Neighborhood Watch:
Using CNNs to Predict Income Brackets from Google Street View Images

Ambika Acharya
aacharya@stanford.edu

Helen Fang
hfang9@stanford.edu

Shubha Raghvendra
sraghven@stanford.edu

Abstract

The US government spends millions of dollars every year collecting income-level data, a time-intensive and expensive process. Due to recent advancements in computing technology and increased availability of open-source images, our group proposes a deep learning framework for predicting the income level of a neighborhood given a Google Street View image. Using transfer learning on pre-trained VGG16 and ResNet18 models, we experiment with retraining different layers and hyperparameter tuning to determine the model that will yield the highest accuracy. We then perform a post-processing procedure to encode geographic information to our classifications. Our best model attained a validation accuracy of 72.0% *prior to post-processing, which is on par with results in related papers[7, 16, 10]; it consisted of a transfer learned ResNet18 model with layers 3 and 4, as well as the final fully connected layer, retrained. Training was conducted using an Adam optimizer and softmax cross entropy loss. Generating saliency maps, confusion matrices, and other visualizations provide valuable insights into improving the model and understanding its behavior.*

1. Introduction

The US government spends millions of dollars each year attempting to collect income-level data throughout the country. This process is not only expensive, but also time-consuming and often inaccurate. Moreover, as Blumenstock et. al write, the accurate and timely collection of economic data is critical to crafting relevant research and policy [4].

We are interested in predicting the income levels of different communities within a city based on images gathered by Google Street View cars [3]. While this is a regression task, we will simplify the problem to predict whether a neighborhood lies in a specific income bracket, as computed by thresholds derived from income distributions. We also narrowed our problems space by concentrating our efforts on a single city, Oakland, CA, which we chose on the basis the high variance in residents incomes.

The input to our algorithm is a Google Street View image of a location in a neighborhood collected in Oakland. We then use a transfer-learned convolutional neural net (CNN) to predict an income bucket for the neighborhood in question, and lastly perform post processing to improve results using geographic clustering. The importance of this work comes in not only the data collection applications mentioned above, but also in the potentially interesting socioeconomic conclusions we can draw in asking the question: "What observable features predispose a locale to low or high poverty levels?"

2. Related Work

In formulating our approach, we looked at papers that used deep learning architectures to predict a wide range of properties of urban locales based on collections of images. For instance, Dubey et. al trained a Siamese-like convolutional neural architecture on a new crowd-sourced dataset containing image pairs and their comparison across six attributes to predict human assessments of a set of images. Their best performing net has an accuracy of 73.5% [7]. Olligschlaeger used a pre-trained artificial neural network to process data from geographic information systems to forecast the emergence of drug hot-spot areas. The model's best result is a mean absolute error of 110.98%, which can be improved with incorporating more optimized neural net features as well as genetic algorithms, which we looked to incorporate. In the same vein, Blumenstock et. al modeled income levels using phone records in Rwanda [4]. Though these papers either had different training data, different prediction objectives, or both, we found their methods instructive in crafting our own approach.

Research in the realm of parsing satellite imagery to discern socioeconomic trends was relevant to our work as well. Weng and Hu also used an artificial neural network and incorporated linear spectral analysis to map impervious surfaces from satellite imagery for a RSME of 12.3%[15]. Jean et. al extracted large-scale socioeconomic indicators from high-resolution satellite imagery by combining CNN and transfer learning and applying this combination on daytime images. This produces an accuracy of 71.6%[16, 10].

However, the most immediately useful literature we consulted was that authored by those who had also worked with a Street View dataset. Rundle et. al explored the viability of using Google Street View images for auditing neighborhood environments and found high levels of concordance for over half of the items [14]. Gebru et. al use deep learning-based computer vision techniques to estimate demographics from vehicles found in Street View images as a proxy for predicting political affiliation[8] [1].

We also consulted papers which discussed supervised learning techniques to learn more about pre- and post-processing methods, visualizations, and error analysis techniques that could be used to understand and improve the accuracy of our deep learning approach. Ordonez and Berg[13] applied classification and regression models on image features using computer vision techniques to predict wealth, uniqueness, and safety, prediction joint features and found that their results correlates with crime statistics. Naik et. al. use supervised learning on Google Street View images and found that Geometric Texton, Color Histograms, and GIST are the best performing features for the prediction of the perceived safety of a street[12]. Indeed, as discussed below, we found that our saliency maps were activated in regions of dense greenery; in our further research, we'd like to attempt a more explicit use of any of the aforementioned features. Zhou et. al use SVMs to recognize the identity of a city based on attribute analysis of geo-tagged images[18]. They generated visualizations to determine salient attributes in city identification, which will help us in determining the most salient features in our neural net.

Additionally, a paper by Zeiler et. al argues that using visualizations of imagery throughout the training process can aid in improving model performance.[17] We used the ideas for visualizations mentioned here, including confusion matrices, to better interpret our models. Another paper discusses the use of image segmentation to improve performance of CNN's for object oriented tasks.[11] While our task isn't strictly object oriented, being able to put bounding boxes around particular objects in an image could highly correlate with wealth. While we don't address this work in our paper, we assessed the added benefits of incorporating segmentation and found that most images have similar objects, such as car and greenery, which would not be distinctive across classes. Lastly we draw inspiration for methodology in predicting sentiment from pictures using CNN[5], as income level can be described parallel to sentiment as a descriptor across different neighborhoods.

[1]Timnit Gebru advised much of our progress on this project

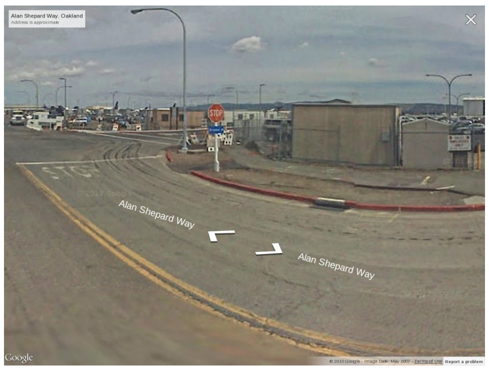


Figure 1: A sample Street View image at latitude 37.714439, longitude -122.214456 and rotation 0.000000.

3. Data

3.1. Collecting Google Streetview Data

The Google Street View data was provided by Timnit Gebru [9] and the Vision group at Stanford. The dataset is a list of url links to images displayed from 6 different angles (0, 60, 120, 180, 240, 300 degrees rotation) for a given latitude-longitude point. We wrote a script to automate the downloading of 40,000 random images of Oakland Street View. These raw images will be the input of our CNN, an example of which is in Figure 3. We elected to begin with Oakland because its income datasets showed significant variance relative to other cities in California, our primary state of interest. Using images of Oakland, we randomly split them into 70% training examples, 20% validation examples, and 10% testing examples. Each image is 224x224x3, where the depth dimension has RGB values.

3.2. Collecting Census Data

In order to label our dataset, we needed to obtain income data corresponding to each lat/long pair specifying an image. To do so, we utilized datasets compiled from the most recent census, obtained from the United States Census Bureau [2]. Specifically, we used data which contain 2015 mean and median household income. Utilizing this data in relation to the Google Street View data required mapping census tract numbers to lat/long coordinates, which we did by automating queries to the FCC's Census Block Conversions API [2]. This produced a set of lat/longs corresponding to the images in the dataset, mapped to average median household income for the year 2015. For the purposes of our baseline approach, we elected to tackle a multi-class classification problem prior to attempting a more difficult regression problem. To do so, we bucketed income levels into three classes, using the following boundary conditions: $\leq 75,000$, $75 - 150,000$, $\geq 150,000$. We selected these

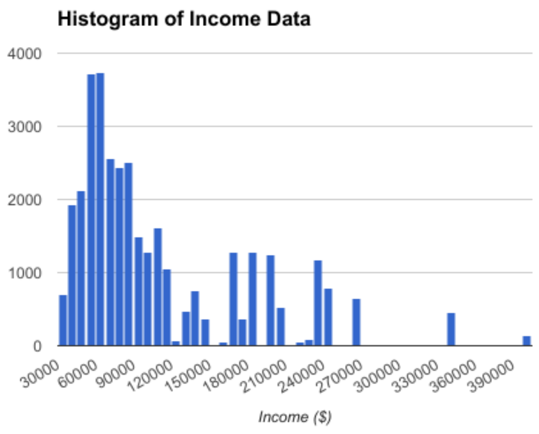


Figure 2: Histogram of 2015 Census income level data for the regions of Oakland, California which make up the 40,000 images in our preliminary training set.

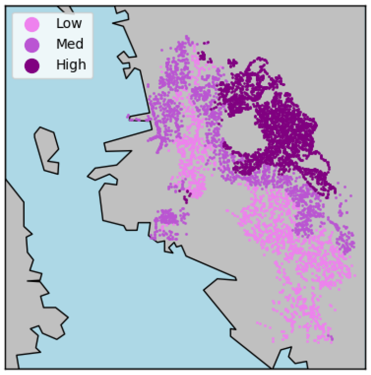


Figure 3: A geographic representation of the Street View imagery, organized by the income buckets we use for classification.

boundaries on the basis of the data distribution of the income level data, shown in the histogram given in Figure 2.

4. Methods

4.1. Baseline Approach: VGG16

For a convolutional baseline, we used a VGG16 model with weights pre-trained on ImageNet images.[6] The model has 16 layers, including convolutional, maxpool and fully connected layers. It uses softmax cross-entropy loss with an Adam optimizer to train the last fully connected layer for 10 epochs. The softmax loss defined for one example i with j classes, where f is a vector for class scores,

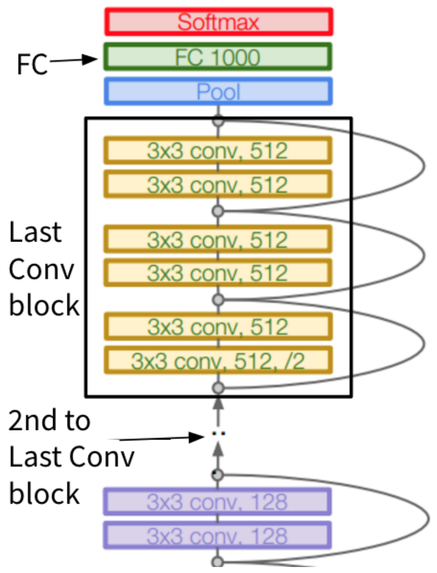


Figure 4: A diagram of the topmost layers of the ResNet18 model. We had 3 separate models fine-tuning FC, the last conv block +FC and the last two conv blocks +FC.

and y_i is the true label is:

$$L_i = f_{y_i} + log \sum_j e^{f_j}$$

We froze weights at all convolutional layers, and mapped the last fully connected layer of the network to three classes, for the three income brackets. Additionally, we made models fine-tuning up until the last convolutional layer and up until the last two convolutional layers to compare performance.

4.2. ResNet

Next, we ran ResNet18 models trained on ImageNet images. ResNet18 has only one fully connected layer at the very end, and uses the concept of stacking small convolutional layers (3x3) as residual blocks to train. We trained the model using the pre-trained default of learning rate 1e-3, Adam optimizer and softmax cross entropy loss for 10 epochs. We made different models, fine-tuning the last fully connected layer to map to 3 classes, then a model adding on the last convolutional block, and finally a model adding on the last two convolutional blocks, as seen in Figure 4. When doing this, we first retrain the layers we wish to fine-tune on our imagery for 10 epochs, then retrain the entire model for another 10 epochs. The residual architecture of the ResNet18 model and its overall performance statistics on other large datasets suggested that it would be a better model choice for this task.

4.3. Hyperparameter Tuning

To improve our best performing model, we tuned the hyperparameters of the model, particularly learning rate and the choice of optimizer. Since the ResNet model was already pre-trained for ImageNet, we decided to only slightly vary learning rate, from 1e-2 to 1e-4, since it was likely to not have a significant affect on model performance. Additionally, we experimented with using Adam, SGD, and SGD+momentum=0.9 optimizers.

4.4. Post Processing

To incorporate geographic information into the classification, we used post-process clustering on the results. After running the model tuned on hyperparameters on the validation set, every image in the set gets a predicted label. We run a modified k-nearest neighbor clustering on the labels: For a given point A we find its k-closest neighbors using the haversine distance for geographic distance, where ϕ_1, ϕ_2 are the latitude, longitude of point A in radians and λ_1, λ_2 are the latitude, longitude of point B in radians:

$$haversine(A, B) = 2arcsin*$$

$$\left(\sqrt{sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + cos(\phi_1)cos(\phi_2)sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$

Then, we take a majority vote on the predicted labels of the k-closest points, and if there is a majority, we assign A the majority label, and otherwise we keep A's predicted label. Because a point is likely to be in the same income bracket as those around it, namely as those in its neighborhood, we use this as a proxy for incorporating neighborhoods into the classification. We run clustering with different values of k to find which gives us the best improvements in accuracy.

4.5. Visualizations

We used saliency maps to visualize what pixels of an image our model was using to classify it [2]. Saliency maps take the absolute value of the gradients of an image and map this to act as an activation map of image features used for classification.

5. Results

5.1. Baseline: VGG16

Using transfer learning on a VGG16 model with weights trained on ImageNet images [3], we removed the last fully connected layer and re-mapped it to our three income classes. We experimented with removing and retraining the last layer (FC8), last 2 layers (FC7, FC8), last 3 layers (FC6, FC7, FC8), and last four layers (Conv5, FC6, FC7, FC8). Each layer was retrained with a learning rate of $1e-3$ and a SGD optimizer with a softmax cross entropy loss function. The best performing model, which removes the last two fully connected layers, reached a validation accuracy of 62.5% The results can be seen in Figure 5.

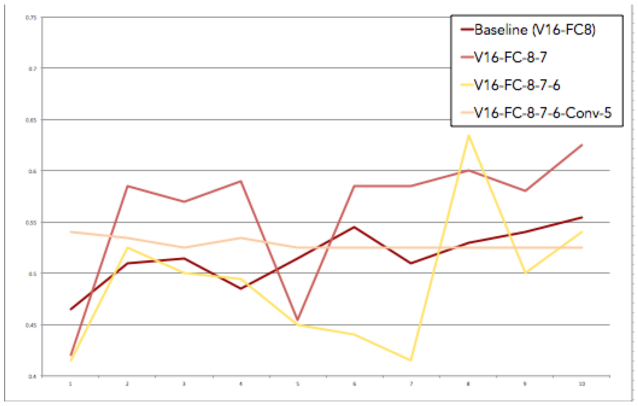


Figure 5: VGG-16 Architectures: Validation accuracy over ten epochs.

While V16-FC-8-7 significantly outperforms a random baseline accuracy of 33.33% (probability of guessing the correct class), we were unable to obtain better results by further changes in the architecture and hyperparameter tuning. VGG16 pre-trained on ImageNet imagery is optimized for image segmentation, and is therefore likely segmenting the images and using segments highly associated with low income neighborhoods to differentiate from those with higher income. We built on these segmentations for our final model.

5.2. ResNet

Using transfer learning on a pretrained ResNet-18 model[4] trained on ImageNet images, we were able to achieve better accuracies through experimenting with different architectural changes.

With our initial parameters, a learning rate of $1e-3$ with an Adam optimizer with a softmax cross entropy loss function, the best performing architecture is RN18 - FC8-Conv4 with a validation accuracy of 71.6%. We chose these parameters to remain consistent with the parameters chosen in the original ResNet-18 model.

On our top performing model, we plotted the geographic locations of the correct and incorrect classifications in Figure13b and generated a confusion matrix seen in Figure 7. From these two figures, we can see that accuracy on medium income level street views is the lowest at 59.7%,

[2]We adapted the PyTorch starter code from Assignment 3 to create saliency maps

[3]VGG16 Tensorflow starter code provided by CA Olivier

[4]ResNet18 Pytorch starter code provided by Justin Johnson: https://gist.github.com/jcjohnson/6e41e8512c17eae5da50aebef3378a4c

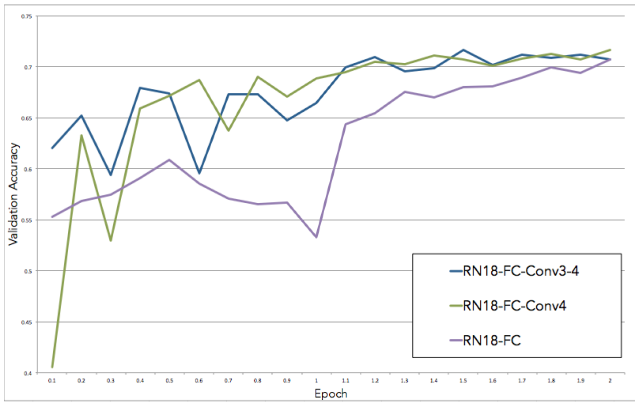


Figure 6: ResNet18 Architectures: Validation accuracy over ten epochs, twice. The first ten epochs (0.1 - 1.0) train just the re-initialized layers at a slightly more aggressive learning rate (for this figure, lr = 1^{e-3}. The next ten epochs (1.1 - 2.0) tune the parameters of the whole model at a more conservative learning rate (1^{e-5}). Please note that any other figures with a scale from 0.1 - 1.0

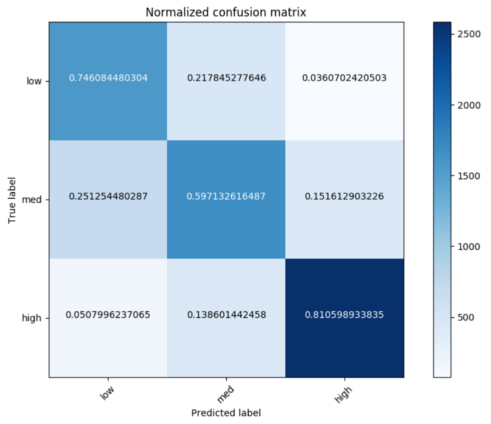


Figure 7: Confusion matrix of our best performing model, RN18-FC-Conv3-4 with learning rate 1e-4, Adam optimizer on the validation set.

with 25% being confused for low income and the remaining 15.2% confused for high income. This could be due to the boundaries for our income brackets. We categorized low, medium, and high income based on the overall distribution of the incomes in Figure 2, but the medium income boundaries of \$75 - \$150k probably overlap significantly with both lower and higher incomes. If we redrew the income boundaries to more accurately reflect socioeconomic divisions, our accuracy will likely improve.

5.3. Hyperparameter Tuning

For hyperparameter tuning, we focused on adjusting the learning rates and the optimizer to determine the highest performing model in terms of accuracy. We found that using a learning rate of $1e-4$ and an Adam optimizer on ResNet18 removing conv block 3, conv block 4, and fully connected layer 8 increased the accuracy to 72.0%, in Figures 8 9. Lowering the learning rate results in ”less” learning over each epoch, but much lower likelihood of overshooting the optimal result. Additionally the adaptive learning rate method Adam is generally the default algorithm used and outperforms methods like SGD and momentum which both manipulate the learning rate globally.

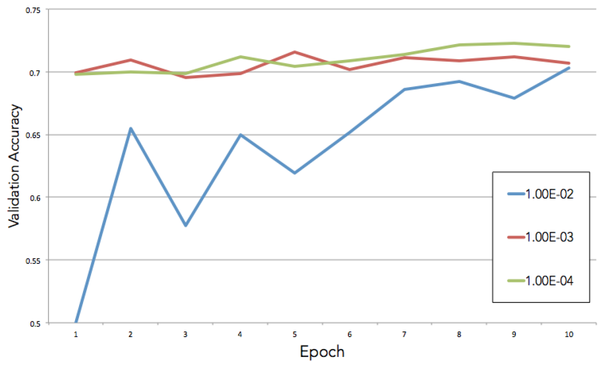


Figure 8: The effect of learning rate on validation accuracy of our best performing model, RN18-FC-Conv3-4.

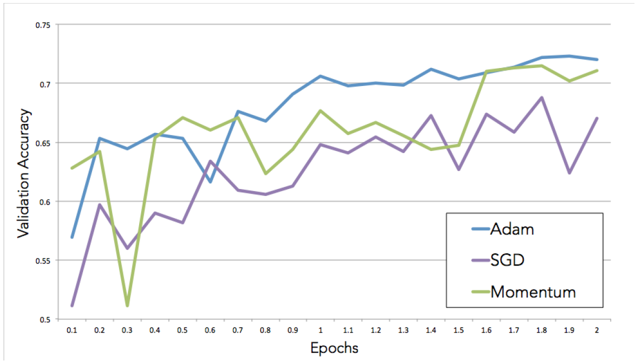


Figure 9: The effect of optimizer choice on validation accuracy of our best performing model, RN18-FC-Conv3-4.

5.4. Post Processing

After performing k-nearest neighbors on the CNN output, we were able to achieve a final accuracy of 85.78% with a k value of 50 in Figure 10. Full results across all models can be found in Table 1.

Generally, the income distribution is relatively uniform in a given neighborhood. As a result, KNN is able to improve our results because through a majority vote procedure, we are able to correct misclassifications on outliers in a neighborhood. However, this method will also misclassify previous correct labels if that majority of the neighborhood was misclassified by the CNN. Visually, sparser regions of misclassification are corrected while dense regions become more misclassified 13c. The latter case is less common, resulting in an overall increase in accuracy.

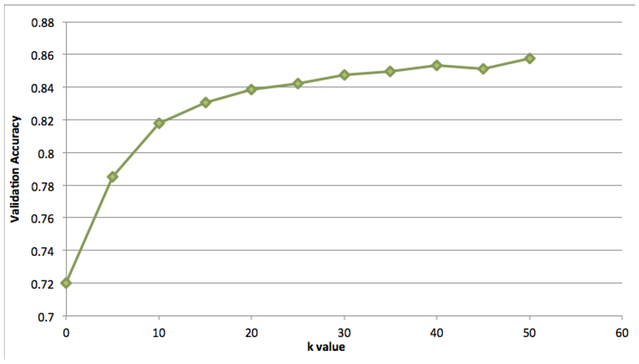


Figure 10: The effect of choice of k for KNN post-clustering on validation accuracy of our best performing model, RN18-FC-Conv3-4.

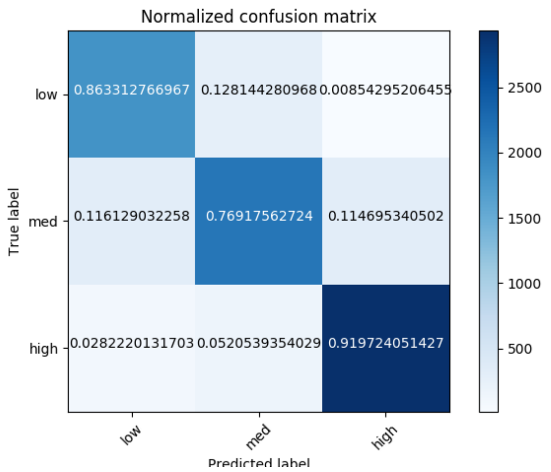


Figure 11: Confusion matrix of our best performing model, RN18-FC-Conv3-4 with learning rate 1e-4, Adam optimizer on the validation set with KNN post clustering with k=40.

5.5. Visualizations

One visualization technique that was instrumental to our data analysis was the creation of saliency maps seen in Figure 13. Saliency maps color pixels in an image proportional to the absolute value of the gradient produced by applying the final layer to an image.

From the low income images presented above the saliency maps, it is evident that many of the locales classified as low-income are industrial areas in Figure 13a. Based on highlighted regions, it is evident that concrete is strongly activated in low-income areas, a trend which we noted in several other images whose ground truth label was low-income. Moreover, we found it interesting that in these and many other images, the vehicles pictured show very low levels of activation. This result potentially suggests that the presence of a vehicle alone is not sufficient to lend meaningful information to the classification of an image; considering that the afore mentioned paper by Gebru et al, which described a classifier specialized for the task of car model recognition, was trained on a much more vast data set. Also notice that in one of the low-income images presented, a patch of graffiti is present; it is significant that this patch is brightly activated in the pictured saliency map. This observation reinforces intuitions we have about the prevalence of graffiti in low-income areas.

The activation of concrete in the low-income saliency maps could be conceptually explained by the fact that often urban areas are dominated by concrete and structures with facades in colors similar to concrete.

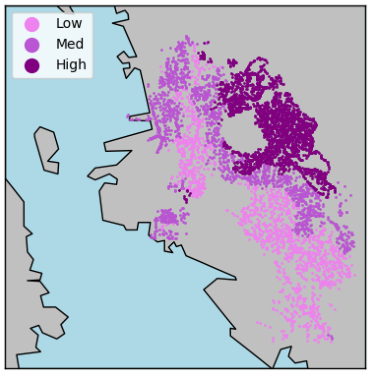
From the high-income saliency maps, it is evident that areas of greenery are activated, suggesting that the presence of foliage is strongly correlated to the presence of high-income neighborhoods from Figure 13c. This finding validated our hunch, based on our own anecdotal experiences, that in cities, greener areas tend to have higher average incomes, and more urban areas have lower average incomes. To us this was a particularly intriguing finding, given that if urban areas lack greenery they also likely lack parks and other public recreation facilities, potentially re-inscribing the income gap in terms of long-term healthcare outcomes.

The medium-income saliency maps were the least conclusive of the set seen in Figure 13b. We believe this may be a result of the fact that the boundaries drawn for the high, medium, and low buckets were drawn somewhat arbitrarily by looking at the histogram of income data. Perhaps splitting into more categories or drawing different boundaries would improve our results; this is worth investigating in our further research. For the purposes of our three-class classification, medium-income saliency maps showed generally high levels of activation; pixels representing both greenery and concrete were activated.

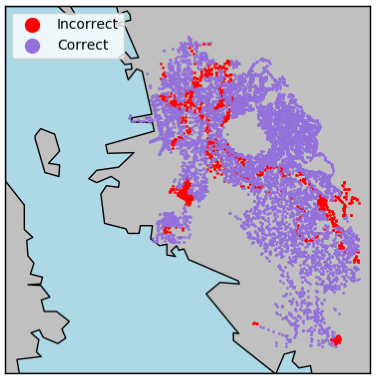
5.6. Results and Error Analysis

We ran our best performing model, RN-18-FC-Conv3-4 with a learning rate of 1e-4 and an Adam optimizer on a held-out test set accounting for 10% of the data, 4,000 images. We then ran post-clustering on these results and report all results in Table 2.

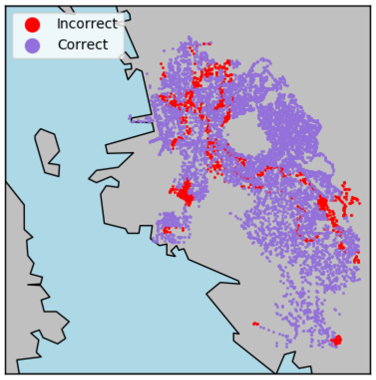
Our validation and test accuracies are very close in number, suggesting that the model did not over-fit to the train nor validation sets and performs well on unseen data. To better understand some of the test examples we didn't learn cor-



(a) True income distribution of images.



(b) RN18-FC-Conv3-4 results.



(c) Post-clustering results.

Figure 12: From left to right we show (a) the geographic distribution of the image data, (b) classification results across all classes, and results after post-process clustering based on geographic location.

Accuracy	**Baseline(V16-FC8)**	**RN18-FC**	**RN18-FC-Conv3-4**	**Post-Clustering**
Train	0.6225	0.7396	0.7237	-
Validation	0.555	0.7069	0.7201	0.8577

Table 1: Table of train and validation accuracy results across all models. All models used a learning rate of 1e-4 and Adam optimizer for 10 epochs on both fine-tuning and the full model. The post-clustering model was run on the results of the RN18-FC-Conv3-4 model and only has a validation accuracy since it was computed post-training.

Accuracy	**RN18-FC-Conv3-4**	**Post-Clustering**
Validation	0.7201	0.8577
Test	0.7045	0.8271

Table 2: Table of validation and test accuracy results across the best performing model, RN-18-FC-Conv3-4 with a learning rate of 1e-4 and an Adam optimizer before and after post-clustering.

rectly, we did error analysis across our three classes. Since 'med' is in between the two classes, the imagery often looks as a overlap of the features detected in the other two classes. Because of this, the majority of our misclassifications came for the med income level. The most interesting misclassifications were between high and low income.

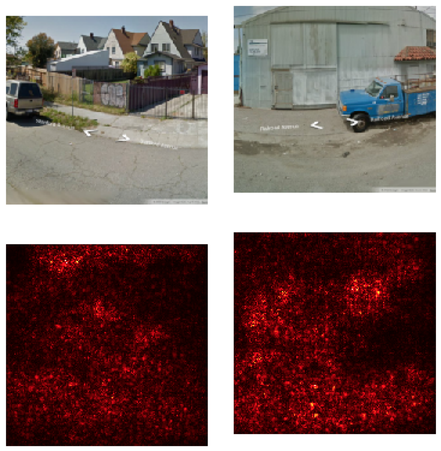
In Figure 14, this image from our test set was classified as high income, when it actually comes from a lower income neighborhood. Because this image contains a lot of greenery, much more than most training images from the low income bracket, we believe the model used this feature as a large predictor during classification. Additionally, the latitude and longitude of this picture put it in a neighborhood which is in between income brackets, meaning re-clustering using geographic location did not change its label.

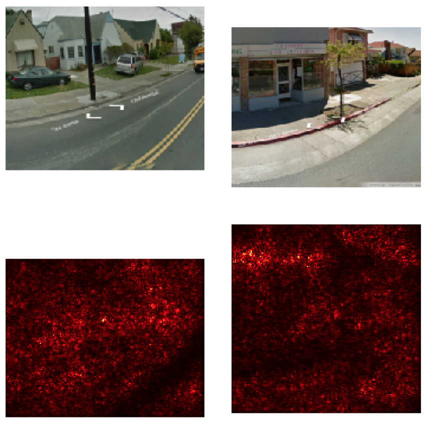
In Figure 15, this image from our test set was classified as low income, when it actually comes from a high income neighborhood. Because there is mainly concrete and a small shop with a partially metallic structure, the model used these features to determine it was low-income. Specifically, the low income class was mainly driven by learning building surfaces and concrete, which was highly activated by this image.

Overall, the main improvement in our model will come when better accounting for the medium income level. While adding post clustering based on geographic location improved med class performance, further improvement could be seen with more medium-specific feature extraction.

6. Conclusion and Future Work

Starting with a VGG16 baseline, we saw a great improvement in performance moving to a ResNet18 architecture. The use of residual blocks and many layers of small convolutional layers are better tuned to the dataset because our dataset was of landscapes, rather than objects meaning it wasn't very similar to ImageNet. Therefore tuning more convolutional block layers was useful to best learn the features of the imagery. RN18-FC-Conv3-4 with a low learning rate of 1e-4 and an Adam optimizer performed the best. Lastly, we saw great improvements (close to 0.15 increase in accuracy) when adding post-clustering based on geographic location. We believe geographic location is essential to this classification problem because areas in one


(a) Saliency maps for low income.


(b) Saliency maps for med income.

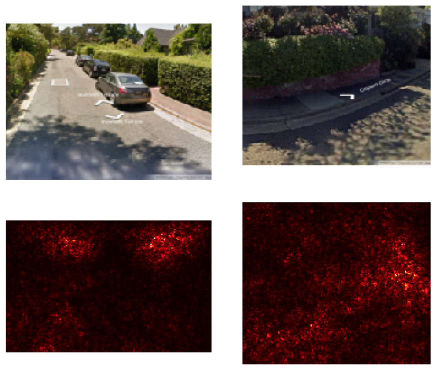

(c) Saliency maps for high income.

Figure 13: From left to right we show saliency maps highlighting which parts of the images RN18-Fc-Conv3-4 used to classify each image for the given class.

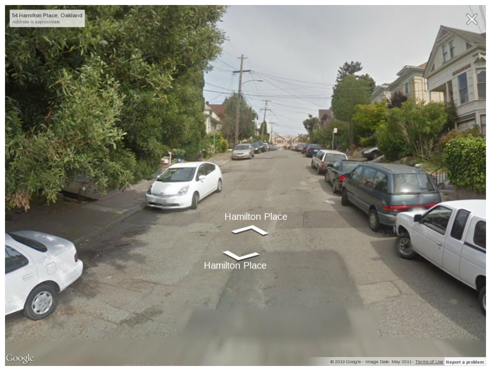


Figure 14: An image at latitude 37.814938, longitude −122.259700 from our test set which was classified as low despite being from a high income neighborhood.

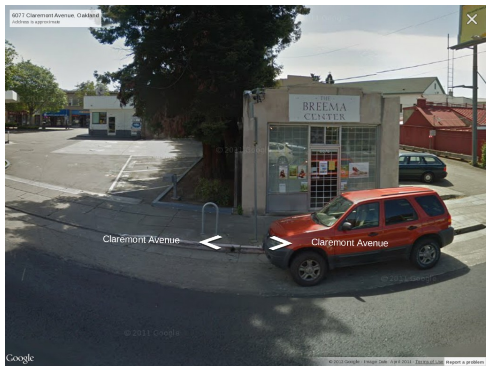


Figure 15: An image at latitude 37.848662, longitude −122.252871 from our test set which was classified as high despite being from a low income neighborhood.

neighborhood are very likely to be in the same income bracket 12, 1. Another significant finding from this work is the types of features the models learned from images relating to each class, as demonstrated in the saliency maps. In particular, we found that vehicle identification was not important(most saliency maps did not activate where there were automobiles), but rather greenery and building surfaces were the most representative of a class. The high income bracket showed a high correlation with greenery, the low income bracket with open concrete spaces and metallic structures, and the med income bracket a combination of the two. Geographically, most of our mistakes were in medium regions, as expected, with high and low income neighborhoods looking very different.

Moving forward, we would focus on using image segmentation to isolate parts of images we believe are highly correlated with income, such as building type and infrastructure. Additionally, we would like to add other cities in California and explore how well our model performs when trained on the data of one city, and tested on others, or if training on data of multiple cities will affect the performance on the Oakland imagery. Overall, the task of classifying income based solely on imagery is quite challenging, and the use of supplementary data such as geographic location and building identification can be extremely useful in aiding this important task. [1]

References

[1] https://github.com/ragggster/neighborhood-watch.

[2] Census block conversions api, Oct 2015.

[3] D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver. Google street view: Capturing the world at street level. *Computer*, 43(6):32–38, 2010.

[4] J. Blumenstock, G. Cadamuro, and R. On. Predicting poverty and wealth from mobile phone metadata. *Science*, 350(6264):1073–1076, 2015.

[5] V. Campos, B. Jou, and X. Giro-i Nieto. From pixels to sentiment: Fine-tuning cnns for visual sentiment prediction. *Image and Vision Computing*, 2017.

[6] F. Chollet. keras. `https://github.com/fchollet/keras`, 2015.

[7] A. Dubey, N. Naik, D. Parikh, R. Raskar, and C. A. Hidalgo. Deep learning the city : Quantifying urban perception at A global scale. *CoRR*, abs/1608.01769, 2016.

[8] T. Gebru, J. Krause, Y. Wang, D. Chen, J. Deng, E. L. Aiden, and L. Fei-Fei. Using deep learning and google street view to estimate the demographic makeup of the us. *arXiv preprint arXiv:1702.06683*, 2017.

[9] T. Gebru, J. Krause, Y. Wang, D. Chen, J. Deng, and L. Fei-Fei. Fine-grained car detection for visual census estimation. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[10] N. Jean, M. Burke, M. Xie, W. M. Davis, D. B. Lobell, and S. Ermon. Combining satellite imagery and machine learning to predict poverty. *Science*, 353(6301):790–794, 2016.

[11] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[12] N. Naik, J. Philipoom, R. Raskar, and C. Hidalgo. Streetscore – predicting the perceived safety of one million streetscapes. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, CVPRW '14, 2014.

[13] V. Ordonez and T. L. Berg. Learning high-level judgments of urban perception. In *European Conference on Computer Vision*, pages 494–510. Springer, 2014.

[14] A. G. Rundle, M. D. Bader, C. A. Richards, K. M. Neckerman, and J. O. Teitler. Using google street view to audit neighborhood environments. *American Journal of Preventive Medicine*, 40(1):94 – 100, 2011.

[15] Q. Weng and X. Hu. Medium spatial resolution satellite imagery for estimating and mapping urban impervious surfaces using lsma and ann. *IEEE Transactions on Geoscience and Remote Sensing*, 46(8):2397–2406, 2008.

[16] M. Xie, N. Jean, M. Burke, D. Lobell, and S. Ermon. Transfer learning from deep features for remote sensing and poverty mapping. *CoRR*, abs/1510.00098, 2015.

[17] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[18] B. Zhou, L. Liu, A. Oliva, and A. Torralba. *Recognizing City Identity via Attribute Analysis of Geo-tagged Images*, pages 519–534. Springer International Publishing, Cham, 2014.