CS230

Utilization of Street View and Satellite Imagery Data for Crime Prediction

Jonah Wu School of Humanities and Sciences
Stanford University
jonahwu@stanford.edu

Johannes Hui
Department of Computer Science
Stanford University
jnhui@stanford.edu

Ricky Toh Wen Xian
Department of Civil and Environmental Engineering
Stanford University
rickytoh@stanford.edu

Abstract

Our project aims to consider the issue of crime risk prediction and how neural networks can be utilized to provide additional resources to support crime prevention strategies. Using VGG-16 architectures trained on street view and satellite images and San Francisco crime data, we attempted to predict relative crime risk at locations in San Francisco. Our report builds off previous work by Najjar et. al and others by exploring the additional use of street imagery, adjustments in image-based parameters, the use of a multi-input model, and more. We fail to achieve successful performance, but we developed insights with regards to the task at hand.

https://github.com/jonah-wu/cs230-geospatial-crime

1 Introduction

In many urban areas, police departments, needing to provide effective coverage to their citizens, find difficulties in determining which areas require the most attention from its resources and personnel. In certain cities, such as San Francisco and Chicago, the intersection between years of detailed, publicly-available crime data sets and the need for additional insights to support a limited police task force creates an opportunity for deep learning and data analytics in a field now known as predictive policing. This field has shown promise: in Richmond, the police department's use of data analytics allowed them to see a 47 % decrease in shootings when first implemented on New Year's Eve 2003 [11]. However, significant concerns have been raised regarding the fairness of these algorithms, particularly when used to characterize which areas are safe and due to their being built on data representing historical discriminatory policing [12][13]. In recent years, researchers have been exploring the use of geospatial data; given the nearly universal extent of geospatial data and its easy accessibility, we see its largest implementation potential in areas that are less data-rich and that therefore cannot take advantage of traditional data analytics on compiled data. Another motivation for exploring satellite imagery is to find different tools to support police departments that has the potential to decrease unwanted biases which have plagued predictive policing strategies [12][13]. Our team explored different implementations of a VGG-16 model basing our approach off of Najjar, et. al. For this project, we utilized different variations of a VGG-16 architecture to train a network to perform crime risk prediction.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

Previous studies in both crime prediction and poverty mapping have provided insights on the potential of Convolutional Networks on spatial data in international scenarios. Najjar et. al utilized CNN transfer learning on satellite imagery using ImageNet and Place205 weights, implementing predictions based on cumulative crime counts to bin geographic areas into "low, neutral, and high" categories of safety. After 60,000 training iterations on a 95%/5% split, they were able to achieve an accuracy of 63.8% to 79.5% [12]. Jean, et. al, in a paper on poverty mapping in data-poor regions, was able to similarly use ImageNet weights and successfully utilized Google Static Maps API to scrape granular satellite data for their project; similar to Najjar, et. al, they used clustering to work around potential noisiness in the data [7].

Additional work in this field conducted by Lin, et. al; emphasized the potential of feature construction as well as the need for larger data sets to stop "information overlap" [9]. Chandrasekar, et. al, unlike Najjar et. al, in their crime prediction of hotspots on crime categorized crimes according to types, instead of pure crime counts, to help get a more granular view that categorization merely according to crime counts would not provide [1]. The success of these projects inspired our team's trust in the applicability of Google API-based satellite and street image data.

3 Dataset and Features

Our data input was based on publicly available crime data provided by the local government of San Francisco on DataSF. The first model utilized a dataset approximately 160,000 crimes recorded since 2018. The data set for the second and third models provided in each log the coordinates, type of crime, and for crimes reported on the SFPD Crime Incident Reporting System between 2003 and the present-day. That data set contains approximately 330,000 crimes, which includes crimes recorded from 2016 onwards, two times the data in the first model. The dataset contains crime type and coordinate address of the crime among other fields [14].

As mentioned above, the images for the model were taken from Google APIs. Images were pulled based on the intersections; intersection coordinates that do not match the particular coordinates for a Google street view image are discarded, but this was not a significant amount. Our model was able to

Figure 1: Sample street view (top) and satellite images (bottom)

utilize all 330,000 crime data points by using the following function to find the closest intersection, which we believed was a good approximate:

$$min(|Longitude_{intersection} - Longitude_{crime}| + |Latitude_{intersection} - Latitude_{crime}|)$$

While this method of data collection was slightly inaccurate, when looking at the specific coordinates of sample data whose closest representative seemed to be significantly off, our method picked out the closest intersection.

4 Challenges

The main challenges for our models were related to data collection and quality of the images used. The lack of good quality images, bad clustering, and lack of data plagued our project. When conducting a qualitative analysis of the images obtained from the Google street view API, it was observed that the images were not consistent, coming from different angles and locations relative to

the surrounding built environment. In addition, it is suspected that the disproportionate stretching of image components contributed to excessive variability. For example, a significant number, of the images included pictures of the road pavement, which we inferred would not be predictive of crime. Utilizing a view with a wider angle for street view images did not show significant improvements.

Moreover, our clustering methods changed significantly over the course of the project. The initial idea to cluster based on crime count, particularly when the range of crime counts per category was equal, failed because the dataset is highly skewed towards lower crimes with only some neighbourhoods having high crime rates. As a result, our team suspected over-fitting and utilized log transforms and oversampling methods as corrective measures, though this may have decreased the model's accuracy. In addition, while there was a sufficient amount of recorded crimes, limitations of intersections to approximately 9,000 meant that there might not have been enough images, particularly given the relatively complex recognition task given to the algorithm, compounded with the above-mentioned lack of image quality.

5 Methods

This project's baseline model is a VGG-16 CNN based on transfer learning from a ImageNet (VGG 16) model. Our team decided to use this model particularly because, with multiple possible parameters that can be analyzed, a robust image-processing machine learning model was required. The loss function used was the Sparse Categorical Cross Entropy loss in Keras. This loss function is defined as the sum of the logarithmic differentials, which deducts from categorizations more than linearly based on distance from real categorization. The loss function, mathematically, is expressed as follows [5][6]:

$$LossFunction = -\sum_{i=1}^{n} t_{o,c} * log(p_{o,c})$$

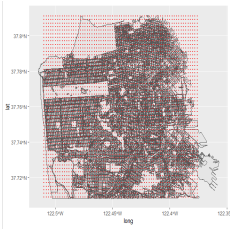
Our baseline strategy in this report was to split, similar to the strategy pursued by Lin et. al, areas in San Francisco into grids [9]. From this and the datasets mentioned in the sections above, the following dictionaries were made: 1. Geoindex-to-Crimes: Each of the grid areas were given i.d.s, or an index, as keys and the cumulative number of crimes committed in the area as its value. This dictionary was created by running through the SF crime database and counting the number of crimes per given intersection and accumulating the crimes from all the intersections for the given grid area. 2. Latitude-Longitude-to-Geoindex: This is a dictionary with each latitude and longitude coordinates acting as a key and its grid region ID as the value.

Instead of making a latitude-longitude-to-crime dictionary directly, these two dictionaries were used to implement binning on the different grid regions using K-means. K was equal to three representing 'Low', 'Medium', and 'High' classes. We had decided on three classes by testing the distortion measure on the data we fed KMeans with. Our first model utilized percentiles but this distributed the regions into a significantly uneven distribution between the classes. This crime count data was paired with another data structure for Google satellite images and Google street view images. As mentioned, this was scraped from the Google Places API. Data augmentation was implemented on this data set: our team trained on two different zooms of satellite data as well as street view based on the model. These images were categorized into the three above-mentioned K-means divisions.

The first model was a standard VGG-16 model with an altered final layer. This model only utilized Google street view data. The final layer used to get the output is a softmax prediction layer. We believed that using ImageNet weights would enable effective transfer learning. Labels for each of the images were generated based on the corresponding lat long's relative crime count percentile. Bottom 40th percentile was placed in the 'low' class. 40th to 70th percentile was placed in the 'medium' class. Above 70th percentile was placed into the 'high' class.

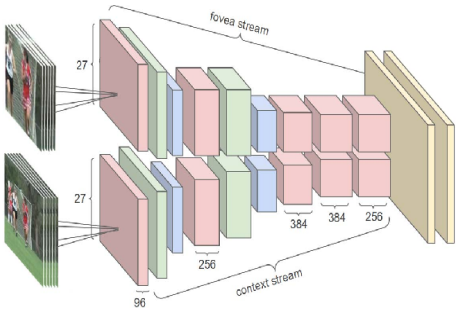
The second model implementation was tested on both satellite and street view images. Moreover, this model used k-means clustering for labelling the images rather than percentiles. Moreover, larger regions (e.g. the grids) were used, and we oversampled from medium and high crime regions. The model also used logits instead of softmax activation for numerical stability, addition of multiple fully-connected layers with RELU activation, and experimented with different optimizers including SGD with momentum.

Figure 2: Generation of larger gridded regions to bin data/images for 2nd and 3rd model



In the third model we used both satellite and street view images in a multi-input architecture built via Keras functional API. Instead of combining the images into one VGG-16 model, Pairs of images were fed through parallel VGG-16 models, the satellite image through one model, and street view through the other. The outputs of each were adjusted such that the two model's output (512 units) was concatenated and combined into a 1028-unit fully connected layer. Three dense fully-connected layers (512-unit, 256-unit, and 128-unit) were used afterwards before utilizing a 3-unit prediction layer. We experimented with dropout layers in an attempt to improve over-fitting to particular mini-batches, which was a concern from the initial model.

Figure 3: Model architecture as seen below but with VGG-16 network parameters taking in street view and satellite images.



6 Experiments/Results/Discussion

For our project, hyper-parameters were mostly empirically tested and verified. Given the size of the available image data (limited by intersections), we wanted to make the most use of the data for training so 90% of the data was allocated to the training set, and 10% to the validation set. The learning rate used for gradient descent, the convolutional filter size (3x3), the size of the max pool (2x2), and the stride (Max pool: 2, convolution: 1) were based on standard VGG-16 implementations. The dropout used in the dual-input model was 0.3 for all the dense layers. To optimize the learning rate, our team decided to adjust mini-batch size as well as implement Adam Optimization. For the first model, a mini-batch size of 256 images was used. For the second and third models, a smaller mini-batch size of 128 was utilized due to the increased number of weights to train.

Other algorithms/deviations for the model that were attempted were as follows:

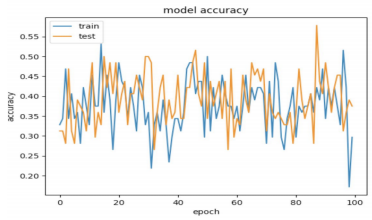
1. Initially, the algorithm was based off of individual intersections, in San Francisco; however, due to signs of data-over-fitting, including a consistently high loss function in spite of higher accuracy values, we decided to attempt to improve performance by combining the intersections into larger areas. Model 2 and Model 3.

2. The inclusion of two different zoom levels for satellite imagery was chosen based on what the team members thought included sufficient objects to analyze. Model 2.

The results for the models were not as successful as hoped for. The first model had an accuracy of approximately 40% with a loss function that had a high variance and that did not decrease significantly.

This was a sign of high bias in the street view model. The second model was an attempt to fix this and resulted in only a slightly higher accuracy at 50%, when used only on satellite imagery. The loss function for this model followed the same pattern of variance similar to the figure below, such that the methods adopted to improve learning were not successful. The last model using the dual input VGG-16 architecture also unfortunately did not result in significant advances, with comparable performance to the second model. Because of the relatively insignificant influence that the model

Figure 4: Sample accuracy training through 100 epochs for 1st VGG-16 model. 2nd and 3rd model variance was comparable



changes had on model performance, we believe that getting more and better quality images, would have been the most helpful in decreasing bias due to the relatively complicated image processing we were expecting the algorithm to do. This is particularly true with street image data because of the large number of visual components that could be analyzed to predict crime, which our team inferred to be more nuanced than previous work on satellite images. In addition, unlike satellite data, these street images are not consistent, often containing distorted objects as well as being in different angles. Therefore, more data was required than our team expected and collected. In terms of image recognition, the accuracy of the model suggests that the VGG-16, as referenced in the above-mentioned sources, is a relatively successful transfer learning scheme. In addition, the lower influence of the region optimization, parallel models etc. does not necessarily mean that, with more data, these implementations would have no effect. Rather, collecting more and higher quality data in addition to these changes would be a good place to get to in continuation with the progress made in our research.

7 Conclusion/Future Work

While this project did show the potential of applying Convolutional Neural Network algorithms towards crime prediction, building off the work of Najjar et. al [12], the results were not successful. Implementation of k-means clustering rather than crime percentiles, larger regional grids, seemed to help. Moreover, at first glance, satellite imagery seemed to be superior to street view imagery for our task. However, these efforts did not significantly decrease the bias of the model, though they did show potential. To build on this work, we would collect significantly more image data to increase the number of intersection pictures processed through the model. It would also be interesting to see the performance on the model when broken down by other characteristics in addition to total crime count including crime type, time of day, season etc. We would also like to build a model which additionally considers features of neighbouring regions, as done by Duan, et al [3].

8 Contributions

Jonah Wu developed the data pipeline between the SF crime data and the images. He implemented the first VGG-16 model, the second VGG-16 model by integrating all the improvements, as well as the multi-input street and satellite imagery model. He did the hyper parameters testing and tested changes to the model architectures. He made the poster and video presentation.

Johannes Hui scraped the Google APIs for the satellite and street images. He developed the idea of using K-means and implemented the algorithm on our data. He assisted with testing hyperparameters.

Ricky Toh Wen Xian helped look up and analyze possible VGG-16 implementations and created a data structure to expand the number of crime counts extracted. Ricky also wrote this final report, and the milestone report. All three team members contributed to the initial literature review done for this project.

References

[1] Chandrasekar, A., Abhilash S.R., Kumar, P. (2015). Crime Prediction and Classification in San Francisco City. Retrieved from https://pdfs.semanticscholar.org/f832/6c812bb1aff4f04d8159059feab984ffd153.pdf?_ga=2.95006684.1781014015.1584178363-1748945776.1584178363.

[2] Dabbura, I. (2018). K-Means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks. Towards Data Science, Medium. Retrieved from https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a.

[3] Duan, L., Hu, T., Cheng, E., Zhu, J., Gao, C. (2017). Deep convolutional neural networks for spatiotemporal crime prediction. In Proceedings of the International Conference on Information and Knowledge Engineering (IKE). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing.

[4] Homicidios 2017. (2018). Datos Abiertos Gov.Co. The Ministry of Information Technologies and Communications, Retrieved from https://www.datos.gov.co/en/Seguridad-y-Defensa/Homicidios-2017/mkw6-468s

[5] How to Use Binary Categorical Crossentropy with Keras. (2019). MachineCurve. Retrieved from https://www.machinecurve.com/index.php/2019/10/22/how-to-use-binary-categorical-crossentropy-with-keras/#categorical-crossentropy-for-multiclass-classification.

[6] How to Use Sparse Categorical Crossentropy in Keras. (2019). MachineCurve. Retrieved from https://www.machinecurve.com/index.php/2019/10/06/how -to-use-sparse-categorical-crossentropy-in-keras/.

[7] Jean, N., et al. (2016). Combining Satellite Imagery and Machine Learning to Predict Poverty." Science. Retrieved from doi:10.1126/science.aaf7894

[8] Knowledge Transfer. (2019). Knowledge Transfer (blog), Androidkt.all. Retrieved from https://androidkt.com/how-to-use-vgg-model-in-tensorflow-keras/.

[9] Lin, Y.L., Yen, M.F., and Yu, L.C. (2018). Grid-Based Crime Prediction Using Geographical Features. International Journal of Geo-Information. Retrieved from https://www.mdpi.com/2220-9964/7/8/298/pdf.

[10] Map of Police Department Incident Reports: 2018 to Present: DataSF: City and County of San Francisco. (n.d.). Retrieved from https://data.sfgov.org/Public-Safety/Map-of-Police-Department-Incident-Reports-2018-to-/jq29-s5wp.

[11] Meijer, A., and Wessels, M. (2019). Predictive Policing: Review of Benefits and Drawbacks. International Journal of Public Administration 42, no. 12, 1031–39. Retrieved from https://doi.org/10.1080/01900692.2019.1575664.

[12] Najjar, A., et al. (2018). Crime Mapping from Satellite Imagery via Deep Learning. ArXiv, Cornell University. https://arxiv.org/pdf/1812.06764.pdf.

[13] Rieland, R. (2018). Artificial Intelligence Is Now Used to Predict Crime. But Is It Biased. Smithsonian Magazine. Retrieved from https://www.smithsonianmag.com/innovation/artificial-intelligence-is-now-used-predict-crime-is-it-biased-180968337/.

[14] SF Crime Heat Map. Data SF. The City of San Francisco. Retrieved from https://data.sfgov.org/Public-Safety/SF-Crime-Heat-Map/q6gg-sa2p.

[15] Street Intersections. DataSF, City and County of San Francisco. (n.d.). Retrieved from https://data.sfgov.org/Geographic-Locations-and-Boundaries/Street-Intersections/ctsg-7znq

[16] UCR Offense Definitions. Uniform Crime Reporting Statistics. U.S. Department of Justice. Retrieved from https://www.ucrdatatool.gov/offenses.cfm.

Data Sources

*Note that these are only references from which code inspiration was taken from explicitly. Documentation is not recorded below. The libraries that were used include scikit-learn, Keras, pandas, numpy, pickle, and more.

A Demo of K-Means Clustering on the Handwritten Digits Data. scikit-learn. scikit-learn. Retrieved from https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html.

Le Bourdais, F. (2015). Frolian's Blog. Retrieved from https://flothesof.github.io/k-means-numpy.html.

Maklin, C. (2018). K Means Clustering Python Example. Towards Data Science, Medium. Retrieved from https://towardsdatascience.com/machine-learning-algorithms-part-9-k-means-example-in-python-f2ad05ed5203.

Python: Find the Closest Key in Dictionary. GeeksforGeeks. Retrieved from https://www.geeksforgeeks.org/python-find-the-closest-key-in-dictionary/.

Python: Find Closest Key in a Dictionary from the given Input Key. Stack Overflow, Stack Exchange Inc. Retrieved from https://stackoverflow.com/questions/7934547/python-find-closest-key-in-a-dictionary-from-the-given-input-key.

Sklearn.cluster.KMeans. scikit-learn. Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html.

Thakur, R. (2019). Step by Step VGG16 Implementation in Keras for Beginners. Towards Data Science, Medium. Retrieved from https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c.