**Comparison of Crowd Counting Strategies for Images with Similar Backgrounds**

**Isaac Pittman, Samuel Bouchard, Joseph Benedict**

University of Massachusetts Lowell

Author Note

Completed as part of the requirements for COMP.4320 (Computer Vision), Spring 2018, taught by Prof. Yu Cao.

&

Completed as part of the requirements for COMP 5450/4220 (Machine Learning), Spring 2018, taught by Prof. Jerome Braun.

**Abstract**

*We propose to count the number of objects in an image by training using a dataset of images with varying numbers of objects place on similar backgrounds. For example, each frame of a video of moving objects from a stationary camera will have many objects on a similar background. We compare the results obtained by training using regularized linear regression using three feature extraction methods: simple linearization of the image's pixels; background removal by subtracting mean pixel values; and, visual bag of words using k-means clustering [2] and SIFT [3].*

**Introduction**

There are many applications which require an accurate count of a given object in an image and they all share the same issue: that it is a menial and time-consuming task which most people hope to avoid doing by hand.

Some such applications include counting pedestrians on a walkway, estimating stadium occupancy, counting vehicles on a highway, counting migratory birds, or counting the appearance of astronomical phenomena. Non-video applications could include counting cells on a microscope slide [1].

We attempt to compare the counting accuracy of various, well known machine learning techniques while varying their parameters, to better understand their effects. Training techniques include linear regression using gradient descent, with and without regularization, in which we vary the learning rate and regularization weight. Feature extraction techniques include simple linearization of the images' pixels using a Hilbert curve, naive background removal by subtracting the mean pixel values, and visual bag of words using k-means clustering and SIFT, in which we vary the number of k-means clusters.

Our dataset for training and testing is the Mall Dataset1 [4][5][6][7], which includes still frames from a public webcam in a mall. The dataset also includes ground truth annotations of the location of each person in the image, represented by 2-dimensional point coordinates. For training, we calculate the total number of people in the image by counting the points.

# Background

In developing our approach for this problem, various other methods were examined. Methods such as those described by Baygin et al [10], where image processing based on Otsu thresholding and Hough transformations tend to yield high quality results and fast computations. However, it is our understanding that these techniques may not be suitably robust, in that they may have trouble detecting and counting objects which overlap each other.

A more state-of-the-art approach presented by Lempitsky and Zisserman [11] takes the form of constructing a density map of an image, such that a sum over the image yields the number of objects contained. This however is somewhat dependent on the original human-made annotations. In their paper, the annotations were loosely positioned on a person's body, which was acceptable due to the robustness of their method. However, it is unclear how a different position (e.g. on the persons head, such as the annotations in the Mall Dataset) would affect the results.

# Approach

## Feature extraction

Pixels as features linearization. To apply the linear regression algorithm to our data set of images, we required each image to be converted to a row vector. First, we converted the image to grayscale, to transform it from a 3-dimensional image matrix to a 2-dimensional matrix of floating point numbers, where each number represented a pixel's intensity. Then, we used a Hilbert curve algorithm to convert the 2-dimensional matrix into a row vector, because the Hilbert curve nicely preserves the locality of pixel clusters during linearization [8]. Although this has no effect on the training accuracy of our current linear regression implementation, since the weighted feature vectors are simply summed, we hope that future works might take advantage of this clustering.

Background removal. To improve training accuracy by removing extraneous details, we implemented a straightforward method of removing the similar backgrounds from images during training. First, we found the mean value for each pixel across all the images, which created a 2-dimensional matrix that represented the "average picture". This was trivial, since the images were all the same size. Then, before linearizing each image, we subtracted this average picture from the image. Because the images contain similar backgrounds, the average picture represented the background. See *Figure 1* for a sample image with the background removed using this technique.

Visual bag of words using SIFT. To linearize the images using visual bag of words with SIFT features, we first extracted 128-dimensional SIFT descriptors for all the images in the dataset. Then, we used k-means clustering to find the 128-dimensional centers of a variable number of clusters, which became our codebook for bag of words. Finally, for each image, we "bagged" the SIFT descriptors of that image into a histogram by calculating which cluster center each descriptor was closest to. We used *VLFeat* to calculate SIFT descriptors, perform k-means clustering, and calculate the distances from each descriptor to the clusters' centers [9].

## Training

We use the linear regression with gradient descent algorithm for all training methods. For comparison, we use L2 regularization. Ten percent of the data were chosen at random as test data for each iteration.

# Results

## Explanation of results

In the dataset, the average number of people per image is 31.1575.

N. The number of iterations in the run.

Average incorrect. The mean and standard error, across *N* runs, of *mean(abs(expected\_count-predicted\_counts))*, where *expected\_count* is a vector containing the actual number of people in each image in the test dataset and *predicted\_counts* is a vector containing the number of people predicted by the algorithm.

Average correct (%). The mean and standard error, across *N* runs, of (*1 - (mean(abs(expected\_count-predicted\_counts)) / mean(expected\_count))) \* 100*.

Iterations. The number of iterations of gradient descent.

Learning rate. The learning rate for gradient descent.

Regularization weight. The coefficient of the regularization term in the cost function used in gradient descent.

Background removed. Indicates whether the background was removed using the background removal approach described above before training.

## Pixels as features

As a baseline, we collected results using the straightforward linearization technique of using each pixel as a feature. The learning rate 1 × 10-6 was chosen via ad-hoc experimentation. A faster learning rate of 1 × 10-5 caused the single-precision floating point numbers representing the weights to underflow, resulting in unusable predictions. (A future implementation could use double-precision floating point numbers, but we were limited to 6 GB of memory for these tests.) A slower learning rate of 1 × 10-7 did not converge quickly enough and achieved only 48% accuracy. See Figure 2 for a graph of the cost rate change of the 1 × 10-7 learning rate.

Results from linearization using pixels as features

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| N | Average incorrect | Average correct (%) | Iterations | Learning rate | Regularization weight | Background removed |
| 5 | 5.49 +/- 0.03 | 82.509 +/- 0.150 | 50 | 1 × 10-6 | 0 | No |
| 5 | 7.32 +/- 0.17 | 76.43 +/- 0.62 | 50 | 1 × 10-6 | 0 | Yes |
| 5 | 14.923 +/- 0.17 | 47.896 +/- 0.331 | 50 | 1 × 10-7 | 0 | Yes |
| 5 | 5.489 +/- 0.072 | 82.270 +/- 0.261 | 50 | 1 × 10-6 | 5 | No |

This straightforward linearization technique was surprisingly effective, achieving 82.5% accuracy. The cost function graph of this run is shown in *Figure 4*. However, the results show that background removal did not improve performance, achieving only 76% accuracy. Changing the regularization weight also did not improve performance significantly.

Further investigation is needed to confirm these results. However, due to the large number of features, computation was very slow for these algorithms. Training took approximately 1.5 hours for each five-iteration run, and preprocessing with background removal took approximately 2 hours, on a 4 core 3.1 GHz laptop with 6 GB of RAM.

## Visual bag of words with SIFT features

Results from linearization using visual bag of words as features

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| N | Average incorrect | Average correct (%) | Iterations | Learning rate | Regularization weight | K-means clusters |
| 20 | 5.336 +/- 0.046 | 83.000 +/- 0.134 | 50 | 1 × 10-6 | 0 | 10 |
| 20 | 5.370 +/- 0.064 | 82.747 +/- 0.189 | 20 | 1 × 10-6 | 5 | 5 |
| 500 | 4.515 +/- 0.011 | 85.504 +/- 0.035 | 1000 | 1 × 10-6 | 0 | 5 |
| 100 | 4.452 +/- 0.012 | 85.704 +/- 0.039 | 10000 | 1 × 10-6 | 0 | 5 |
| 100 | 4.415 +/- 0.014 | 85.822 +/- 0.044 | 1000 | 1 × 10-5 | 0.01 | 100 |
| 100 | 4.295 +/- 0.022 | 86.214 +/- 0.070 | 1000 | 1 × 10-4 | 0.01 | 100 |
| 100 | 4.287 +/- 0.022 | 86.238 +/- 0.073 | 1000 | 1.1 × 10-4 | 0.001 | 100 |

The visual bag of words technique resulted in fewer features, so training speed increased significantly, to less than one minute per 100-iteration run. This allowed much greater flexibility when tuning the parameters and enabled us to run more experiments. We found that *N=100* allowed the runs to complete quickly, while still giving a small standard error.

Starting with 10 k-means clusters resulting in 83% accuracy, which is better than we achieved using pixels as features. Lowering to 5 k-means clusters and increasing the number of gradient descent iterations to 1000 improved accuracy to 85.5%. The cost function graph for this run, shown in *Figure 3,* hinted that even after 1,000 iterations the training algorithm was still improving. This was confirmed when, with 5 k-means clusters and 10,000 gradient descent iterations, accuracy improved to 85.7%. However, the time to run with 10,000 iterations was prohibitive, so subsequent tests were performed with 1,000 iterations each.

To find out whether additional k-means clusters would improve performance, we extracted a feature set containing 100 k-means clusters. Initially, its accuracy was close to 5 k-means clusters (not shown in the results table above). But, increasing the learning rate to 1.1 × 10-4, which was the fastest learning rate possible on the test hardware without underflowing the single-precision floating point weights, and adding regularization weight of 0.001 improved performance further, allowing us to achieve a final accuracy of 86.238%.

# Possible Improvements

Although the SIFT feature extraction was significantly more computationally efficient than the pixels as features implementation, research suggests that the classic David Lowe SIFT algorithm can be further optimized. One paper explored the possibility of replacing the Euclidian distance with a linear combination of Chebyshev and Manhattan distances during distance transforms, since the Chebyshev distance is always less than the Euclidian distance, and the Manhattan distance more than the Euclidian. This resulted in a decrease in computation time proportional to the number of key points being used [12]. Similarly, another paper reported increased computational efficiency when using Fourier Feature Transforms (FFT) and Inverse Fourier Feature Transforms (IFFT) to create Gaussian Scale-space pyramids in the frequency domain [13]. These alterations did not appear to have a sizeable effect on data accuracy and could therefore conceivably improve the implementation performance without jeopardizing the integrity of the resulting data.

# Conclusion

The simplest implementation of linear regression on a set of images, where each pixel represents a feature, achieved surprisingly good accuracy of 82.5%. However, due to the large number of features, it was very computationally expensive. Our background removal technique did not improve the accuracy.

Visual bag of words using k-means clustering and SIFT was much less computationally expensive yet more accurate. Due to the smaller number of features, it allowed a higher learning rate, and due to its speed, we were able to run more iterations of gradient descent. We found that given optimal parameters, we could achieve an accuracy of 86.238%.

# Acknowledgements

Thanks to Ian London for his description of the visual bag of words technique using SIFT, which is available at https://ianlondon.github.io/blog/how-to-sift-opencv/

References

[1] A. Lehmussola, P. Ruusuvuori, J. Selinummi, H. Huttunen, and O. Yli-Harja. **Computational framework for simulating ﬂuorescence microscope images with cell populations.** IEEE Trans. Med. Imaging, 26(7):1010–1016, 2007.

[2] G Csurka, C Dance, L Fan, J Willamowski, C Bray. **Workshop on statistical learning in computer vision**, ECCV 1 (1-22), 1-2

[3] David G. Lowe, **"Distinctive image features from scale-invariant keypoints,"** *International Journal of Computer Vision,* 60, 2 (2004), pp. 91-110.

[4] C. C. Loy, S. Gong, and T. Xiang, **From Semi-Supervised to Transfer Counting of Crowds,** Proceedings of IEEE International Conference on Computer Vision, pp. 2256-2263, 2013 (ICCV)

[5] K. Chen, S. Gong, T. Xiang, and C. C. Loy, **Cumulative Attribute Space for Age and Crowd Density Estimation,** Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 2467-2474, 2013 (CVPR, Oral)

[6] C. C. Loy, K. Chen, S. Gong, T. Xiang, **Crowd Counting and Profiling: Methodology and Evaluation,** S. Ali, K. Nishino, D. Manocha, and M. Shah (Eds.), Modeling, Simulation and Visual Analysis of Crowds, Springer, vol. 11, pp. 347-382, 2013

[7] K. Chen, C. C. Loy, S. Gong, and T. Xiang, **Feature Mining for Localised Crowd Counting,** British Machine Vision Conference, 2012 (BMVC)

[8] B. Moon, H. V. Jagadish, C. Faloutsos and J. H. Saltz, **"Analysis of the clustering properties of the Hilbert space-filling curve,"** *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 1, pp. 124-141, Jan/Feb 2001.

[9] A. Vedaldi and B. Fulkerson, **"VLFeat: An Open and Portable Library of Computer Vision Algorithms",** <http://www.vlfeat.org/>, 2008

[10] M. Baygin, M. Karakose, A. Sarimaden, and E. Akin, **"An Image Processing based Object Counting Approach for Machine Vision Application"**, International Conference on Advances and Innovations in Engineering (ICAIE), 2018.

[11] V. Lempitsky and A. Zisserman, **"Learning to Count Objects in Images"**, NIPS, 2010.

[12] D. Zhu and X. Wang, **"A Method of Improving SIFT Algorithm Matching Efficiency"**, Institute of Electrical and Electronics Engineers (IEEE), 2009.

[13] Y. He, G. Deng, and Y. Wang et. al, **"Optimization of SIFT Algorithm for Fast Feature Image Extraction in Line-Scanning Ophthalmoscope"**, *International Journal for Light and Electron Optics*, vol. 152, pp. 21-28, Jan 2018.

Footnotes

1The Mall Dataset is available for download at <http://personal.ie.cuhk.edu.hk/~ccloy/downloads_mall_dataset.html>

2Team member contributions:

Design: JB, SB, IP

Implementation & results collection: IP

Paper: IP, SB, JB

Presentation: SB

Figures



Figure 1. A comparison of the original image with the image after the background has been removed by subtracting the "average picture", which is made up all the mean pixel values of each pixel across all the images.

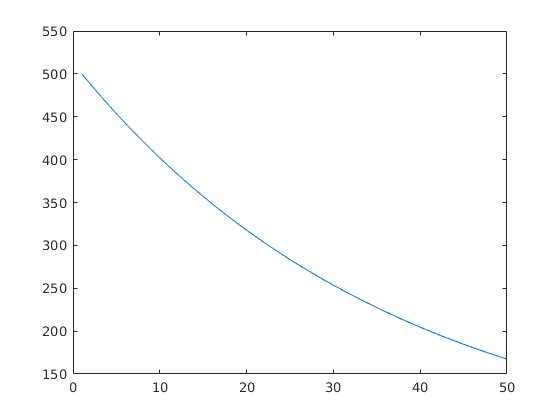


Figure 2. The cost function decrease for iterations of gradient descent using pixels as features with learning rate 1 × 10-7.

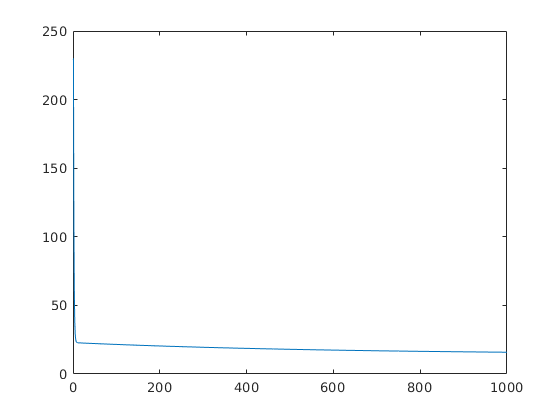


Figure 3. The cost function decrease for iterations of gradient descent using visual bag of words with learning rate 1 × 10-6, 5 k-means clusters, and 1000 iterations.

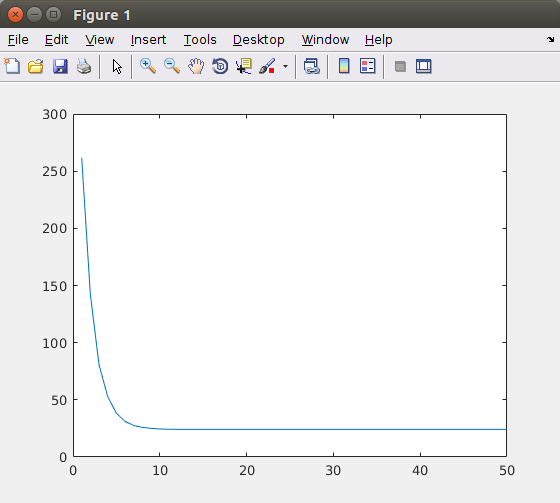


Figure 4. The cost function decrease for iterations of gradient descent using pixels as features with learning rate 1 × 10-6.