Image-Caption Geolocation

Ramya Bhaskara, Nicholas Mohammad, Rajiv Sarvepalli
Department of Computer Science, University of Virginia, Charlottesville, VA 22904
[rsb4zm, nm9ur, rs7uxf]@virginia.edu

Abstract

Image geolocation, classifying the location of an input image, is a difficult problem in computer vision with many applications. In recent years, large datasets of geotagged images have become readily available for researchers to use, and interest in the area has increased. Current state-of-the-art models like img2gps use deep image classification approaches in which the world is split into a quadtree and the model predicts which cell an input image resides in. Unlike these approaches which focus solely on vision, we propose to include not only visual data in our model, but also textual. To elaborate, our model will estimate geographic location with a multi-modal model, which leverages both an image classifier and text-based geolocation parser. Our results indicate that differentiation between geographically similar locations is improved by the use of hierarchical models, and that while a text parser can disambiguate explicit locations from text near perfectly, it is more challenging to disambiguate colloquial, misspelled, and more specific locations.

1. Introduction

In recent years, "anonymous" social media sources have become increasingly popular, with the rise of sites like Reddit, where true identities are typically masked with usernames. With this comes the need to scan the content you post, making sure it doesn't reveal anything about your identity. Our goal with this project is to create a tool that allows users to scan their desired images and text pairings to see whether it reveals too much personal information about themselves (represented by a privacy score). The applications of such a tool can be expanded to all forms of social media, allowing users to control the amount of information they are sharing with the world.

2. Related Work

Early work on visual-based geotagging from photographs include [1, 5] where scene matching and deep neural net image classification are used to identify the location of a single image. All of these previous work concentrate solely on an image to identify its location, and don't rely on any potential captioned textual data that comes along with it. Furthermore, these works focus solely on trying to locate where an image is in the world, whereas our work aims to identify how vulnerable an image-text pair is to geolocation. We show in this work that a multi-modal model can be created to score an image and corresponding caption based on how likely it is that one of these previous works could find the location with high accuracy. To the best of our knowledge, our work is one of the first to generate scores for text-image pairs based on their vulnerability to be localized to a specific gps location.

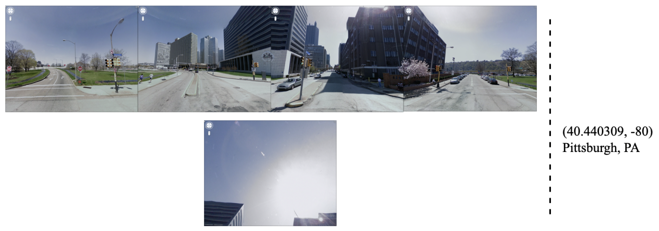

Figure 1. Five Google Street View images associated with a single location.

3. Dataset

In order to limit the problem space due time and compute constraints, we choose to use data from small subset of cities for image geolocation. We utilized a dataset comprised of 62,058 Google Street View images that encompass 3 primary cities: Pittsburgh, Orlando, and Manhattan [7]. These images are paired with their respective geographical coordinates (latitude and longitude). For each coordinate, there are 4 images to cover the 360 degree view from that location, 1 additional image to show the worm's eye view (Figure 1).

Figure 2 shows the label spread of the images across these three major regions. Since the images are from Google Street View, it is similar to images of locations that might be posted on social media. Other datasets could lend

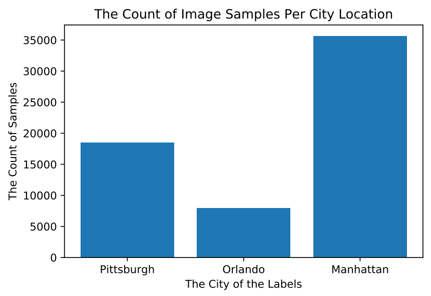

Figure 2. The count of image samples from the three major cities for which the dataset collected data. The label distribution is not uniform, and somewhat unbalanced which introduces some challenges for training the model.

better to the task at hand, since they encompass much larger regions. However, due to time and compute constraints, this dataset was chosen for its smaller size. Additionally, the main focus of interest is not actually finding locations, but understanding what features allow for identification of a location. Therefore, encompassing the entire world is less necessary but would certainly provide additional pertinent information.

For the text portion, the model used parses text and disambiguates locations via database lookup, specifically using the OpenStreetMap database, which contains location nodes that are identified by relationship (i.e. Scotland is in the UK) [4]. The parsing tool we employed uses relationships to group these locations hierarchically for quick lookups (i.e. $UK \longrightarrow Scotland \longrightarrow Edinburgh$).

4. Methods

4.1. Image Model

Several different ways are anyalzed for splitting an image dataset with geographical coordinates for each image into groupings for classification. There are two main ways to analyzed:

1. Splitting based on the different cities in the dataset (3 major cities)
2. Splitting based on hierarchical clustering

The first method is an obvious way of splitting data, but in order to give our model a greater understanding of what composes locality information in an image, the data can be clustered in a hierarchy. The first grouping will be cities and a secondary grouping can be splits inside a city (referred to as districts). In Figure 4, we demonstrate how our KMeans model was able to cluster the city into clear smaller districts. Our goal is to accurately classify districts rather than just cities, because it makes the problem more challenging, and will force our models to better understand what features help distinguish locations that are geographically similar (thus yielding a better proof of concept).

The privacy score for images will be the complement of the district prediction confidence (because the more uncertain your model, the less information is divulged). Additionally, the privacy score will be considered 1 if the city score is too low since if the model cannot even determine the city confidently, the location is relatively unknown. Essentially the image privacy score is,

$$\text{Privacy Score}_{\text{image}} = \begin{cases} 1 & \text{if city conf.} < c \\ 1 - \text{district conf.} & \text{if city conf.} \geq c \end{cases}$$

where c is cutoff value chosen for city confidence and conf. means confidence score.

We use three different basic architectures to provide a point of comparison and to better illustrate the nature of the task. First, we treat the problem as multi-class problem of both cities and district. In this case, the model simply has sigmoid layer at the end and has 33 outputs (3 cities + 3*10 districts). This model outputs into a one-hot vector, but this problem formulation does not make much sense in our context. However, we include for the sake of comparison and as a first pass attempt at a solution. Second, we simply have our model output two distinct outputs: a city and district classification. This is simply trained using a summed loss of each outputs compared to the respective target (city or district). Finally, we use feed-backward model where the outputs are feed back into inputs for a child class. This model is defined using a class hierarchy and created through simple-hierarchy-pytorch, a python library. All the models use a backend of a pretrained ImageNet-trained ResNeXt-101 model with cardinality 32 and a bottleneck width of 8 [6]. The model architecture of the hierarchical model model is described in Figure 3. The expectation is that this model performs the best since the model gains an understanding of relationship between cities and their districts.

We are primarily interested in its ability to extract locality information (or features) from images. However, measuring the ability of a network to accurately measure the location information in a given image is a harder task, so accuracy does increase our confidence that it is extracting relevant features to classify upon. First, the model was trained on the basic categories of the three major cities included in the dataset. Then using the hierarchical classifications discussed above, the model was trained to classify both the outer city and inner portion within the city. This aspect could benefit from a more complex model that can understand the relationship of the defined hierarchy or a dataset that better defines what relevant image features are for locality.

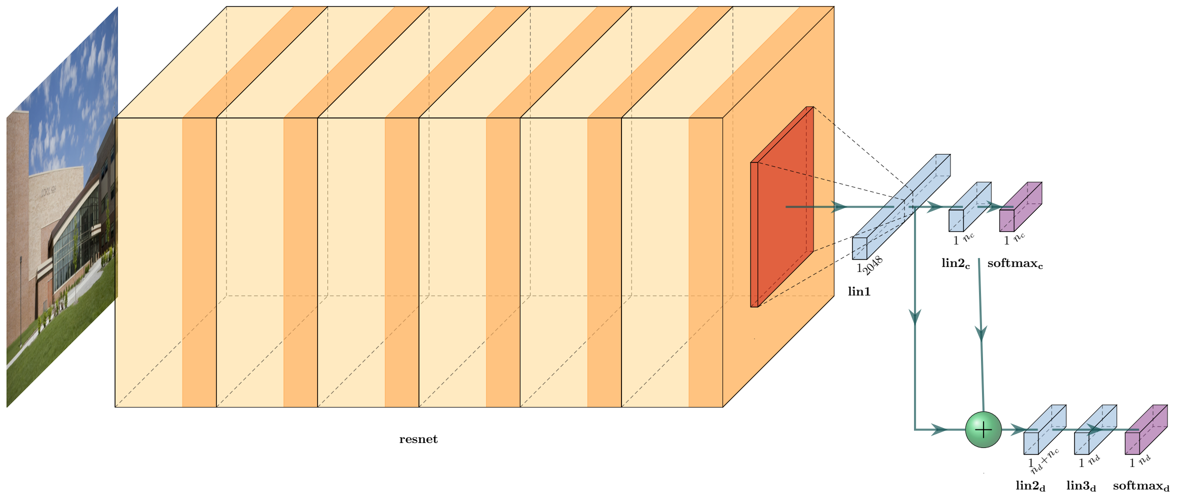

Figure 3. Here we show a figure detailing our hierarchical model components. The component resnet refers to torch implementation of pretrained ImageNet-trained ResNeXt-101 model with cardinality 32 and a bottleneck width of 8 [6]. The n_c refers to the number of cities (3), and n_d is number of districts in each city (10). The plus sign within a ball is indicative of a concatenation of it's inputs. This figure was created using Haris Iqbal's neural network drawing tool. [2]

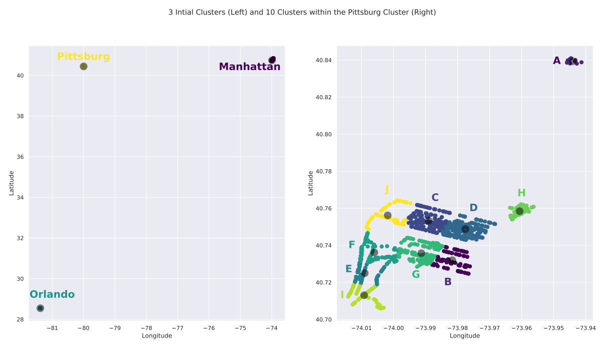

Figure 4. On the left is clustering of the locations into initial categories, and on the right is clustering of one of those initial categories. The shown points are 1,000 data samples in each depiction.

4.2. Text Model

The text pipeline consists of a model that, given text, returns locations that are within that text. The most intuitive way to do this is toponym parsing, which disambiguates locations from text utlizing context from the sentence. The toponym parsing tool, geoparsepy, uses the OpenStreetMap database to disambiguate locations from cleaned, tokenized, and entity matched text [3]. The library executes entity matching by using all potential n-grams, and checking across a corpus of entity names to avoid false positive hits. The derivation of locations is done hierarchically, providing superseding locations along with explicit locations from the text (i.e. Virginia and Charlottesville, VA). The pipeline will provide a score that measures the extent of information about location can be found from the text, calculated by the confidence in the prediction and the prediction specificity.

Each text input generates several candidate locations whose confidences lie between 0 and 300. A location with confidence 0 means it should be thrown out, locations that are likely to be close by to the ground truth are assigned a value of $\geq$ 2. Furthermore, super regions (entities that contain several other regions) in text will allow for confidences $\geq$ 10. If any geolocation predictions intersect (i.e their representative regions overlap), their confidence score is increased by 100.

When the model is run, a confidence vector $\vec{v_c}$ and location vector $\vec{v_{loc}}$ are returned that contain the predicted locations and their confidence. The max confidence $c*$ calculated in equation 1 is used to determine which location from $\vec{v_{loc}}$ to use. This is done to scale the returned confidence score between 0 and 1 by using thresholds.

The final text privacy score is a function of the confidence and the specificity of the prediction (area covered by the predicted location).

$$c^* = \begin{cases} .5 & \text{if } max(\vec{v_c}) = 1 \\ .5 + .25 * \frac{\vec{v_c}}{99} & \text{if } 1 < max(\vec{v_c}) < 100 \\ .75 + .25 * \frac{\vec{v_c}}{300} & \text{if } 100 \leq max(\vec{v_c}) \leq 300 \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Privacy Score}_{\text{text}} = (1 - c^*) \times min(\frac{area}{s}, 1) \qquad (1)$$

where s is an arbitrary constant to represent the area of a small specific location, meant to lower the privacy score if a very specific location is predicted confidently.

The evaluation of this model is less clear cut than the image model, since there is no train or test set of data to evaluate. Thus, our primary form of evaluation will be observational, testing whether it can find explicit lo-

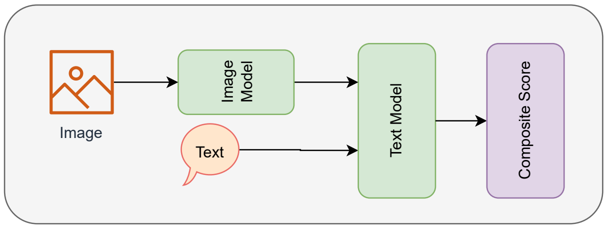

Figure 5. This is a high level architecture of how composite privacy scores are calculated.

cations (clear locations) and implicit locations (colloquialisms, misspellings, etc.).

4.3. Combining Image and Text Model

The goal of the combined model is to account for both text and image when determining location confidence. The results from the two models are combined to provide a single privacy score for an image/caption pair. This score is a measure of how much personal information regarding location is being divulged to the public.

Since the text model already considers overlap of locations when determining confidence, the best way to get a composite score is to concatenate the predicted city to the text input, and input this into the text model. The final score for both the image and text is a function of the confidence and the specificity of the prediction (similar to the equation in 4.2). This pipeline is represented in Figure 5.

5. Experiments and Results

5.1. Image Models

Using the three different model architecture's described in the image model section, we report their performance in Table 1 on the validation image dataset. The performance is recorded as precision, recall, and f1-score (average of recall and precision) for district categorization. The performance is the hierarchical model is the best for districts, which is intuitive since the model gains an understanding of what city the image is in before predicting a districts. However, the gap between two-output is quite small. The gap might widen when the distinct portions of a network are larger (approaching closer to two networks for cities and districts), but due to time and compute constraints we severely limited the expansiveness of the hierarchical network. Using the hierarchical model architecture, Figure 6 shows the confusion matrix for the city predictions. Clearly, the model can very accurately predict the city given an image. Due to our dataset containing a small subset of the world's cities (only 3), this very high accuracy is to be expected. Determining districts is much more of challenges so those are smaller subsets of area and often there are several images that look very similar to humans. Therefore, the reasonably high district accuracies presented in Table 1 increases the likelihood that the model is understanding what components of location are revealing.

In Figure 7, we illustrate a qualitative result from the hi-

Method, Class	Precision	Recall	F1-Score
Multi-Class, City	0.848	0.823	0.834
Two-Outputs, City	0.991	0.982	0.986
Hierarchical, City	**0.995**	**0.991**	**0.992**
Multi-Class, District	0.256	0.220	0.201
Two-Outputs, District	0.712	0.735	0.716
Hierarchical, District	**0.740**	**0.755**	**0.744**

Table 1. Results on the district prediction from training the three different model architecture described in 4.1 for 10 epochs, with learning rate of 0.0001 with 80% of the dataset as training ($\sim$ 50,000 images). For each model arch, the first section includes the city classification performance and second section includes district classification performance. The hierarchical model outperformed the others for all metrics for both city classification and district classification.

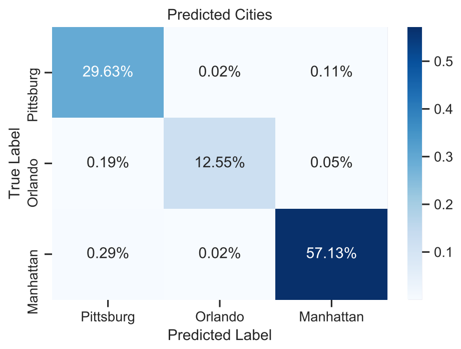

Figure 6. The confusion matrix for city predictions for the hierarchical architecture. The annotations are the percent of the total validation dataset in each categorized section ($\sim$ 12,000 images).

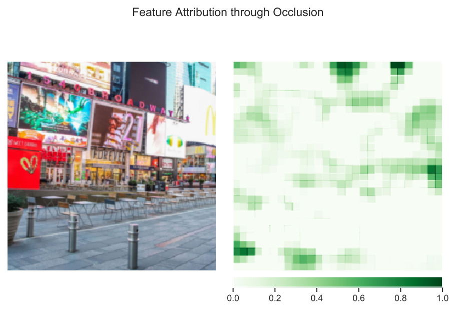

Figure 7. A sample Manhattan photo (on the right) that was not part of our dataset being tested. The feature attribution study is also shown (on the left). The darker the green the more relevant those parts of the image were. The model classified this as Manhattan with 1.000. The image privacy score for this sample was 0.013

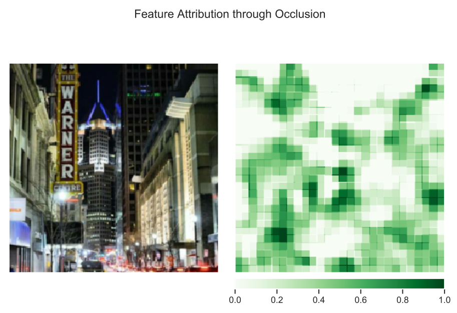

Figure 8. A sample Pittsburg night-time photo (on the right) that was not part of our dataset being tested. The feature attribution study is also shown (on the left). The darker the green the more relevant those parts of the image were. The model classified this as Pittsburg with 0.911 which is relatively low since it is very overconfident model when predicting cities. The image privacy score for this sample was 0.555.

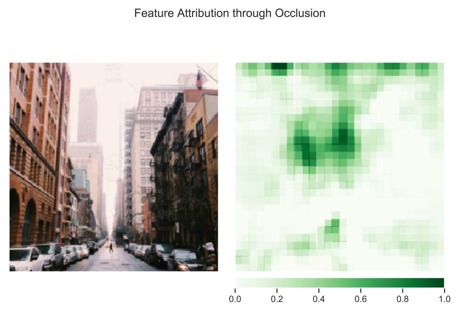

Figure 9. A sample Manhattan photo (on the right) that was not part of our dataset being tested. The feature attribution study is also shown (on the left). The darker the green the more relevant those parts of the image were. The model classified this as Pittsburgh with 0.998 which is overconfidently incorrect. The image privacy score for this sample was 0.322.

erarchical image model along with the feature graph. This feature graph shows which features were most relevant in the prediction of the image to be in that specific location. For images outside the three given cities, the model tends to be overconfident in its guess when the image looks like a portion of the city. This intuitively makes sense, since, in the model's perspective, the only locations that exist are the three cities. Expanding the scope of the dataset (including more locations) would help lower this confidence, or using a dataset geared specifically to this task. Additionally, the dataset biases could play a major role in representing too much on features such as weather and sky color. As seen in Figure 8, the night-time images slightly work, but the lower confidence of the model is symptomatic of the dataset bias. The dataset biases result from many (if not all) of the images are taken during the daytime, and perhaps even around similar days and times. This could lead the image model to inaccurately correlating aspects that are dependent on time with the location such as the sun's position. Finally, Figure 9, our image model's weakness is displayed. Even when it incorrectly classifies a city, the image model is still very confident with its choice. The small sample of cities is most likely the cause of this phenomenon. Therefore, expanding the dataset to many more locations would greatly lower the model's confidence and hopefully increases its ability to understanding what compromises location.

5.2. Text Model

The Text model seemed to perform well on explicit locations and toponyms, but not so well on implicit locations. Below are a few examples (scored 0-100):

1. Going to Disneyland in Orlando: Predicts Orlando FL with privacy score 50
2. Just landed at Pittsburgh International: predicts Pittsburgh, PA with privacy score 50
3. Went shopping in Charlottesville: predicts Charlottesville, VA with privacy score 39
4. Saw the statue of liberty and empire state building: No prediction with privacy score 100

Some additional limitations we observed from the text model are that it is not able to derive colloquial locations from text (i.e. cannot find "Philadelphia" from "Philly"). The text parser also is unable to disambiguate more specific locations, such as street names or store name . This can be seen in sentence 4 above, where it is unable to recognize 'Statue of Liberty' or 'Empire State Building' as locations, yielding a 100% score. It is also unable to disambiguate misspelled locations.

6. Conclusions

Image and text localization together is a difficult task, especially when there aren't appropriate datasets for location prediction that combine the two. Our goal was to create a working methodology to use text and images together to predict a location without such a dataset. We were able to create working proof of concept, geolocalizing images into about 30 districts with a hierarchical model, and disambiguating location names from text via parsing. However, both models had their detriments. The image model was very confident with mispredictions, and the text model was unable to disambiguate implicit location names.

Since we saw lots of deficiencies in the text model, improvements could be made by using parts of speech (i.e. nouns) in a sentence to disambiguate locations. Combined with an expansion of the lookup database to include more specific locations, this would allow the parser to detect more specific locations with less computational intensity.

References

[1] J. Hays and A. Efros. Estimating geographical information from a single image.

[2] H. Iqbal. Plotneuralnet v1.0.0. `https://github.com/HarisIqbal88/PlotNeuralNet`, 2018.

[3] S. E. Middleton, G. Kordopatis-Zilos, S. Papadopoulos, and Y. Kompatsiaris. Location extraction from social media: Geoparsing, location disambiguation, and geotagging. *ACM Trans. Inf. Syst.*, 36(4), June 2018.

[4] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.

[5] N. N. Vo, N. Jacobs, and J. Hays. Revisiting IM2GPS in the deep learning era. *CoRR*, abs/1705.04838, 2017.

[6] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017.

[7] A. Zamir and M. Shah. Image geo-localization based on multiple nearest neighbor feature matching using generalized graphs. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PP(99):1–1, 2014.