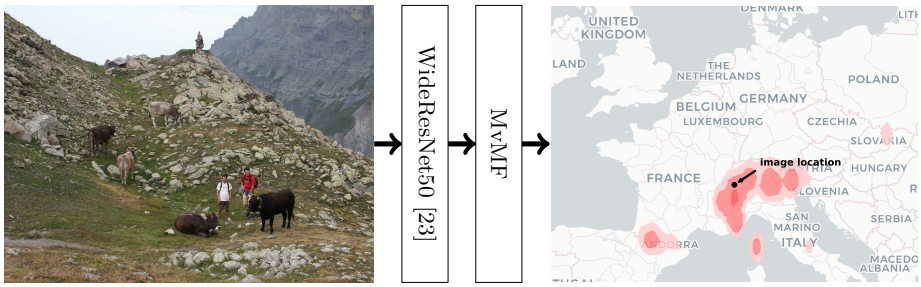

Fig. 1. To geolocate an image, we first generate features using the WideResNet50 [23], then pass these features to our novel *mixture of von Mises-Fisher* (MvMF) output layer. The MvMF outputs a probability distribution over the earth's surface, and is particularly well-suited for visualizing the output of hard-to-geolocate images.

the images contain all the information needed to exactly geolocate the images. Existing geolocation algorithms work well on strongly localizable images. These algorithms [e.g. 1, 22, 21, 16, 15] use deep neural networks to extract features, and can therefore detect the subtle clues needed to differentiate these images.

Most images, however, are only *weakly localizable* because the image does not contain enough information to exactly geolocate it. Consider the image in Figure 1 of two men hiking. An expert can use clues like the geology of the mountains, the breed of cattle, and the people's appearance to determine that this image was taken in the Alps. But the Alps are a large mountain range, and there is simply not enough information in the image to pinpoint exactly where in the Alps the image was taken. Existing geolocation algorithms are overconfident when predicting locations for these images. These algorithms use either image retrieval [7, 8, 1, 21] or classification methods [22, 16, 15] to perform geolocation, and these procedures do not take advantage of the earth's spherical geometry. They therefore cannot properly represent the ambiguity of these weakly localizable images.

In this paper, we introduce the MvMF output layer for predicting GPS coordinates with deep neural networks. The MvMF has three advantages compared to previous methods:

1. The MvMF takes advantage of the earth's spherical geometry and so works with both strongly and weakly localizable images.
2. The MvMF has theoretical guarantees, whereas no previous method has a theoretical analysis.
3. The MvMF interpolates between the image retrieval and classification approaches to geolocation, retaining the benefits of both with the drawbacks of neither.

In our experiments, we use the WideResNet50 [23] convolutional neural network to generate features from images, but we emphasize that any deep neural network can be used with the MvMF layer. We provide TensorFlow code for the MvMF layer at `https://github.com/mikeizbicki/geolocation`.

In Section 2, we review prior work on the image geolocation problem. We focus our exposition on how previous methods ignore the earth's spherical geometry, and how this causes them to require more training data. Section 3 presents the MvMF output layer to fix these problems, while Section 4 provides the experimental results. Section 5 concludes the paper with a discussion on other possible application areas for the MvMF output layer.

2 Prior Work

Prior image geolocation methods use either image retrieval or classification techniques. This section describes the limitations of these techniques. We use standard theoretical results to show that these techniques require excessively large training datasets, and conduct novel experiments to verify that these theoretical flaws affect real world performance.

2.1 Image Retrieval

Im2GPS [7, 8] was the first image geolocation system. The most important component of this system is a large database of images labelled with GPS coordinates and manually constructed features. To determine the location of a new image, Im2GPS performs a k-nearest neighbor query in the database, and outputs the average GPS coordinates of the returned images. Im2GPS-deep [21] significantly improved the results of the Im2GPS system by using deep neural networks to generate features.

These image retrieval systems have two basic disadvantages due to the curse of dimensionality. First, they have poor theoretical guarantees. Let d denote the dimensionality of the image feature vector ($d \approx 1 \times 10^5$ for Im2GPS and $d = 512$ for Im2GPS-deep). Then standard results for k-nearest neighbor queries show that in the worst case, a database with $\Omega(d^{d/2})$ images is needed for accurate queries (see Theorem 19.4 of [17]); that is, the amount of training data should grow exponentially with the number of feature dimensions. This is unrealistic for the feature dimensions used in practice. Second, image retrieval systems are slow. The nearest neighbor search is performed over millions of images, and because the dimensionality of the space is large, data structures like kd-trees do not speed up the search as much as we would like.

In Section 3.3 below, we show that the MvMF model this paper introduces can be interpreted as an image retrieval system. The MvMF, however, takes advantage of the earth's spherical geometry to avoid the curse of dimensionality.

2.2 Classification

Classification-based geolocation methods were introduced to overcome the performance limitations of image retrieval methods. The basic idea is as follows. First, the surface of the earth is partitioned into a series of classes. Then the standard cross entropy loss is used to classify images. The estimated position is

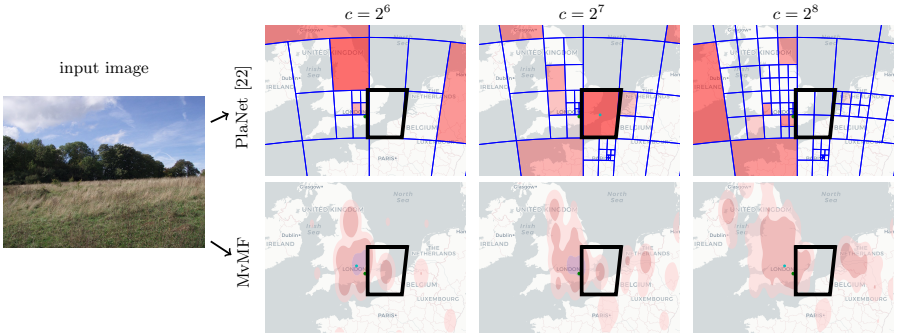

Fig. 2. The PlaNet [22] method's performance is highly sensative to the number of classes c. Consider the highlighted region. When $c = 2^6$, PlaNet assigns no probability to the region. (Brighter red indicates classes with higher probability.) When $c = 2^7$, PlaNet has split many other cells, causing the probability of the highlighted region to increase. When $c = 2^8$, PlaNet splits the highlighted region, causing the probability to drop again. This effect is exaggerated for weakly localizable images because many classes should be assigned high probability. In comparison, when the number of classes increases for the MvMF loss, the output smoothly takes on the shape of the underlying geography, which is the desired output for a weakly localizable image of grass.

the center of the predicted class. These methods have better theoretical guarantees than the image retrieval methods and are faster because they avoid the expensive nearest neighbor queries. Unfortunately, these methods have two flaws: tuning the number of classes is difficult, and the cross entropy loss is not well correlated with geolocation performance.

We illustrate these flaws on the PlaNet algorithm [22], which is the earliest and most influential example of classification-based geolocation methods. The algorithm divides the world into a series of classes using an adaptive partitioning scheme based on Google's S2 geometry library. The classes are constructed according to the following recursive procedure: the world is initially divided into 6 classes; the class with the most images is then subdivided into smaller classes; and this procedure is repeated until the desired number of classes c is reached. Figure 2 shows an example class tiling generated using the PlaNet method for three different values of c.

The number of classes c is a hyperparameter that must be manually tuned,[3] and tuning this parameter is an instance of the classic bias-variance trade off. Recall that the *bias* of a model (also called the *approximation error*) is the error of the best possible model in a given class, and the *variance* (also called the *estimation error*) is the statistical error induced by the finite size of the training set. We make the following claim about PlaNet's geolocation method.

Claim 1 *Increasing the number of classes c reduces the model's bias but increases the model's variance.*

[3] The original PlaNet paper chose a value of $c \approx 2^{15}$.

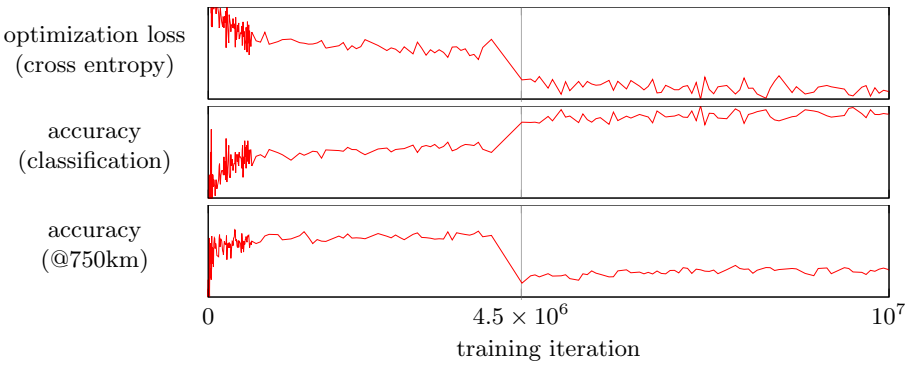

Fig. 3. Previous classification-based geolocation methods minimize the cross entropy loss [22, 16, 15], but these plots show that minimizing the cross entropy does not necessarily improve geolocation accuracy. (*Top*) Stochastic gradient descent directly minimizes the cross entropy loss, so the cross entropy decreases as training progresses. (*Middle*) The cross entropy loss is a convex surrogate for the classification accuracy, so classification accuracy increases as training progresses. (*Bottom*) The cross entropy is not a convex surrogate for the accuracy @750km (which is the fraction of data points whose estimated location is within 750km of their true location), so the geolocation accuracy does not necessarily increase as training progresses. In this particular run, the cross entropy loss at iteration 4.5×10^6 improves dramatically, but the geolocation accuracy worsens dramatically. This effect can be observed at all distance scales and for all hyperparameter values.

To understand this claim, observe that when c is small, the geographic area of each class is large, so fine-grained predictions are not possible, and the model has large bias. Increasing the number of classes c reduces the size of each class, allowing more fine-grained predictions, and reducing the model's bias. To see how c effects the model's variance, we appeal to Theorem 4 of Hazan et al. [9] that shows that the variance of a classifier using the cross entropy loss grows as $\Omega(cd)$. Therefore, as c increases, the variance must increase as well. Finding the optimal value of c is a difficult balancing act, as shown in Figure 2.

In order to reduce the variance inherent in classification methods, the CPlaNet method [16] and the ISN method [15] use multiple cross entropy output layers to reduce the total number of classes needed. Both methods lack theoretical guarantees, and the optimal number of classes c still requires careful tuning.

A second problem with PlaNet, CPlaNet, and ISN is these methods all use the cross entropy loss for training. The cross entropy is closely associated with classification accuracy, but as we show in Figure 3, is not necessarily correlated with geolocation performance. The fundamental problem is that the cross entropy loss does not incorporate knowledge about the earth's spherical geometry.

In Section 3.2 below, we show that our MvMF method has an interpretation as a classification-based geolocation method. In contrast to all previous methods, however, the MvMF uses a loss function that exploits the earth's spherical ge-

ometry and so is highly correlated with geolocation performance. We also show that the variance of the MvMF grows as $O(d)$, and so increasing the number of classes c reduces the model's bias without increasing the variance.

3 Geolocation via the MvMF

The MvMF is the first geolocation method that exploits the earth's spherical geometry, and it is specifically designed to overcome the disadvantages of the image retrieval and classification methods for geolocation. In this section, we first introduce the MvMF as a probabilistic model, then describe two alternative interpretations of the MvMF as a classification model with a non-standard loss or as an image retrieval model using non-standard features. A powerful property of the MvMF model is that it can interpolate between the classification and image retrieval approaches to geolocation, getting the best of both techniques while avoiding the limitations of both. We prove that when the MvMF's parameters are properly initialized, only $O(d)$ training data points are needed.

3.1 The probabilistic interpretation

This subsection formally introduces the MvMF output layer as a mixture of von Mises-Fisher distributions. Then we describe the training and inference procedures.

The *von Mises-Fisher* (vMF) distribution is one of the standard distributions in the field of directional statistics, which is the study of distributions on spheres. The vMF can be considered the spherical analogue of the Gaussian distribution [e.g. 13] and enjoys many of the Gaussian's nice properties. Thus, the *mixture of vMF* (MvMF) distribution can be seen as the spherical analogue of the commonly used Gaussian mixture model (GMM). While the MvMF model has previously been combined with deep learning for clustering [5] and facial recognition [6], we are the first to combine the MvMF and deep learning to predict GPS coordinates.

Formally, the vMF distribution is parameterized by the *mean direction* $\mu \in \mathbb{S}^2$, and the *concentration parameter* $\kappa \in \mathbb{R}^+$. The density is defined for all points $\mathbf{y} \in \mathbb{S}^2$ as

$$\text{vMF}(\mathbf{y}; \mu, \kappa) = \frac{\kappa}{\sinh \kappa} \exp(\kappa_i \mu^\top \mathbf{y}). \tag{1}$$

An important property of the vMF distribution is that it is symmetric about μ for all $\mu \in \mathbb{S}^2$. As shown in Figure 4, a gaussian distribution over GPS coordinates does not account for the earth's spherical geometry, and is therefore not symmetrical when projected onto the sphere.

The *mixture of vMF* (MvMF) distribution is a convex combination of vMF distributions. If the mixture contains c component vMF distributions, then it is parameterized by a collection of mean directions $M = (\mu_1, ..., \mu_c)$, a collection of concentration parameters $K = (\kappa_1, ..., \kappa_c)$, and a vector of mixing weights $\Gamma \in \mathbb{R}^c$ satisfying $\sum_{i=1}^c \Gamma_i = 1$. Notice that we use capital Greek letters for the parameters of the mixture distribution and lowercase Greek letters for the

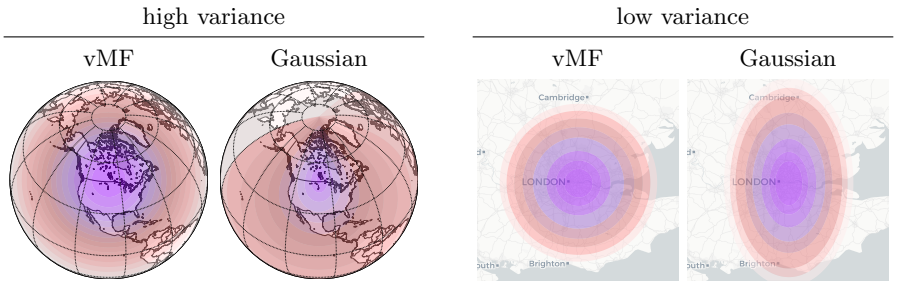

Fig. 4. The vMF distribution takes into account the curvature of the earth's surface, and so contour lines are equidistant from the center at all scales and locations. The Gaussian distribution over GPS coordinates, in contrast, becomes elongated far from the equator, and has discontinuities at the poles and at longitude $\pm 180°$.

parameters of the corresponding component distributions. The density is given by

$$\text{MvMF}(\mathbf{y}; M, K, \Gamma) = \sum_{i=1}^{c} \Gamma_i \, \text{vMF}(\mathbf{y}, M_i, K_i). \tag{2}$$

To construct the MvMF loss function from this density, we assume that the mean direction and concentration parameters do not depend on the input features. The mixing weights are parameterized using the standard softmax function as

$$\Gamma_i(\mathbf{x}; W) = \frac{\exp(-\mathbf{x}^\top \mathbf{w}_i)}{\sum_{j=1}^{c} \exp(-\mathbf{x}^\top \mathbf{w}_j)}. \tag{3}$$

where $W = (\mathbf{w}_1, ..., \mathbf{w}_c)$ and each $\mathbf{w}_i \in \mathbb{R}^d$. Taking the negative log of Equation (2) and substituting Γ_i gives us the final MvMF loss:

$$\ell_{\text{MvMF}}(\mathbf{x}, \mathbf{y}; M, K, W) = -\log \sum_{i=1}^{c} \Big(\Gamma_i(\mathbf{x}; W) \, \text{vMF}(\mathbf{y}, M_i, K_i) \Big). \tag{4}$$

When training a model with the MvMF loss, our goal is to find the best values for M, K, and W for a given dataset. Given a training dataset $(\mathbf{x}_1, \mathbf{y}_1), ..., (\mathbf{x}_n, \mathbf{y}_n)$ the training procedure solves the optimization

$$\hat{M}, \hat{K}, \hat{W} = \underset{M,K,W}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \ell_{\text{MvMF}}(\mathbf{x}_i, \mathbf{y}_i; M, K, W). \tag{5}$$

Training mixture models is difficult due to their non-convex loss functions, and good initial conditions are required to ensure convergence to a good local minimum. We use the following initialization in our experiments: initialize the W randomly using the Glorot method [4]; initialize μ_i to the center of the ith class used by the PlaNet method; and initialize all κ_i to the same initial value κ^0. We suggest using $\kappa^0 = \exp(16)$ based on experiments in Section 4.

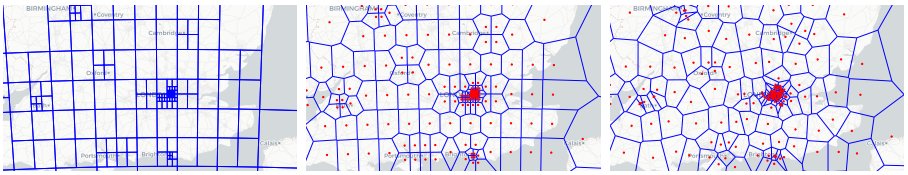

Fig. 5. (*Left*) Classes near London created using the PlaNet method. (*Middle*) Classes of the MvMF method with the μ_i initialized from the centers of the PlaNet classes. (*Right*) After training with the MvMF loss, the μ_i have shifted slightly to better fit the data, resulting in a new class partition.

The estimated GPS coordinate $\hat{\mathbf{y}}$ of a feature vector $\mathbf{x}$ is the coordinate with minimum loss. That is,

$$\hat{\mathbf{y}} = \operatorname*{arg\,min}_{\mathbf{y} \in \mathbb{S}^2} \ell_{\text{MvMF}}(\mathbf{x}, \mathbf{y}; M, K, W). \tag{6}$$

Notice that this optimization is distinct from (5). This optimization does not get evaluated during model training, but only during inference. This optimization is non-convex, and may have up to c distinct local minima. Algorithms exist for finding the minima of mixture models [3], but these algorithms require significant computation. Calculating $\hat{\mathbf{y}}$ for a single image may be feasible, but calculating $\hat{\mathbf{y}}$ for an entire test set is prohibitive. The classification interpretation of the MvMF loss presents an easy to interpret, computationally more efficient method for inference.

3.2 Interpretation as a classifier

The MvMF model can be interpreted as a classification model where each component represents a class. The mixture weights $\Gamma_i(\mathbf{x}, W)$ then become the probability associated with each class. The estimated location $\tilde{\mathbf{y}}$ is then the mean direction of the class with largest weight. Formally,

$$\tilde{\mathbf{y}} = \mu_{\tilde{i}}, \quad \text{where} \quad \tilde{i} = \operatorname*{arg\,max}_{i \in \{1,\dots,c\}} \Gamma_i(\mathbf{x}, W). \tag{7}$$

Because this optimization is over a discrete space, it is extremely fast. When the mean directions M are initialized using the centers of the PlaNet classes, then there is a one-to-one correspondence between the MvMF classes and the PlaNet classes, albeit with the class shapes differing slightly (see Figure 5). In our experiments in Section 4, we use $\tilde{\mathbf{y}}$ as the estimated position.

Another advantage of the MvMF classes over the PlaNet classes is that the MvMF classes are fully parameterized by M. This means by jointly optimizing both W and M, we can learn not only which classes go with which images, but where on the earth the classes should be located.

3.3 Interpretation as an image retrieval method

We now describe how the MvMF model interpolates between classification models and image retrieval models. Recall that

$$\Gamma_i(\mathbf{x}, W) = \frac{\exp(-\mathbf{x}^\top \mathbf{w}_i)}{\sum_{j=1}^{c} \exp(-\mathbf{x}^\top \mathbf{w}_j)} \propto \exp(-\mathbf{x}^\top \mathbf{w}_i). \tag{8}$$

Solving for the $\tilde{i}$ in Equation (7) that maximizes Γ_i is therefore equivalent to finding the $\mathbf{w}_i$ that minimizes the inner product with $\mathbf{x}$. Minimum inner product search is a well studied problem, and in particular, it can be reduced to nearest neighbor search [2]. Therefore, when the number of classes equals the number of data points (i.e. $c = n$), and for each i we have $\mu_i = \mathbf{y}_i$ and $\mathbf{w}_i = \mathbf{x}_i$, then solving Equation (7) to find the output class is equivalent to solving a nearest neighbor problem.

3.4 Analysis

Our analysis states that the MvMF's estimation error converges to zero at a rate of $O(\sqrt{d/n})$, where d is the number of feature dimensions and n the number of data points. This is in contrast to nearest neighbor methods (which converge at the exponential rate $\Omega(dn^{1/d})$ [Theorem 19.4 of 17]), and the cross entropy loss (which converges at a rate of $\Omega(\sqrt{cd/n})$ [Theorem 4 of 9]). Because c and d are both large in the geolocation setting, the MvMF loss requires significantly less training data to converge.

We use three assumptions to simplify our analysis. The first assumption states that our analysis only applies to the convergence of the W parameter. We argue that this is a mild assumption because W is the most important parameter to learn. The second assumption states that our features have bounded size, which is true of all image deep neural networks. The final assumption states that our analysis requires a good initial parameter guess. This is unsurprising as mixture models are known to be highly sensitive to their initial conditions. We now state these assumptions formally and describe their implications in more detail.

Assumption 1 *We optimize only W using stochastic gradient descent (SGD). In particular, we do not optimize M, K, or the deep network generating features.*

SGD is an iterative algorithm, where each iteration t considers only a single data point $(\mathbf{x}_t, \mathbf{y}_t)$ sampled uniformly at random from the underlying data distribution. On each iteration t, the model weights are denoted by W_t. These weights are updated according to the rule

$$W_{t+1} = W_t - \eta_t \frac{d}{dW_t} \ell_{\text{MvMF}}(\mathbf{x}_t, \mathbf{y}_t; M, K, W_t)$$

starting from some initial W_0. The variable η_t is called the step size.

SGD is the most common algorithm for optimizing deep neural networks, and in practice it is common to train all parameters of a network at the same time. Analyzing such training procedures, however, is a difficult open problem due to the highly non-convex nature of neural networks. To better understand these systems, it is common to analyze the convergence of a single parameter while holding all others fixed, and that is our strategy.

The W parameter is the most important to analyze because it determines which class an image will be assigned to. The M and K parameters determine properties of the classes, and their convergence does not significantly effect geolocation. The M parameter is initialized to the classes of the PlaNet method, so improvements in M can only make it better than the PlaNet method. And the K value does not affect the classification-based inference in Equation (7).

Assumption 2 *For all* $\mathbf{x}$*, we have* $\|\mathbf{x}\| \leq \sqrt{d}$*.*

This is a standard assumption for the analysis of SGD algorithms, and it is equivalent to assuming that the individual features are bounded. Let $\mathbf{x}_i$ denote the ith feature in $\mathbf{x}$, and assume that each feature $\mathbf{x}_i \in [-1, 1]$. Then,

$$\|\mathbf{x}\|^2 = \sum_{i=1}^{d} \mathbf{x}_i^2 \leq \sum_{i=1}^{d} 1 = d.$$

It is common to scale images so that each pixel value is in the range $[-1, 1]$, and all deep neural networks are designed to keep their output features bounded.

Assumption 3 *Let* W^* *be a (possibly local) minimizer of* ℓ_{MvMF}*. Let* $\mathcal{W}$ *be a convex subset of* $\mathbb{R}^{d \times c}$ *containing* W^* *such that* ℓ_{MvMF} *is convex in* $\mathcal{W}$*. Finally, for all time steps* t*, we assume that* $W_t \in \mathcal{W}$*.*

This is our most complicated assumption. Informally, it states that we limit our analysis of ℓ_{MvMF} to a convex region around a possibly local minimum W^*. We must limit our analysis to this convex region because existing analyses of SGD work only for convex losses and the ℓ_{MvMF} is non-convex.

We argue this is not a limiting assumption for two reasons. First, SGD will eventually converge to a region satisfying the properties of $\mathcal{W}$. To see this, observe that SGD will converge to a local minimum with probability 1, and (due to the smoothness of ℓ_{MvMF}) every local minimum of ℓ_{MvMF} is contained in a convex set $\mathcal{W}$ over which ℓ_{MvMF} is convex. Second, if SGD does not converge to a sufficiently good local minimum, the procedure can be repeated from a different random initialization until a good local minimum is reached. In our experiments, however, we found that it was never necessary to rerun SGD from a different initialization.

Theorem 1. *Under Assumptions 1-3, at each iteration* t*, the MvMF's estimation error is bounded by*

$$\mathbb{E}\left(\ell_{MvMF}(\mathbf{x}, \mathbf{y}; M, K, W_t) - \ell_{MvMF}(\mathbf{x}, \mathbf{y}; M, K, W^*)\right) \leq 2\sqrt{\frac{d}{n}}. \tag{9}$$

method	training images	features	output method
Im2GPS [7, 8]	6×10^6	custom	retrieval
Im2GPS-deep [21]	28×10^6	VGG [18]	retrieval
PlaNet [22]	126×10^6	InceptionV1 [19]	classification
CPlaNet [16]	30×10^6	InceptionV3 [20]	classification (modified)
ISN [15]	16×10^6	ResNet [10] + custom	classification (modified)

Table 1. Details of the training environment for previous work on image geolocation.

We sketch the proof of Theorem 1 here and defer a full proof to the Appendix for space reasons. Standard results show that the estimation error of SGD is bounded by $\rho/\sqrt{n}$, where ρ is an upper bound on the gradient of the loss function. We show that for the MvMF loss, the gradient is bounded by $\rho \leq \sqrt{d}$. The cross entropy loss used by other classification-based geolocation methods has worse performance guarantees because its gradient is bounded by $\rho \leq \sqrt{cd}$. The $\sqrt{cd/n}$ convergence rate for the cross entropy loss is known to be tight [9], and so the MvMF loss has strictly better convergence.

4 Experiments

The empirical performance of an image geolocation system is determined by three factors: the training data, the feature generation method, and the output method. Prior work on image geolocation introduced improvements in all three areas, making it difficult to determine exactly which improvement was responsible for better performance (see Table 1 for a summary). In particular, no prior work attempts to isolate the effects of the output method, and no prior work compares different output methods side by side.

In our experiments, we follow a careful procedure to generate a standard training dataset with standard features so that we can isolate the effects of the output method. Because prior work does not follow this procedure, an exhaustive empirical comparison would require reimplementing all previous methods from scratch. This is infeasible from both a manpower and computational perspective, so we focus our comparison on the PlaNet method, since it is representative of classification methods using the cross entropy loss. We show that the cross entropy-based methods require careful tuning of the hyperparameter c, but that our MvMF's performance always improves when increasing the number of classes c (as our theory predicts). This leads to significantly better performance of the MvMF method.

4.1 Procedure

We now describe our standardized training procedure. We pay special attention to how it improves upon previous training procedures for comparing output methods.

Training Data. The previous methods' training datasets are not only of different sizes, but also sampled from entirely different distributions. The Im2GPS [7, 8] and Im2GPS-deep [21] methods download data from Flickr, then filter the images using user specified tags to remove weakly localizable images. For example, it is assumed that images with the tag `#birthday` are likely to be of indoor scenes with few geographic clues, and so images with this tag are removed from the dataset. The CPlaNet [16] and ISN [15] datasets are also acquired from Flickr, and they introduce other criteria for filtering the data to ensure only high quality images are included. The PlaNet [22] dataset uses geotagged images crawled from all over the web with no filtering. The dataset is much larger, but is significantly harder to train from, because the data is noisier with more weakly localizable images. Most of these datasets are proprietary and not publicly available.

For our training data, we use a previously existing publicly available dataset of geotagged images from Mousselly et. al. [14]. This dataset contains about 6 million images crawled from Flickr,[4] and the crawl was designed to be as representative as possible of Flickr's image database. The only filtering the dataset performed was to remove low resolution images. This dataset therefore comes from a distribution more similar to the PlaNet dataset than the other datasets.

Features. Even if all previous models had been trained on the same data, it would still be impossible to directly compare the efficacy of output methods because each model uses different features. We use the WideResnet50 model [23] to generate a standard set of features in our experiments. WideResnet50 was originally trained on the ImageNet dataset for image classification, so we "fine-tune" the model's parameters to the geolocation problem. We chose the WideResnet50 model because empirical results show that fine-tuning works particularly well on resnet models [12], and the WideResnet50 is the best performing resnet model.

Fine-tuning a model is computationally cheaper than training from scratch, but it is still expensive. We therefore fine-tune the model only once, and use the resulting features in all experiments. To ensure that our fine-tuned features do not favor the MvMF method, we create a simple classification problem to fine-tune the features on. We associate each image with the country the image was taken in or "no country" for images from Antarctica or international waters. In total, this gives us a classification problem with 194 classes. We then fine-tune the WideResnet50 model for 20 epochs using the cross entropy loss, WideResnet50's standard feature augmentation, and the Adam [11] variant of SGD with a learning rate of $1 \times 10^{-}5$. This took about 2 months on a 4 CPU system with a Tesla K80 GPU and 64GB of memory. Because this fine-tuning procedure uses a cross entropy loss, the resulting features should perform especially well with cross entropy geolocation methods. Nonetheless, we shall see that the MvMF loss still outperforms cross entropy methods.

[4] The dataset originally contained about 14 million images, but many of them have since been deleted from Flickr and so were unavailable to us.

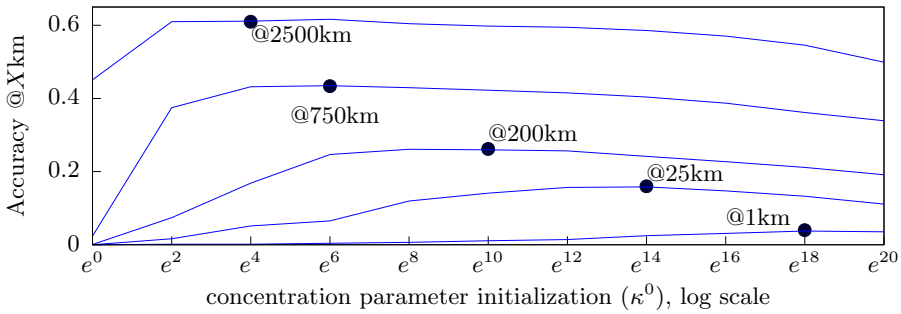

Fig. 6. Higher values of κ^0 result in better performance at fine grained prediction, and lower values of κ^0 result in better performance for course-grained prediction.

4.2 Results

We perform 3 experiments using the standardized training data and features described above.

Tuning the MvMF's hyperparameters. In this experiment, we set $c = 2^{15}$ and train MvMF models with $\kappa^0 = 0...20$. The results are shown in Figure 6. Accuracy @Xkm is a standard method for evaluating the performance of a geolocation system, and is equal to the fraction of data points whose estimated location is within Xkm of the true location. (Higher values are better.) For small X, Accuracy @Xkm measures the ability to geolocate strongly localizable images, and for large X, Accuracy @Xkm measures the ability to geolocate weakly localizable images.

We see that large values of κ^0 cause better geolocation for strongly localizable images, and small values of κ^0 cause better geolocation for weakly localizable images. This behavior has an intuitive explanation. When κ^0 is small, the variance of each component vMF distribution is large. So on each SGD step, weights from vMF components that are far away from the training data point will be updated. If the image is weakly localizable, then there are many locations where it might be placed, so many component weights should be updated. Conversely, when κ^0 is large, the component variances are small, and so only a small number of components get updated with each SGD step. Strongly localizable images can be exactly located to a small number of components, and so only a few components should be updated. We suggest using a value of $\kappa^0 = 16$ as a good balance, and use this value in all other experiments.

Tuning the number of classes *c*. This experiment demonstrates that c must be carefully tuned in the PlaNet method, but that increasing c always increases performance of the MvMF method. We emphasize that the original PlaNet paper [22] does not report results on the tuning of c, and so observing these limitations of the PlaNet method is one of the contributions of our work.

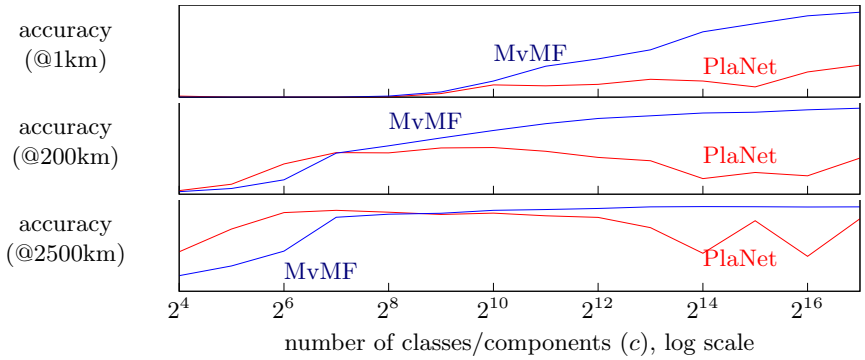

Fig. 7. The performance of the MvMF output layer increases monotonically as we increase the number of mixture components c, whereas the perfomance of PlaNet depends unpredictably on c.

We train a series of models using the MvMF loss and PlaNet loss, varying c from 2^4 to 2^{17}. Theoretically, both methods support class sizes larger than $c = 2^{17}$, but our GPU hardware only had enough memory for 2^{17} classes. Figure 7 shows the results. For all X, we observe that PlaNet's performance is highly unpredictable as c varies, but the MvMF method always has improved accuracy as c increases.

Note that Figure 2 shows qualitatively why the PlaNet method is more sensitive to c than the MvMF. In that figure, as c increases and classes get split, the probability that was previously assigned to those classes gets completely reallocated. Similarly, Figure 3 illustrates a single training run of the PlaNet method. Because the cross entropy loss does not directly optimize the desired outcome (geolocation), improvements to the cross entropy loss sometimes result in worse geolocation performance.

Fine-tuned performance. In this experiment, we select several cross entropy and MvMF models and perform a second round of fine-tuning, this time with their true loss functions. We fine-tune with the Adam optimizer running for 5 epochs with learning rate 1×10^{-5}, which takes approximately 2 weeks per model on a single GPU. We evaluate the resulting model against the standard Im2GPS test set introduced by [7]. The results are shown in Table 2. When using the standardized training data and features, the MvMF loss significantly outperforms the cross entropy loss.

In Table 2, we also include results reported in the original PlaNet paper [22]. These results use a training data set that is 2 orders of magnitude larger than the standardized training set, and so have significantly better performance than the cross entropy loss on the standard training set. This illustrates that the training data has a huge impact on the final model's performance. Surprisingly, the MvMF loss trained on standardized training set with only 6 million data points outperforms the PlaNet method trained on 126 million images. Other

			accuracy @				
loss	data/features	c	1km	25km	200km	750km	2500km
cross entropy	PlaNet [22]	$\approx 2^{15}$	**8.4**	24.5	37.6	53.6	71.3
cross entropy	standardized	2^{13}	1.0	4.1	10.1	24.8	44.8
cross entropy	standardized	2^{15}	0.6	2.0	7.3	26.1	49.9
cross entropy	standardized	2^{17}	1.8	6.0	11.8	27.9	51.3
MvMF	standardized	2^{13}	4.6	28.0	35.4	50.5	73.4
MvMF	standardized	2^{15}	6.0	31.2	41.1	58.0	75.7
MvMF	standardized	2^{17}	**8.4**	**32.6**	**39.4**	**57.2**	**80.2**

Table 2. Results on the Im2GPS test set [7]. The MvMF loss significantly outperforms the cross entropy loss at all distances when using standardized data and features. The MvMF loss trained on the standardized features even outperforms the PlaNet method, which was trained on a much larger dataset and required significantly more computation.

models have been evaluated on the Im2GPS test set as well [e.g. 7, 8, 22, 21, 16, 15], but we do not report their performance here because we could not do a fair comparison where all models were trained using the same training data and features.

5 Conclusion

The MvMF is the first method for image geolocation that takes advantage of the earth's geometry. The MvMF has better theoretical guarantees than previous image retrieval and classification methods, and these guarantees translate into better real world performance. We emphasize that the MvMF layer can be applied to any geolocation problem, not just image geolocation.

Acknowledgments

We thank an anonymous reviewer for identifying a mistake in the first version of our proof. E. Papalexakis was supported by the Department of the Navy, Naval Engineering Education Consortium under award no. N00174-17-1-0005 and the National Science Foundation CDS&E Grant no. OAC-1808591. V. Tsotras was supported by National Science Foundation grants IIS-1527984 and SES-1831615.

References

1. Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, June 2016.

2. Yoram Bachrach, Yehuda Finkelstein, Ran Gilad-Bachrach, Liran Katzir, Noam Koenigstein, Nir Nice, and Ulrich Paquet. Speeding up the xbox recommender system using a euclidean transformation for inner-product spaces. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 257–264. ACM, 2014.
3. Miguel A. Carreira-Perpinan. Mode-finding for mixtures of gaussian distributions. *TPAMI*, 22(11):1318–1323, 2000.
4. Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AIStats*, pages 249–256, 2010.
5. Siddharth Gopal and Yiming Yang. Von Mises-Fisher clustering models. In *ICML*, pages 154–162, 2014.
6. Md Hasnat, Julien Bohné, Jonathan Milgram, Stéphane Gentric, Liming Chen, et al. von mises-fisher mixture model-based deep learning: Application to face verification. *arXiv preprint arXiv:1706.04264*, 2017.
7. James Hays and Alexei A Efros. Im2gps: estimating geographic information from a single image. In *CVPR*. IEEE, 2008.
8. James Hays and Alexei A Efros. Large-scale image geolocalization. In *Multimodal Location Estimation of Videos and Images*, pages 41–62. Springer, 2015.
9. Elad Hazan, Tomer Koren, and Kfir Y Levy. Logistic regression: Tight bounds for stochastic and online optimization. In *COLT*, 2014.
10. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, pages 630–645. Springer, 2016.
11. Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
12. Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? *arXiv preprint arXiv:1805.08974*, 2018.
13. K.V. Mardia and P.E. Jupp. *Directional Statistics*. 2009.
14. Hatem Mousselly-Sergieh, Daniel Watzinger, Bastian Huber, Mario Döller, Elöd Egyed-Zsigmond, and Harald Kosch. World-wide scale geotagged image dataset for automatic image annotation and reverse geotagging. In *MMSys*, 2014.
15. Eric Muller-Budack, Kader Pustu-Iren, and Ralph Ewerth. Geolocation estimation of photos using a hierarchical model and scene classification. In *ECCV*, 2018.
16. Paul Hongsuck Seo, Tobias Weyand, Jack Sim, and Bohyung Han. Cplanet: Enhancing image geolocalization by combinatorial partitioning of maps. *arXiv preprint arXiv:1808.02130*, 2018.
17. Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
18. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
19. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.
20. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.
21. Nam Vo, Nathan Jacobs, and James Hays. Revisiting im2gps in the deep learning era. In *ICCV*, pages 2640–2649. IEEE, 2017.
22. Tobias Weyand, Ilya Kostrikov, and James Philbin. Planet-photo geolocation with convolutional neural networks. In *ECCV*, pages 37–55. Springer, 2016.
23. Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.