Using Generated Images in Crowd Counting

Daniel Christopher Biørrith, Simon Fogh Thomsen, and Lukas Koch Vindbjerg

***Abstract*—Crowd counting is a computer-vision task that has made significant progress, yet it still remains a challenging task to create models small enough to be applicable to real-time applications. This is in big part because of the small amount of datasets with accurate annotations, and creating new such datasets is a time-consuming and difficult task. Instead, multiple techniques exist to boost and increase already existing datasets. In this paper, we explore one such technique, called PromptMix, testing the effect it has on a lightweight model. It creates synthetic images using diffusion models, feeding them with prompts generated from existing datasets. In addition to this, we introduce a new way to generate crowd images, based on the SPADE architecture, where segmentation masks are generated from existing datasets, and used as input to generate synthetic images. Images from both models are annotated using state-of-the-art heavyweight models, after which they're mixed with the real dataset to test possible increases in performance. In this paper, we cannot conclude a definite increase in performance based on our experiments, though this gives room for a lot of possible exciting future work.**

I. Introduction

In today's world, computer vision is becoming ever more prominent. In the world of crowd counting, models have seen a large increase in performance in recent years, following a general trend of increasing the size of the models [1]. Even though such heavyweight models perform very well, the computational resources they require to make a single prediction make them infeasible for many cases of deployment, as they will be too slow to work in real-time applications. Lightweight models may be deployed on IoT devices and in smart cities, yet their smaller size means they perform much worse than their state-of-the-art heavyweight counterparts. Due to a range of complexities such as inconsistent perspectives, scale variation, and occlusion, they achieve less accurate results.

Because of these complications, a particular limiting factor is the lack of availability of large, diverse datasets with accurate annotations for training deep learning models. Obtaining more data is a very time-consuming and difficult task. Instead, many people have tried several techniques to boost the quality and size of the available training data. Such techniques include data augmentation, environment simulations, using graphical/game engines, and generating synthetic data [2].

This paper will present one such technique, building on the PromptMix method to be introduced later. First, it will present related work to the subject at hand, whereafter it will present the methods used to boost datasets. The next section will describe the experimental setups for obtaining the results, which will be introduced and discussed in the section thereafter. Finally, we will conclude upon these results, and present possible future work.

D. Biørrith, S. Fogh Thomsen, and L. K. Vindbjerg are with the Department of Electical and Computer Engineering, Aarhus University, Aarhus, Denmark. (e-mail: 201909298@post.au.dk; 201906472@post.au.dk; 201906015@post.au.dk)

Project report for the course 'Computer Vision'.

II. Related Work

To overcome the innate complexities of crowd counting, traditionally, two fundamental approaches have been identified: detection-based and regression-based methods. Furthermore, recent leaps in the field of generative models have given great prominence to crowd generation as a promising contribution to the potential for training more robust and generalized crowd-counting models.

A. Crowd Counting Methodes

The early approaches to crowd counting can be classified into two main categories: detection-based and regression-based. Detection-based methods typically involve identifying individuals based on classifiers in an image and then counting the number of detections. While this approach works well for e.g. detecting faces, its performance declines when applied to larger, more dense crowds. This is in large part due to the complication of occlusion as this hampers the clear distinction of target features.

However, even though this is a more simple and limited approach, the aforementioned issues are handled in more contemporary methods, as proposed in "Rethinking Counting and Localization in Crowds: A Purely Point-Based Framework" by Song et al. [3]. Song introduces the Point to Point Network (P2PNet) which focuses on individual localization. P2PNet uniquely addresses the challenges of traditional detection-based models by using a matching strategy with the Hungarian algorithm. This maps the location of each prediction of a head to the ground truth, which addresses the issue of occlusion.

The methodological solution to the limitations of detection-based approaches appears in the regression-based method as they are able to extract low-level features. They essentially map the input pixels to a single output, being the count. In recent years, literature has used a counting-by-density approach that relies on models trained to estimate the people density on a pixel level so that the total number can be obtained by integration, without any explicit detection being required.

One such approach is the paper of Zhang et al. (2022), who proposed Cross-Domain Attention Network (CDANet) [4]. CDANet is designed to generalize the model to unlabeled domains for both unsupervised realistic-to-realistic and synthetic-to-realistic crowd-counting samples. CDANet works by utilizing Cross-Domain Attention Module (CDAM). This module is designed to observe relations in cross-domain attentive information. This aids in enhancing the crowd-informative features derived from the model which has a significant

influence on the model's adaptability. The proposed model worked by feeding both the target source- and target data through an encoder to obtain a feature representation $\{\mathbf{f}^s, \mathbf{f}^t\}$ These features are then updated through a discriminator to obtain $\{\mathbf{f'}^s, \mathbf{f'}^t\}$. CDAM is then used to compute attention to find correlations between $\mathbf{f}^s$ and $\mathbf{f}^t$, $\mathbf{f'}^s$ and $\mathbf{f'}^t$. A consistency penalty is applied to the attention in order to improve the domain-invariance of the model. Finally, the obtained attentive features are fed to a decoder to generate a density map.

B. Use of synthetic data

To address the data scarcity issue, researchers have turned to data augmentation and synthetic image generation, particularly crowd generation. One such approach is explored in the paper "Locating and counting heads in crowds with a depth prior" by Lian et al. who propose a dual-path guided detection network (DPDNet) with the purpose of handling dense and tiny head detection by the use of RGB-D [5]. To achieve this, they introduce synthetic data from the video game Grand Theft Auto V (GTA-V) as the game engine allows for depth values at any point, providing a large-scale virtual RGB-D dataset. This synthetic dataset consists of 16,911 images and 6,245,510 heads, making it one of the largest RGB-D datasets.

A paper by Bakhtiarnia et al. (2023) introduced PromptMix, a method to create synthetic data to artificially boost the size of training data. This is done by the use of image captioning and, as of this writing, state-of-the-art diffusion models (DM) for image synthesis. This paper will be based on and heavily inspired by the methods of PromptMix, with the intent to further investigate the method and its effects, as well as investigate other image synthesis tools.

III. Methods

The following section will describe the methods we applied in order to create artificial datasets for boosting the performance of lightweight models, as well as how we trained a lightweight model to test their effects. It is based on assumptions that for a given arbitrary computer vision task, at least one dataset and one high-performing heavyweight model, yet not necessarily trained on the said dataset, exist. The full pipeline is depicted in figure 2. As seen, we use two different approaches for generating synthetic images: PromptMix and Gaugan.

A. PromptMix

The PromptMix method generates semantic prompts (descriptive text) based on the images in the dataset, with the possibility of adding manual prompts, and feeds these to a text-to-image generative model. It consists of the following 6 stages: prompt generation, prompt modification, image generation, image filtering, data mixing, and prompt filtering.

Prompt Generation & Modification In the first stage, a prompt p^i it extracted from a given image from the training set I_i, which is done using a DNN f_{cap}:

$$p^i = f_{cap}(I_i), 1 \leq i \leq N \tag{1}$$

Wherein N is the number of training samples. With this, a bag of prompts is created $P = p^1, ..., p^N$. The prompt generation is handled using ClipClap [6], which uses the CLIP encoder that maps text and images to a shared vector space [7] and the GPT-2 tokenizer [8] that transforms the vector to a rich and descriptive text in the style of: "*fans cheer during the match*" or "*crowds gather to watch the parade*". The second stage is to optionally add manual prompts, or modified to the liking of the user. For more info, see the original paper.

Image Generation & Filtering: In the third stage, a fixed number k of images are generated, denoted $I_i^1, ..., I_i^k$, which is done for each prompt $p_i \in P$. This is done using a stable diffusion model f_{dm} with a random seed as input for each image.

$$I_i^g = f_{dm}(p_i, r_i^g), p_i \in P, 1 \leq g \leq k \tag{2}$$

Where r_i^g is a distinct random seed. Diffusion Models are generative models consisting of a forward process, that transforms a signal to noise, and a reverse process trying to denoise the signal, by learning the patterns of the data that it's trained to generate. In the Promptmix method, the semantic prompts are used to guide the model in the latent space, to generate images that match the given prompt [9].

The generated images are then annotated using one or more pre-trained heavyweight DNN, denoted as f_{ann}. Thus, for each image, when using l different annotators $f_{ann}^1, ..., f_{ann}^l$, we obtain l annotations $A_{i,1}^g, ..., A_{i,l}^g$ where:

$$A_{i,j}^g = f_{ann}^j(I_i^g), 1 \leq j \leq l \tag{3}$$

Many different pre-trained heavyweight models exist, and deciding on which ones to use serves as a hyper-parameter. Which one we use will be introduced in the Experiments section. As not all images are of the same quality, the fourth stage is to filter out images of poor quality. This will be done based on the annotations, where a large spread in different annotations $f_{ann}^1, ..., f_{ann}^l$ would imply the image might fall outside the target domain. This is done by calculating a quality score q, if more than one annotation is available, which is assigned to each image I_i^g. It is calculated as

$$q_i^g = \frac{max_{j \neq j'} d(A_{i,j}^g, A_{i,j'}^g)}{max_j d(A_{i,j}^g, 0)} \tag{4}$$

Where d is a distance measure, such as the difference in the total count of two annotations. Based on the calculated quality score, images below a certain threshold r_{img} may be filtered out of the set.

Data Mixing & Prompt Filtering: The fifth stage is to choose a single annotation for each image, in order to be able to train the lightweight image. The best-found annotation would be from the heavyweight model trained on the target dataset. Once chosen, a fake dataset will now be created, ready to train on. However, as the magnitude of the synthetic images may be many times larger than the magnitude of real images, a subset D_{sample}^t of the synthetic images will be chosen randomly at each epoch t. Here, the size of the subset r_{mix} is a hyper-parameter. The very last sixth stage is a final filtering through the model, which will not be explained more in-depth in this report.

B. SPADE

Though PromptMix is a good method for creating synthetic datasets, exploring other ways of creating images is worth doing. Specifically, this means changing stage one, stage two, and half of stage three (the part of it where images are generated) in the PromptMix method. Instead of doing prompt creation and using stable diffusion, we tried generating images using SPADE.

SPADE [10] (commercially known as GauGan) is popular for its ability to generate photorealistic images from simple sketches [11]. In the context of this project, it is trained to generate images of crowds from semantic masks m^i generated from images from the trainset I_i, which is done through transfer learning as the model originally is trained to generate landscapes. The masks are generated using a DNN f_{mask}:

$$m^i = f_{mask}(I_i), 1 \leq i \leq N \tag{5}$$

Wherein N is the number of training samples. Collecting masks for each image gives a bag of masks $M = m^1, ..., m^i$. Just as when working with prompts these masks may be manually inspected and changed. From each mask m^i, k different images I_i^k may be created by the use of the SPADE DNN, denoted f_{spade}, where

$$I_i^g = f_{spade}(m^i, r_{vec}), m_i \in M, 1 \leq g \leq k \tag{6}$$

Where r_{vec} may be a random latent vector. With the image obtained, the stages now follow the PromptMix stage 3 (specifically the annotation), 4, 5, and 6.

1) Semantic Mask Generation: The Semantic Image Segmentation Masks were obtained by combining three different layers of masks. Two of these were obtained with models built on DeepLabV3 [12], a semantic segmentation model, and one with a model built on a MobileNet V3 backbone [13]. The three different masks are:

- **Ground Truth Heatmap**: The Ground Truth Heatmaps are generated from a crowd-counting dataset by applying convolution with a Gaussian kernel. After, simple thresholding is applied to the heatmap to mark the areas of highly-dense crowds (see figure 1b for an example).
- **DeepLabV3 Crowd Segmentation**: This is generated using a pretrained DeepLabV3-model, provided by the PyTorch [14] framework, with a limited number of output classes. It is used to segment the portion of the image, with a crowd (see figure 1c for an example).
- **DeepLabV3 Surrounding Segmentation**: A pretrained DeepLabV3-model [12] on the CityScapes dataset [15]. It is used to segment non-crowd portions of the image (see figure 1d for an example).

Finally, these masks combined in various ways could represent a very low-dimensional latent representation of an image, with a focus on the crowd part of the image, as seen in figure 1.

C. Real Data Sources

With the capability of generating synthetic images, the training of the model can be substantially increased. However, using exclusively synthetic data for this project would not be sufficient due to the inherent challenge of domain discrepancy between the generated and real-world data. Although synthetic image generation is designed to emulate the real world, it will not be able to perfectly capture the intricate and various features found in actual real-world scenes. Hence, real-world data is included to improve performance and is additionally required for validation and testing, given that this is the overall objective of the model.

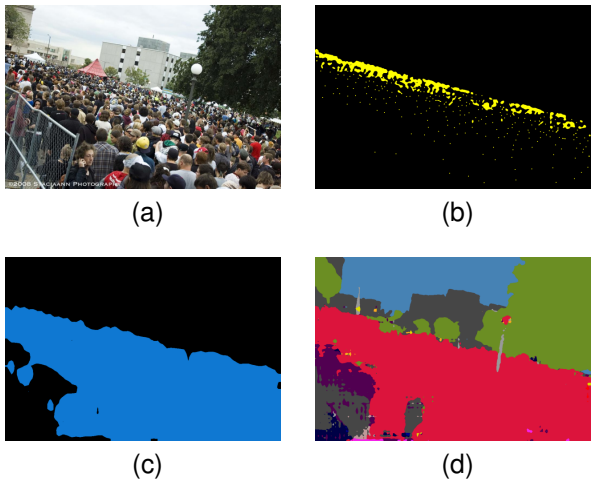

Fig. 1: Different segmentation methods performed. (a) is the real image, (b) is the ground truth heatmap, (c) is the segmentation of the whole crowd, and (d) is the segmentation of surrounding objects.

1) ShanghaiTech Dataset: The ShanghaiTech Dataset is a well-known crowd dataset, divided into two parts, Part A and Part B consisting of 482 and 716 images respectively [16]. For this project ShanghaiTech Part A (SHTA) will be the primary source of real data. SHTA is comprised of pictures collected from the Internet and each image is provided with dot-annotation ground truth, which is transformed into density maps using a Gaussian filter.

Apart from SHTA, other prominent and highly recognized crowd datasets were considered such as ShanghaiTech Part B (SHTB), or UCF [17]. However, images from the SHTB dataset were collected from the streets of Shanghai, resulting in a more similar aesthetic overall. So the greater diversity from SHTA was considered more interesting for the purpose of the project. Furthermore, UCF was a much more challenging dataset with wider ranges in crowd counts, fewer samples of just 50 images, and higher resolution images. It was a concern that this might dilute the discernible trend in the impact of synthetic images, which led to the selection of the SHTA dataset.

D. Annotation of Synthetic Data: Heavy-weight Crowd Counting Model

The annotations A_i are needed to provide a pseudo ground truth for training a lightweight crowd-counting model. As outlined in Section II, a variety of established machine learning

methodologies f_{ann} have been proposed to achieve satisfactory performance when training a crowd-counting model. Nontheless, as density-based approaches are considered state-of-the-art, and given that the chosen datasets use heatmaps, this will likewise be the approach for annotating the synthetic images.

The annotation is done using a pretrained crowd counting model on the SASNet architecture [18] whose output is transformed to coordinates for persons in the crowd. Subsequently, the coordinates are convoluted with a Gaussian Kernel to create a ground truth heatmap.

From this, the method of acquiring the actual crowd count from the heatmap is fairly straightforward. For a value V_{ij} for the pixel (i, j) the crowd count $\hat{C}$ is calculated as the sum of all pixels.

$$\hat{C} = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} V_{ij} \tag{7}$$

When the ground truth heatmaps are generated the values will be distributed so that the sum adds up to the count of people. This will therefore be the same predicted count for the model if the predicted heatmap is close enough.

E. Light-Weight Crowd Counting Model

The target model is inspired by PromptMix's ResCSRNet model. This model was chosen due to its computational efficiency, attributed to it having few dilated convolution layers. Furthermore, shallow ResNets have been shown to possess good feature extraction capabilities [19].

The model is based on a modified ResNet-18 architecture which is pretrained on ImageNet [20]. The model is built up of a head, consisting of all but the four last layers of the ResNet-18 model. This is used for the majority of the feature extraction from the input images. This is then passed to six pairs of dilated convolutions with ReLU activation functions, followed by a final convolution layer.

As a significant emphasis of the project is to observe the impact of the relative performance of lightweight models by including synthetic images, a reduction in computation time is viewed as an added benefit since absolute performance is of less concern.

F. Synthetic Data Utilization

To observe the impact synthetic images have on a model's performance, the implementation will have a systematic approach aimed to observe a trend in the model's behaviour . Therefore, the approach to model training will involve training on synthetic images generated via different methods and with varying real-to-synthetic ratios.

This means that the real dataset will remain the same for every distinct model trained, while the size of the synthetic data will be different. Thus, in order to explore the impact of varying synthetic real-to-synthetic image ratios, models were trained to analyze the effect of increasing the volume of synthetic images. This involved training a baseline model with zero synthetic images and incrementally increasing the number of synthetic images up to ten times the number of real images. The goal is to provide a broad spectrum to evaluate their influence on the capability of the model and try to observe any trends.

G. Evaluation metrics

The evaluation of the performance of our crowd-counting model is primarily dependent on two metrics: the Mean Squared Error (MSE) and the Mean Absolute Error (MAE). These metrics are used for training and validation respectively and are a comparison between the ground truth and the predictions.

During training, the MSE is used as the loss function. It quantifies the average difference between the prediction and the ground truth of the density maps for each pixel. For a single image, the MSE is computed as follows:

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (P_i - A_i)^2 \tag{8}$$

The MSE is calculated for each image sample where P_i is the predicted density at pixel i and A_i is the actual density at pixel i for the total number of pixels N. With this approach, the model will be optimized based on the accuracy of the heatmap, providing much more intricate details during training, than simply relying on the predicted crowd number.

As mentioned, the MAE is used for assessing the model's performance during validation and testing. As opposed to MSE which squares the error of the prediction leading to a disproportionately larger cumulative error – a property relevant for model optimization – the MAE is a more comprehensible and direct measure of the performance. The MAE is used to directly compare the predicted crowd count to the actual crowd count as shown in eq. 9.

$$MAE = \frac{1}{M} \sum_{j=1}^{M} |C_j - \hat{C}_j| \tag{9}$$

Here, M is the total number of samples. C_j is the actual value for the j-th sample and $\hat{C}_j$ is the predicted value for the j-th sample.

By using both these metrics, a comprehensive evaluation of the model's performance can be established by measuring the difference in predicted and actual performance at both the pixel and image levels.

IV. Experiments

A. Synthetic dataset with PromptMix

We use the PromptMix pipeline to generate synthetic images for the datasets SHTA, SHTB and UTC-QNRF. For SHTA and SHTB, 10 images are synthesized per prompt p_i, all with a different seed r_i^g. All were synthesised to be the same size as the image the prompt was generated from.

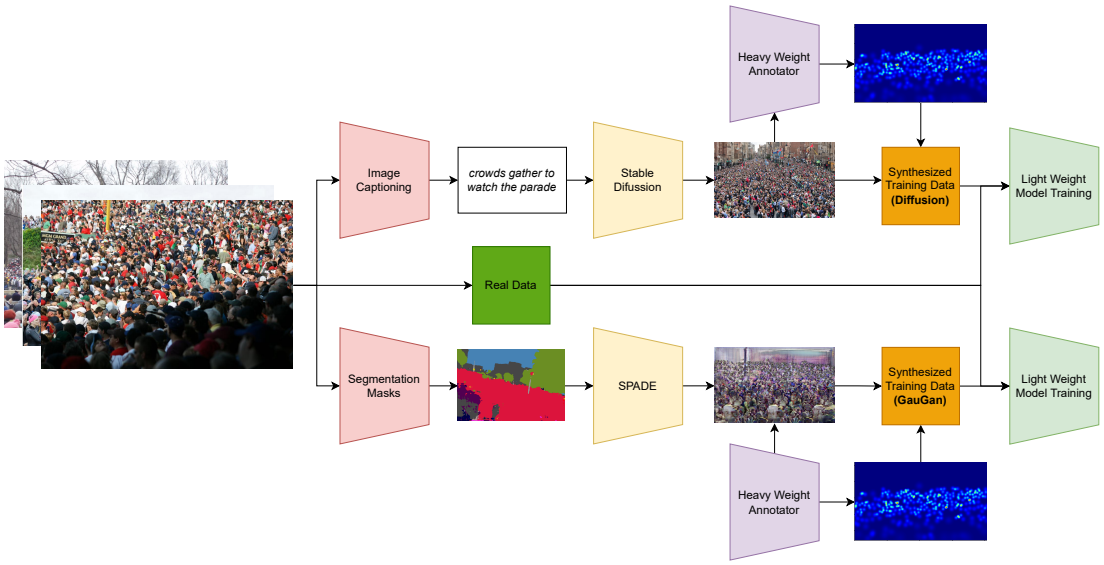

Fig. 2: Overview of the whole pipeline. Two methods of generating images of crowds exist, one using PromptMix and one using GauGan. These are used to synthetically increase datasets to help train better lightweight models.

1) CLIP: Each image from the datasets SHTA, SHTB and UCF has been captioned with the ClipClap caption generator, using a open source pre-trained ViT-B/32-model, which then is tokenized with the pretrained open source GPT2-model to readable text[1]. Each prompt is suffixed with the size of the image the caption was generated from. Additionally, it is noted whether the image is in color or greyscale.

2) InvokeAI: Images are synthesized based on the prompts generated by the ClipClap model using the open-source Stable Diffusion V1.5 model [22] with 50 steps for each image. The amount of images generated for each prompt can be seen in table I[2]. When multiple images are synthesized based on the same prompt, the seed r_i^g is changed.

Dataset	Images/ prompt	Real Images	Synthetic Images	Test Images
SHTA	10	300	3000	182
SHTB	10	400	4000	316
UCF-QNRF	1	1197	1197	334

TABLE I: Overview of the datasets after adding synthesized images.

B. Synthetic dataset with SPADE

Multiple CNN models have been trained with mixed segmentation masks. This will describe the process of generating the masks and how the model is trained.

1) Segmentation Masks: The segmentation masks may be composed of three different layers of masks stacked on top of each other, as described in section III-B1. This includes a 21-class mask segmenting the entire picture (SS), a 1-class mask segmenting crowds (CM), and a 1-class segmentation of dense crowd (HM). A threshold of 0.01 will be used for the dense-crowd overlay. A segmentation mask will be generated for each train- and test image.

2) Training: Three main models will be trained, as presented in table II. They are each trained with varying masks, datasets, and image sizes. In all three models, each image is preprocessed with scaling and cropped to make squared images, the G- and D-loss are tracked, and a snapshot of the model is taken every 10 epochs. With this, the results and models are stored continuously, making backtracking possible.

Name	# Input labels	Dataset	Image size
SHTA_V2	1 (HM)	SHTA	256x256 px
SHTA_V3	23 (SG+CM+HM)	SHTA	256x256px
SSU_V2	23 (SG+CM+HM)	SHTA+STHB+UCT	512x512px

TABLE II: The three different models that will be trained using the GauGan method. They will be trained on varying input labels, datasets, and image sizes.

As the SSU_V2 model is the one trained on the most images of the largest size, and with all masks, we expect this will have the best performance. Thus, this will also be the model used to create the synthetic dataset used to boost the lightweight model. Said dataset will also be compared to the PromptMix dataset by the use of the quality factor q. From this, 830 images will be created to make the synthetic dataset.

3) Heavyweight model: Both datasets are annotated in the same way in order to keep it consistent. Two different heavyweight models, f_{ann}^1 and f_{ann}^2, are used to annotate every image. With this, a quality score q_i^g for each image I_i^g may be calculated, where $g = \{1, 2\}$ is the two different model's annotations. In the calculation, the distance measure is the total difference in crown counting, defined as

$$d(A, B) = |\sum_{i,j} A_{ij} \sum_{i,j} B_{i,j}|. \quad (10)$$

[1]Prompts were generated for SHTA, while the remaining (SHTB, UCF) was supplied from the original paper [21]

[2]Images were synthesized for the SHTA prompts, while the images for (SHTB, UCF) was supplied from the work on [21]

With this, all images beneath a certain filter limit (set in the training state) will be excluded from the dataset. The heavyweight models used to create the annotations were two pre-trained SASNet architectures, where one is trained on SHTA and the other on SHTB. The annotations used for the training were from the ones trained on the target dataset, meaning the one trained on SHTA.

C. ResCSRNet/Model training

To initialize the training of the light-weight model a few modifications had to be made to the original PromptMix implementation.

a) Dimension modification: After passing the images through the model, there were some dimensional inconsistencies between the target- and predicted density maps. Thus, a costume padding layer was incorporated into the model training process to ensure proper fit and compatibility.

b) Data Splitting Strategy and Modifications: Furthermore, to monitor performance during training, both the validation and test datasets were used. The ratio between the training and validation datasets was kept consistent with an $80/20$ split with the real-to-synthetic ratio kept the same for both. The test dataset comprised exclusively of the 182 real images from the SHTA dataset. The reason for this configuration is twofold. Firstly, the validation dataset served as an ongoing metric, offering insight into the model's performance on unseen data that was similar to the training set. Secondly, the test dataset provided the final measure of real-world performance. However, in the original PromptMix implementation, no synthetic images were included in the validation dataset. This setup effectively made the validation dataset equivalent to the test dataset. To rectify this, modifications were made to the dataloader, allowing the inclusion of synthetic images in the validation dataset.

With everything compatible the following configurations of the model were trained on a Nvidia GeForce RTX 3060 GPU. Given the number of models to train, each will be trained for 200 epochs. For the purpose of this project, this will likely be sufficient. Models where the loss decreases quicker often demonstrate better performance once the loss settles [23].

1) Establishing Baseline with Original PromptMix Configurations: To establish a baseline for the testing, an initial training session was conducted from the original PromptMix paper, to recreate the results. This created a reference in which further performance results could be compared and assessed. The baseline model was trained with the following hyperparameters shown in Table III.

2) Models with varying amounts PromptMix generated images: The second test of the project was to train models with different amounts of synthetic data.

The training maintained a constant proportion of real data, split into 240 training and 60 validation samples. Synthetic data were progressively introduced, in the following proportions relative to the real dataset size:

$$[0\%, 50\%, 100\%, 500\%]$$

Hyperparameters	
Parameter	Value
Batch size	1
Learning rate	10^{-3}
Weight decay	10^{-4}
Number of epochs	500
Optimizer	AdamW
Scheduler step size	1
Scheduler gamma	0.99
Loss function	MSE
Limit on fake data	3000
Filter limit	10000

TABLE III: Hyperparameters for Baseline Model Training

3) Model using SPADE-generated images: Finally, a model was trained using a different generated dataset, specifically one produced with SPADE. As there were 830 images generated using this approach, it was again decided to include the scheduler and the original learning rate for training this model.

V. Results

The following section will present the obtained results across the different experiments. It is split up into a section presenting the results of the image generation, and a section presenting the results of training the lightweight models models.

A. Image generation

1) Promptmix: The results of the generated images using the stable diffusion model much resemble those of the original PromptMix paper. An example of the two such images is shown in figure 3. As seen in the figure, though they're not perfect, the images very much resemble crowds of different kinds, and seems to be a good candidate for boosting datasets.

(a) (b)

Fig. 3: Two examples of PromptMix synthetic images.

2) GauGan: Through training of the SPADE model, a clear trend towards overfitting appeared, as may be evident from the samples shown in figure 4. It starts to learn intricate details of the training images, such as the image lighting, building looks, and banner colors.

An example of a synthetic image obtained with GauGan is shown in figure 5, which of course is tested on a mask the model has never been exposed to. The results show that the

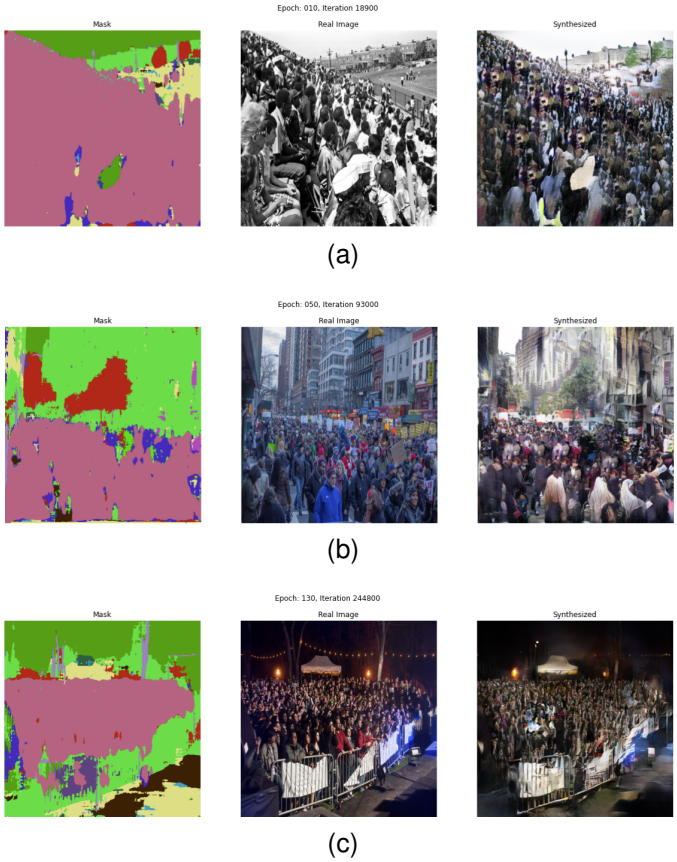

Fig. 4: Samples taken at the 10'th, 50'th, and 130'th epochs of the training of the GauGan model. It shows a clear overfit on the images, with details that would only be copied due to overfitting, such as banner color and lighting.

model creates images that to a certain extent resemble images of crowds.

Not all images are of the same quality, and to the test of the human eye, some of them do not seem up to quality. However, they are good enough for a test with them as a boost to existing datasets will be performed, and the quality score will be used to possible filter out images possibly outside the target domain.

3) Comparing GauGan to PromptMix: As the goal of using GauGan was to find other, perhaps better, methods for generating synthetic images, a direct comparison between the two created synthetic datasets (PromptMix and SSU_V2) was intuitive. For this, we made use of the quality score defined in equation 4. This was done using two different distance functions: MAE and the difference between the two crowd counts. A quality score was found for each image, where two annotators (Two SASNet models, trained on SHTA and SHTB) were used. The distribution of the quality scores are depicted in figure 6a and 6b for MAE and crowd count difference, respectively.

Apparent from the two figures is that the general quality score of the PromptMix dataset is worse than the quality score of the SSU_V2 dataset, meaning the images of GauGan seems to fall more inside the target domain. This is rather surprising, as the images generated by PromptMix generally look better

Fig. 5: Example of a GauGan result. The first image is the real image from which the mask is created, the second is the created mask and the third is the synthetic image. The fourth is the annotation created using a heavyweight DNN model.

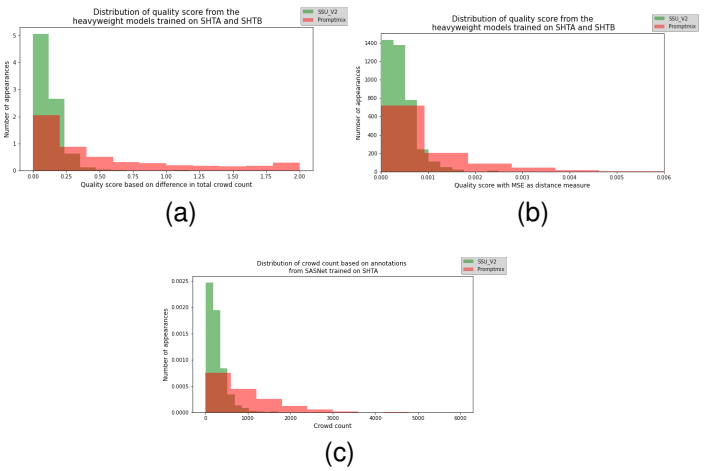

Fig. 6: A direct comparison between the images generated by GauGan (SSU_v2) and the images generated by PromptMix. (a) shows the distribution of quality score with MAE as the distance function, (b) shows the distribution of quality score with the difference between the total count in crowd counting as the distance function. (c) shows the general distribution of the crowd counts across both datasets. This was done with two SASNet models, one trained on SHTA and one on SHTB.

to the human eye than the images generated by GauGan. To possibly investigate the cause, the distribution of crowd counts was plotted in a histogram from the two datasets, which may be seen in figure 6c. From the figure, it's evident that GauGan has a smaller spread than PrompMix, and generally a lower crowd count, which might one of the causes for the better quality score. Though the equation theoretically takes this into consideration by dividing by the max occurrence, this perhaps

is not enough to account for the smaller spread found in the GauGan dataset.

B. Crowd Counting Model

After executing all iterations of the models, with the different datasets, the resulting performance after 200 epochs is displayed in Table IV.

Models	Train MAE	Validation MAE	Test MAE
PromptMix 1000%	23.88	246.06	152.00
0% Synthetic	22.42	246.56	155.47
50% Synthetic	13.56	276.01	129.22
100% Synthetic	19.69	316.35	117.2
500% Synthetic	23.65	366.67	138.53
GauGan	22.16	138.76	129.49

TABLE IV: Results from the models' performances after 200 epochs for different synthetic images and ratios

The first results with PromptMix have been an attempt at recreating the results from the PromptMix paper. The following models represent different percentages of synthetic data and the final result is the model using the SPADE-generated images. From these, the unfortunate conclusion has to be drawn, that the results from PromptMix were not fully met. The cause of this can be attributed to many different factors and pinpointing the precise reason is non-trivial. A plausible reason could be the necessity of including the custom pooling layer during training.

Looking at the performance of the models with varying synthetic data, some surprising observations emerge. The performance is remarkably similar across all models, and the addition of more synthetic data seems to have very little impact. A clear overview of the performance differences can be seen in 7. The most distinguishable impact is the steady increase in validation MAE compared to the more constant Train- and Test MAEs with more synthetic images. This is likely not due to a lack of training, as the models have clearly overfitted after 200 epochs. A more reasonable explanation might be that, despite of ResCSRNet's great performance, its feature extraction capabilities may not be sufficient to demonstrate any discernible trends.

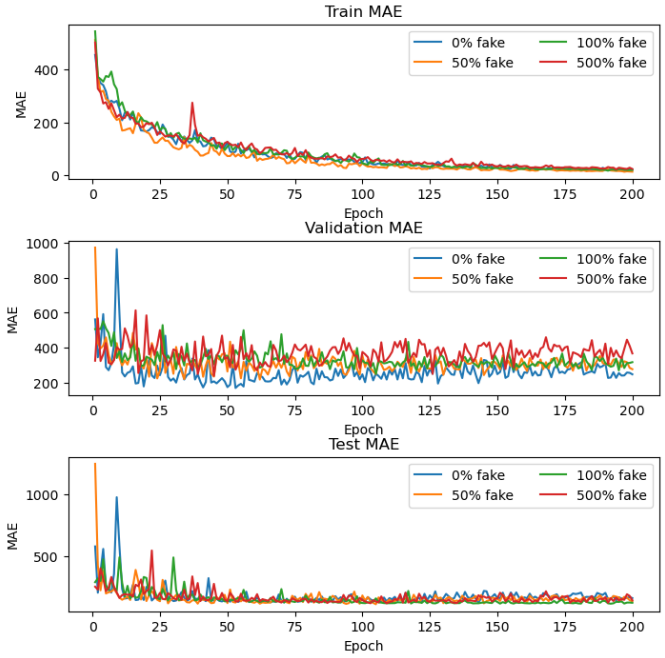

Fig. 7: MAE for the training, validation, and testing for different iterations of the model with varying synthetic data ratio

Another surprising result is that the validation MAE is higher than the Testing MAE, even though the training dataset and validation dataset are more similar compared to the testing dataset. The reason for this is difficult to assert for sure. Still, one possible explanation could be the inconsistent quality of the annotated synthetic data. If the real data annotations are accurate while the synthetic data is more inconsistent, the model might struggle to perform on unseen synthetic data due to its inherent noise. However, it could be capable of handling the real data as there is more consistency from the training to the testing set. This is just speculation and it could be just as plausible that the data is loaded incorrectly.

Lastly, the performance of the model trained on the SPADE synthetic data shows great promise, especially in the validation MAE. However, these results should be evaluated critically since the performance on different datasets is not always comparable. As discussed earlier, the average count of people on the SPADE images tended to be lower than the ones produced by PromptMix. Given that the MAE evaluation metric takes absolute error into account, lower value targets naturally permit less potential error.

VI. Conclusion

This paper explored the effect of PromptMix, a method for artificially boosting the size of training data by creating synthesized images based on the existing 'real' ones using stable diffusion. Furthermore, it introduced a new method for creating synthetic datasets SSU_V. This was done by using transfer learning on GauGan, a model originally intended for creating images of landscape based on an input mask, by training it to generate images of crowds based on existing training data. The two datasets were directly compared, where SSU_V2 provided better results based on quality score, though this is to be taken skeptically for reasons discussed. Even though extensive testing was performed to see the effect of scaling up the number of synthetic images, we could not make a precise conclusion regarding the effects of the PromptMix method. This was the case for both synthetic datasets. The reasons behind the faulty results were discussed, and it is likely more a fault of ours, rather than a fault of the method itself. However, due to the limitation of the project, we did not have the time or resources to discover the reasoning behind these errors.

VII. Future Work

Unsurprisingly, the fist step would be to address the issues with the results of the model performance. This would be

necessary to obtain a fundamental baseline for all further experiments.

In addition to this, an interesting approach would be to attempt a larger variety of lightweight models on the same data to see if there could be any interesting features not picked up by ResCRSNet.

Lastly, for this project, it was attempted to generate and use a dataset generated by the GTA V engine like the solution proposed by Lian et al. [5]. During this project, target heatmaps and depth maps were successfully generated. However, due to the limited time of the project, this data was not processed to a state where it was compatible with the model architecture.

REFERENCES

[1] Papers With Code. Locating and counting heads in crowds with a depth prior, 2023.

[2] Goran Paulin and Marina Ivasic-Kos. Review and analysis of synthetic dataset generation methods and techniques for application in computer vision.

[3] Qingyu Song, Changan Wang, Zhengkai Jiang, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yang Wu. Rethinking Counting and Localization in Crowds: A Purely Point-Based Framework. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3345–3354, Montreal, QC, Canada, October 2021. IEEE.

[4] Anran Zhang, Jun Xu, Xiaoyan Luo, Xianbin Cao, and Xiantong Zhen. Cross-Domain Attention Network for Unsupervised Domain Adaptation Crowd Counting. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(10):6686–6699, October 2022. Conference Name: IEEE Transactions on Circuits and Systems for Video Technology.

[5] Dongze Lian, Xianing Chen, Jing Li, Weixin Luo, and Shenghua Gao. Locating and counting heads in crowds with a depth prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):9056–9072, 2021.

[6] Ron Mokady, Amir Hertz, and Amit H. Bermano. ClipCap: CLIP prefix for image captioning.

[7] sentence-transformers/clip-ViT-b-32 · hugging face.

[8] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and others. Language models are unsupervised multitask learners. 1(8):9.

[9] Xihui Liu, Dong Huk Park, Samaneh Azadi, Gong Zhang, Arman Chopikyan, Yuxiao Hu, Humphrey Shi, Anna Rohrbach, and Trevor Darrell. More control for free! image synthesis with semantic diffusion guidance. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 289–299. IEEE.

[10] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic Image Synthesis with Spatially-Adaptive Normalization, November 2019. arXiv:1903.07291 [cs].

[11] GPT-3 Demo. GauGAN2 by NVIDIA | discover AI use cases.

[12] DeeplabV3 pretrained (CityScapes) repository.

[13] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for MobileNetV3.

[14] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[15] Cityscapes dataset – semantic understanding of urban street scenes.

[16] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 589–597, 2016.

[17] Haroon Idrees, Imran Saleemi, Cody Seibert, and Mubarak Shah. Multi-source multi-scale counting in extremely dense crowd images. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2547–2554, 2013.

[18] Qingyu Song, Changan Wang, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Jian Wu, and Jiayi Ma. To choose or to fuse? scale selection for crowd counting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 2576–2583.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[20] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[21] Arian Bakhtiarnia, Qi Zhang, and Alexandros Iosifidis. PromptMix: Text-to-image diffusion models enhance the performance of lightweight networks, January 2023. arXiv:2301.12914 [cs].

[22] runwayml/stable-diffusion-v1-5 · hugging face.

[23] Leslie N Smith. A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. *arXiv preprint arXiv:1803.09820*, 2018.