Crowd Counting P2P Net

Piyush Bhatia (210720) , Electrical Engineering, IIT Kanpur

Professor: Dr. Tushar Sandhan Course: Intelligent Pattern Recognition (EE798R)

Introduction

This paper introduces a framework for localizing individuals within crowds, emphasizing the practical demands of advanced crowd analysis tasks rather than merely counting individuals.
Traditional methods often prioritize counting accuracy at the image level, which can overlook critical nuances essential for high-level analysis. To bridge this gap, we propose the density Normalized Average Precision (nAP), an innovative performance
evaluation metric that integrates both the accuracy of predicted points and the effectiveness of counting. Central to our framework is the Point-to-Point Network
(P2PNet), which directly predicts point proposals for head coordinates in an image,
along with their confidence scores.
We develop a novel method to directly predict a set of point proposals with the coordinates of heads in an image and their confidences. Our Point-to-Point Network (P2PNet) directly receives a set of annotated head points for training and predicts points during inference. The nAP metric facilitates a detailed evaluation of localization errors and counting performance, ensuring our framework addresses the complexities of crowd analysis.

Methodology

Preprocessing Dataset

The ShangaiTech dataset is divided into two parts: Part A has 182 training images and 300 test images, while Part B contains 316 training images and 400 test images, all with annotated text files and ground truths. To adapt the dataset to the required annotation format for a specific paper, I created a script called preprocess_dataset1.py to remove the last three columns from the text files in the test and training image folders, leaving only the (x, y) coordinates. I also set up the dataset structure as outlined in the GitHub README by creating a DATA ROOT folder and organizing the training and testing datasets accordingly using another script, preprocess_dataset3.py. Finally, I generated the train.list and test.list text files as specified in the README by running preprocess_dataset2.py.

Proposal Matching

The Proposal Matching Algorithm optimizes the association between ground truth points and predicted proposals using a one-to-one matching strategy. It employs a pair-wise cost matrix DDD, which considers both spatial distance and the confidence scores of proposals defined as:

$$D\left(P,\hat{P}\right)=\tau||p_i-\widehat{p_j}\,||^2\ -\widehat{c_j}$$

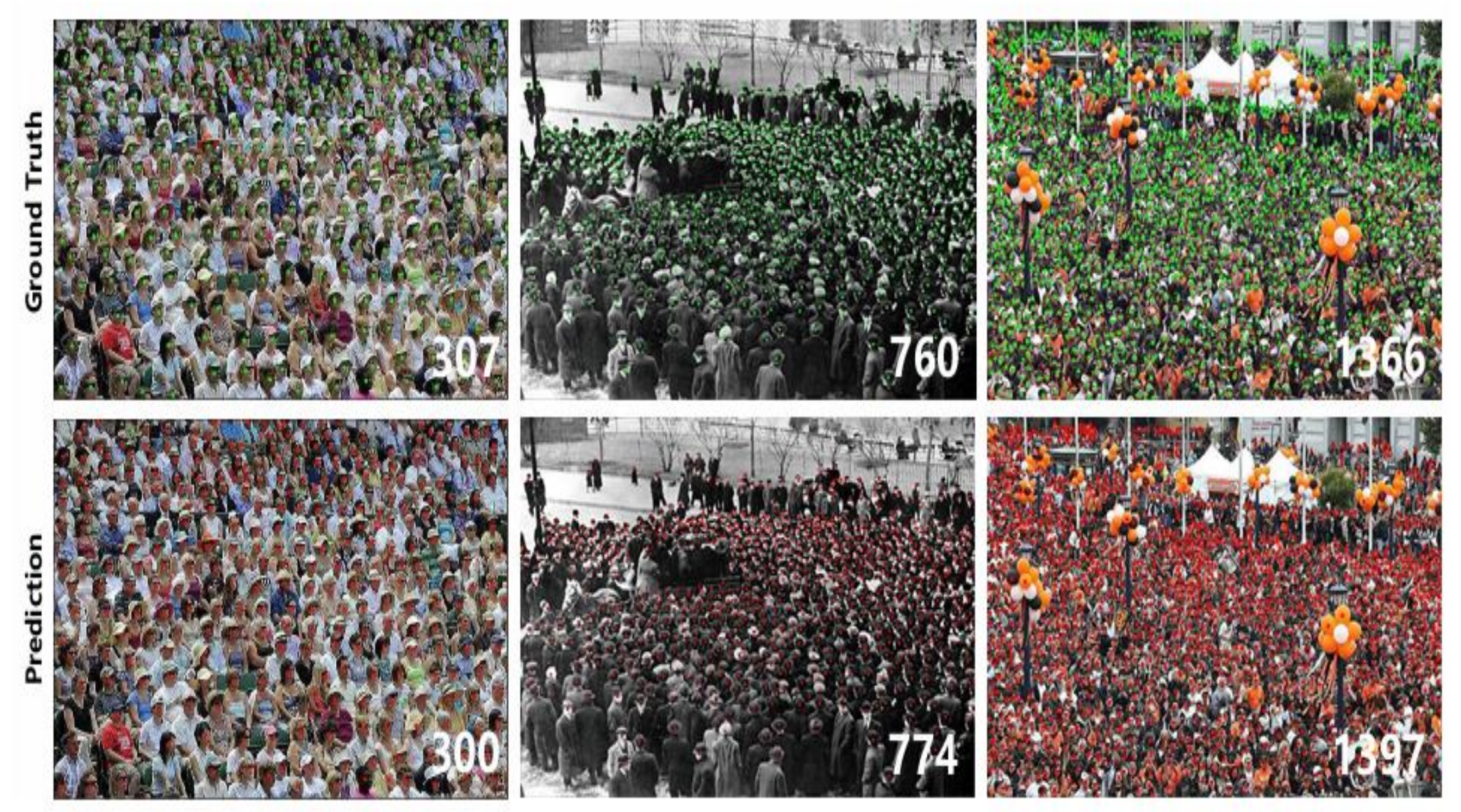
Here τ balances the effects of pixel distance and $\widehat{c_j}$ is the confidence score. By applying the Hungarian algorithm we match ground truth points to proposals ensuring more proposals than ground truths so that excess proposals are labeled as negatives. The matched proposals form a positive set Ppos, while unmatched ones make up the negative set Pneg. This approach enhances predictive accuracy and model performance.

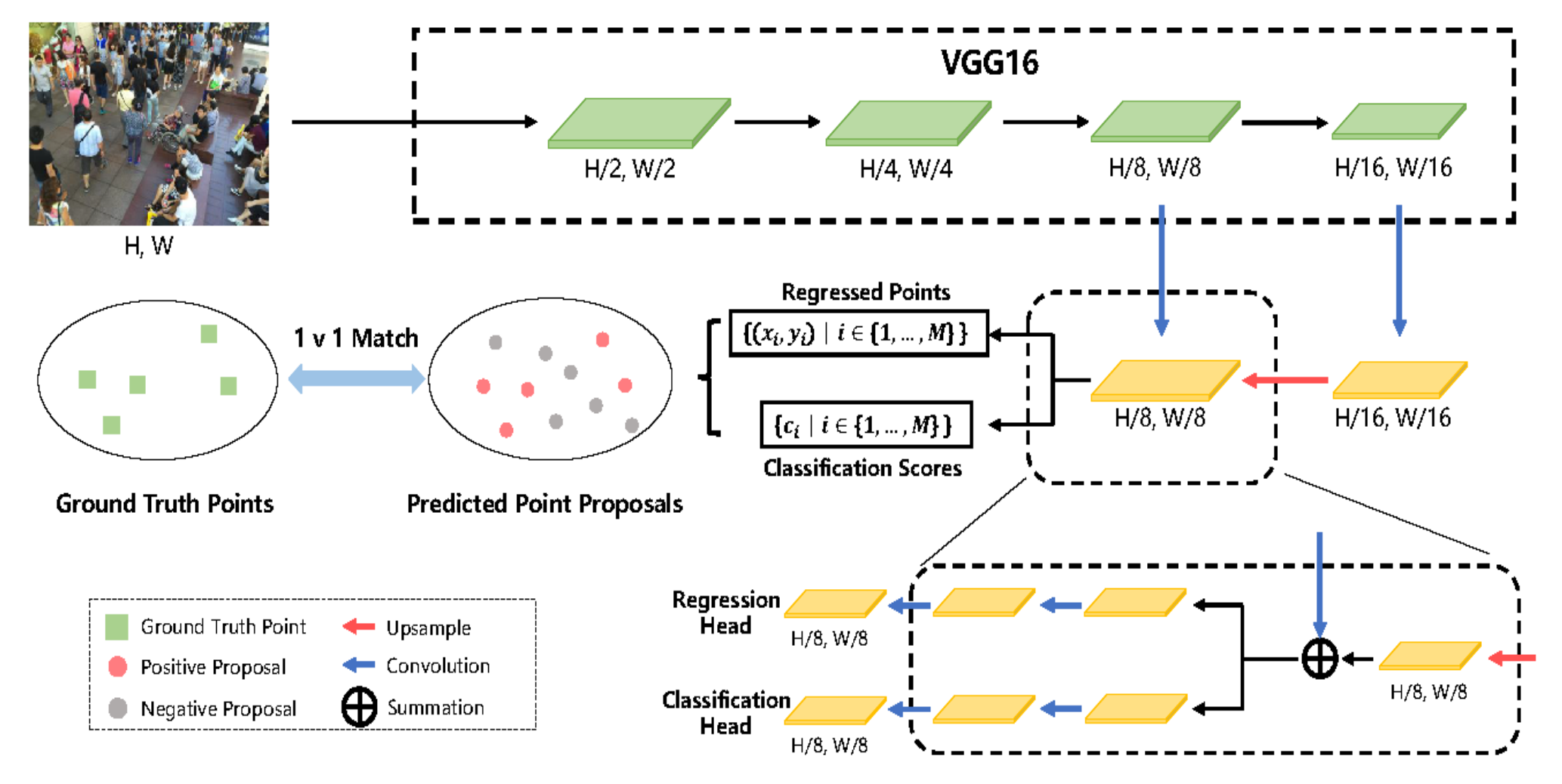
Improvement

We propose an improvement to the model through Adaptive Point Dropout During Training, which enhances results and potentially increases training speed. This technique randomly drops certain ground truth points from annotations based on the local density during each training iteration, particularly in densely populated areas. The original density map D(x,y) is generated using Gaussian kernels centered at each ground truth point, while a dropout mask r_i determines whether each point is retained or dropped introducing controlled sparsity in the modified density map $\widetilde{D}$ (x,y). This adaptation encourages the model to generalize better and reduces overfitting by focusing on overall density distributions rather than memorizing specific point locations. The expected density map reflects a scaled-down version in high-density regions, promoting model robustness. Although fewer points are processed, additional calculations to determine density and dropout probabilities may offset speed gains, resulting in similar or slightly faster training time.

Results

Prediction obtained on Image





Conclusions

- For the Dataset SHTech Part A according to the resources present model is trained for 500 epochs which took nearly 11 hours and got MAE score of 57.27 and MSE score of 91.01 which is significantly close to mentioned score in paper which is obtained by running 3500 epochs.
- For the Dataset SHTech Part B according to the resources present model is trained for 500 epochs which took nearly 12 hours and got MAE score of 8.85 and MSE score of 13.84 which is significantly close to mentioned score in paper which is obtained by running 3500 epochs.
- Therefore, if having sufficient resources and running model for 3500 epochs it can easily approach the mentioned scores.