Access the code, data, and analysis at github repo

Using computer vision and AI techniques to improve crowd safety management

Project report - January 2024

Anton Irvold
University of Southern Denmark
anirv20@student.sdu.dk

Christoffer Krath
University of Southern Denmark
chkra19@student.sdu.dk

Administrative Info

Collaborators from Event Safety

- Sofie Dahl, sofie@eventsafety.dk
- Christian Sejlund, christian@eventsafety.dk

SDU supervisor

- Sune Lundø Sørensen, slso@mmmi.sdu.dk

Context and Background

As seen from Figure 1 accidents, fatalities and injuries at festivals and concerts worldwide are on the rise, with just below 5,000 worldwide deaths in the decade of 2010-2019. This can be attributed to several causes, according to Raineri (2004): First of all, an increase in the popularity of outdoor music festivals resulting in more and larger festivals with large crowds. Also, high-risk behavior among crowd attendants and the performing artist's music, behavior, and stage show affects the crowd's safety. Cultural influence has also played a large part in the safety of outdoor musical events. This can be behavior such as crowd surfing, moshing, stage diving and the like. Sudden panic in a crowded place can thus affect the safety of the crowd.

It is worth mentioning that these figures and descriptions of situations are from media outlets. As such, it's not the full story. While these statistics might make it seem that general safety and incidents at festivals and the like have gotten worse, it is the opposite, It is generally incredibly safe to attend these events, but using newer technologies we will attempt to enhance it further.

At the same time, festivals in Denmark and the rest of the world have had a focus on mitigating the safety risk in large crowds. This can be seen by the establishment of the Event Safety Foundation in 2015. This is a joint venture between the Skanderborg Festival Group (Smukfest) and Muskelsvindfonden (Grøn Koncert). This foundation has the purpose of securing the safety of the crowds at these two large Danish festivals, as well as many other events in Denmark. They do this through knowledge sharing, professionals and trained volunteers, courses and counseling. This is the company that this project will be done in collaboration with.

To have the best possible outcome for this project, Event Safety invited us to Grøn Koncert and Smukfest in July and August to see how they work and gather security video footage of real crowds at festivals to use as training data. We will also use Event Safety for their practical and theoretical knowledge of crowd safety to use in the system, user feedback and user testing.

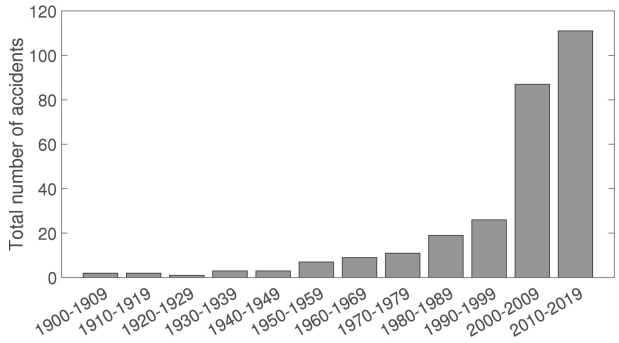
Problem Statement

Can computer vision software and AI techniques be leveraged to improve crowd overview for security guards, by receiving video feed from large crowds, and ultimately improve crowd safety?

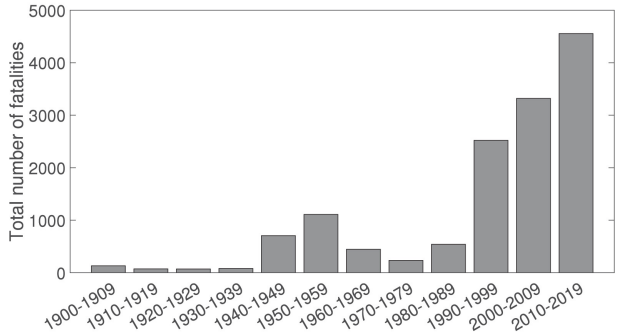
Timeline

Date	Activity
July-August	Gathering data at attended festivals in collaboration with Event Safety

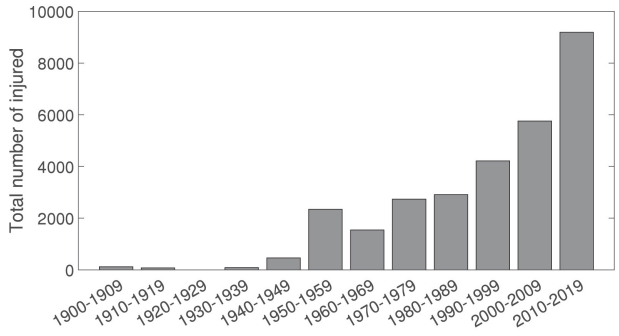
Date	Activity
August	Drafting the project description
31-08-2023	Delivering the project description
09-09-2023	Final discussion with Event Safety regarding the proposed solutions, requirements and the project going forward
September	In-depth analysis of use cases (with Event Safety) In-depth analysis of required technologies and methods used in academic literature. Design of the system
25-09-2023	Progress update with Event Safety regarding the design of the system
October	Implementation of proof of concept system (primarily backend)
26-10-2023	Progress update with Event Safety and small-scale user testing
31-10-2023	Attending Crowd Safety Course hosted by Event Safety in Copenhagen
November	Implementation of proof of concept system
December	Final implementation of the system Documentation of the system User testing and evaluation



(a) Accidents by decade



(b) Fatalities by decade



(c) People injured by decade

Figure 1: Fatalities in crowds graph (Figure from Feliciani et al., 2023)

Analysis

Acronyms and definitions

Acronyms	Definition
CSMS	Crowd Safety Management System
Crowd Counting	The defintion of crowd counting in this report follows the definition from Chan et al. (2008): "[…] a privacy-preserving system for estimating the size of inhomogeneous crowds, […] without using explicit object segmentation or tracking"
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
Crowd Surges	A dangerous crowd situation where your movement is controlled by the crowd and not your own actions.
Mosh pit	A crowd situation where people vacate an area in the crowd followed by violent or aggressive dancing. Sizes of Mosh pits vary greatly.
Technical user	A developer or technician that can implement the CSMS in the technical setup of a concert or venue
Non-technical user	A security guard or someone viewing crowd footage through the CSMS without any technical or software-related background
Choke point	Point with high crowd density and crowd influx
SASNet	Scale-adaptive selection network Song et al. (2021)

System specification

Using cameras mounted around a hotspot of a crowd or on a stationary drone, we aim to develop a platform where an AI model receives live footage from these cameras, uses the model to segment the crowd, and learns what a dangerous situation might look like. This could be one or more of the following (Raineri, 2004):

- Several people moving into the crowd from a specific direction create a dangerous pressure point
- More than a specified amount of people in a marked area resulting in unsafe conditions (ie. >6 people per square meter)
- Omnidirectional or directional movement in the crowd resulting in a dangerous situation
- An area in the crowd suddenly being void of people, perhaps hinting at a mosh pit or an emergency

These are some of the situations this project aims to systematically detect and alert professionals by providing meaningful feedback.

The system would consist of the following:

1. A backend that receives the data from the cameras developed in either Java or Python, with an implementation of a machine learning model to handle the video feed according to the bullet list above and exposing this data through an API
2. A simple frontend or GUI (Graphical User Interface) to display this information in a meaningful way. This could be through a heat map, a numerical estimate of a risk factor, or other visual output.

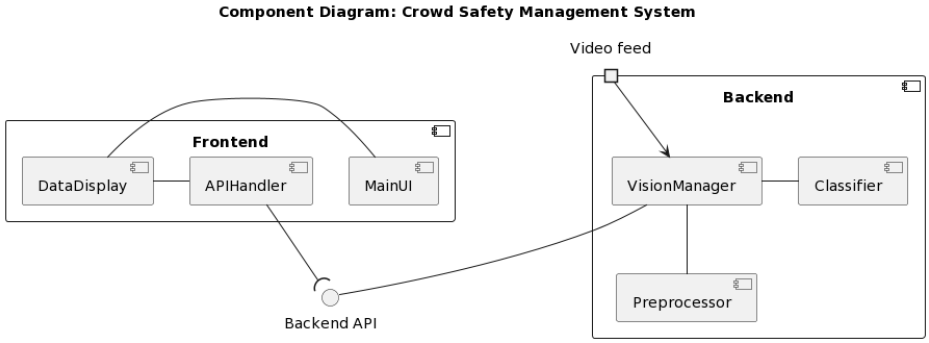


Figure 2: Component diagram of the system

An initial component diagram that maps this described system can be seen in Figure 2.

Functional Requirements

ID	MoSCoW	Requirement
1	M	The system **must** be able to count the amount of people in the crowd with a precision of <100 MAE
2	M	The system **must** be able to segment the crowd into virtual sections for further processing
3	M	The system **must** be able to create a heatmap of the crowd density
4	S	The system **should** be able to calculate the "real" density pr. area unit
5	S	The system **should** be able to detect the movement of crowd sections
6	S	The system **should** be able to correct for camera distortion, warp and perspective
7	S	The system **should** be able to detect choke points in the crowd movements
8	S	The system **should** be able to generate a summarizing report of the concert with statistics of (crowd density, crowd count, risk factors, choke points, and other) relevant safety indicators after the event
9	C	The system **could** be able to estimate a numerical risk factor based on available factors
10	C	The system **could** be able to generate a live, or delayed, video overlaid user interface
11	W	The system **will not** be able to identify dangerous situations that might call for cautionary actions such as crowd surge, mosh pits, falls, blocked exits, etc.
12	W	The system **will not** be able to use data gathered elsewhere at a concert such as alcohol sales, average crowd age, sound, artists, etc. to give a more precise crowd profile and thus risk factor

Non-functional requirements

ID	MoSCoW	Requirement
1	M	The system **must** protect the privacy of the personal data
2	S	The system **should** have a user-friendly interface that is easy to manage for both technical and non-technical users
3	S	The system **should** have adequate documentation / technical specification for technical users
4	S	The system **should** have adequate user manuals for non-technical users
5	C	The system **could** have high reliability that is not based on the visual circumstances and environment (e.g. sunlight, stage light, audience flashlights, and other visual effects) or report confidence based on environment
6	C	The system **could** be scalable to simultaneous interoperability between multiple cameras
7	C	The system **could** integrate with existing CCTV software systems at venues

Detailed Requirements

Risks

More often that not a project will encounter problems that can hinder the progress or outcome. By being well prepared we are able to mitigate some of these risks. The following table describes some of the risks we might encounter in this project.

ID	Name	Affects	Description	Mitigation
R01	Code is lost	Project, product	If the code for some reason is lost or google colab i suddenly insaccessible	Having backups of our code on github
R02	Technology constraints	Product	If the technology previously used in the project suddenly becomes inadequate for the remaining requirements defined in the project	Find alternative technologies or solutions for our requirements before a barrier might be reached. Discuss with supervisor if alternatives exist.

ID	Name	Affects	Description	Mitigation
R03	Personal Conflicts	Product, project	A project related or personal conflict between the 2 members of the project having an effect on either further development or the project as a whole.	Keeping a friendly and open mindset in the work will take us far. Depending on the nature of a potential conflict we would either discuss and solve it outside of work or discuss with our supervisor.
R04	Unusable data	Project, Product	If the data provided by the drone or from EventSafety turns out to be unusable due to one or more factors: Resolution, perspective, Distortion etc.	Making sure that provided data is on par regarding the requirements of our project. Some data might be fixed by using mathematics but in general a proper standard for data is preferred

ID	Name	Affects	Description	Mitigation
R05	Company collaboration ends	Project	If EventSafety or the group decides to end the collaboration around the project early.	Making sure that we listen to and adapt to feedback from EventSafety to keep our good relationship with them

Risk assessment

To fully assess the risks described above it is absolutely vital that the probabality and severity of each individual risk is assessed. This results in the following formula:

$$Effect = Probabilty * Severity$$

The probablity and severity is given each given a score between 1 and 5. 1 being a low probabilty/severity and 5 being a very high probabilty/severity. The effect is then calculated based on these numbers.

ID	Probablity	Severity	Effect	Notes
R01	1	3	3	The severity is based on the amount of code lost
R02	2	2	4	GPU limitations are the primary factor here. However, we dont see the need for more than what google colab can provide.

ID	Probablity	Severity	Effect	Notes
R03	1	1	1	Given the nature of previous work in semester projects and friendship, this is highly unlikely
R04	4	3	12	The severity really depends on how bad the data is and how well we are able to adapt to it
R05	1	5	5	Severity would depend on when in the process EventSafety would cancel our collaboration.

Method and theory

Convolutional Neural Network (CNN) for Computer Vision

Convolutional Neural Networks (CNN) are a type of Artificial Neural Networks (ANN) that are used to "solve difficult image-driven pattern recognition tasks." (O'Shea & Nash, 2015) ANN's function on an input in the form of a multi-dimensional vectors, which will be distributed to a number of hidden layers. The hidden layers will be weighted by evaluating how a stochastic change within itself affects the final output (backpropagation). Deep learning is an ANN with multiple hidden layers. ANN's can be either supervised: Being trained on a set of labeled training data, or unsupervised: Having no labeled training data, but instead minimizing the result of the cost function. According to O'Shea and Nash (2015) and Lempitsky and Zisserman (2010), supervised learning is usually necessary for image-focused pattern recognition tasks.

CNN's are similar to ANN's in almost every way, with the only difference being that CNN's allow to encode image specific features such as edges, textures, color patterns and shape features. This reduces the complexity of a traditional ANN

trained on image data, where a vector representation grows in size exponentially depending on the image size. Reducing the feature vector also reduces the risk of overfitting the model. This is done through the process of convolution. In the convolution layer, local regions of the image vector are scanned for image relevant features. Usually the features are run through a pooling layer, reducing its computational complexity by sampling and down-scaling the features. This feature map is then fed to a traditional ANN, with a given amount of hidden layers.

Solving Scale-Variation with SASNet

Scale variation is one of the main challenges with crowd counting, since perspective in the image will lead to different sizes of heads. One solution to this is proposed by Song et al. (2021). SASNet relies on the fact that heads in a local path share roughly the same scale. It is currently not a common approach to solve scale variation on a continuous scale. SASNet solves it discretly but in 5 discrete steps using a weighted average. This bridges the gap between discrete feature extraction and continuous feature extraction. This architecture can be seen in Figure 3.

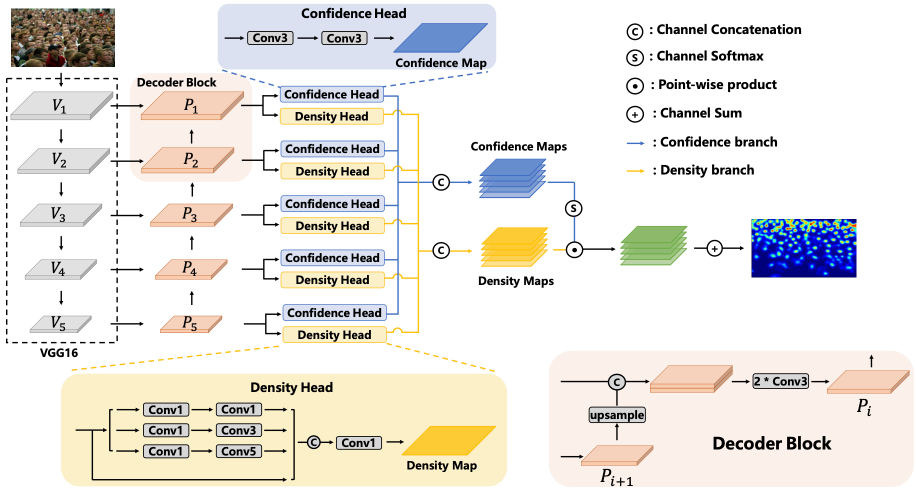


Figure 3: Architecture of the SASNet model (Diagram from Song et al., 2021)

As it can be seen from the diagram, SASNet uses the first 13 convolutional layers from VGG-16 (Simonyan & Zisserman, 2014) in the encoder. SASNet then produces 5 feature levels with 5 levels of downsampling (V_n). It produces 5 levels of predictions (P_n). This is what is referred to as the "U-shaped backbone". These are then fed to a confidence branch and a density branch that produces a confidence map and density map for each of the 5 levels. The complete density map is a product of the sum of the individual density and confidence branches using a softmax function for the confidence.

CUDA

Design

About Crowd Counting

Crowd counting is a field of computer vision research that has developed quickly in recent years. It is a technique that aims to count the instances of any object in an image, no matter the context, density or type of object. (Song et al., 2021) This differentiates it from object detection which is usually trained on one specific type of object by extracting certain visual features. The benefits from using crowd counting rather than object detection is:

- Crowd counting is less intensive on computing resources.
- Crowd counting performs better in crowded scenes with varying scales of objects and overlapping objects.
- Crowd counting is more robust in environments with changes in light, without the need of specific training data.
- Crowd counting is not as exposed to the risk of over fitting the model. (Li et al., 2021)
- Crowd counting does not single out individuals, but looks at the crowd holistically (Chan et al., 2008)

For these reasons crowd counting is a method that can be used for many domains including traffic and parking analysis, pedestrian crowds, corn and crop counting, and much more. This project will use crowd counting methods to analyse, count and estimate density of concert crowds.

(Li et al., 2021) says the current leading approach to crowd counting is the use of a CNN. This paper, contrary to our definiton of crowd counting, includes object detection as a method of crowd counting. It also includes regression based methods and, of course, CNN's. The main challenge with CNN's for crowd counting is to differentiate between the large scale variations in the countable human heads. (Li et al., 2021) discuss several categories of approaches for CNN based models to solve the scale variation problems, one of which is multi-scale fusion. This approach is the category that this paper proposes for further research, as it shows promising results in the balance between performance and accuracy.

For the reasons outlined here from O'Shea and Nash (2015) and Li et al. (2021), this project will focus on the use of CNN based models for crowd counting.

One of the useful datasets for crowd counting, "Shanghaitech" comes from the paper Zhang et al. (2016). In this paper there are defined two datasets, "part a" and "part b". For the purposes of the CSMS test data, "Shanghaitech Part A" is a more representative dataset to use, since it contains more densely populated crowds from a birds eye view. The dataset is labeled with one dot representing roughly the middle of a person's head. This is the approach outlined in one of the founding papers of crowd counting Lempitsky and Zisserman (2010). In Lempitsky and Zisserman (2010), it is explained how their approach to crowd counting, which has become the primary method in the field, is to create a density

function F as a function of the pixels in an image I. This density function is analoguous to the physical idea of density, however not a direct representation of the same concept, as physical density is based on physical area. Integrating F over the entire image I will return the number of people in I. (Lempitsky & Zisserman, 2010) Each object in the image I should be represented by a normalized 2D Gaussian kernel with the mean at the person's head. One disadvantage with this approach, is that the kernel for objects close to the edge of the image will not be counted as a whole object when summed. It is not assumed that this will pose a significant source of error for the purpose of the CSMS.

Regarding SAM model for Crowd Counting:

One of the original ideas for solving this problem was to use Meta's SAM model. During the preliminary research sessions, it was discovered that SAM was not adequate in counting people but better suited for object detection. The following citation from Ma et al. (2023) highlights the problems well:

"Although the Segment Anything model (SAM) has shown impressive performance in many scenarios, it currently lags behind state-of-the-art few-shot counting methods, especially for small and congested objects. We believe that this is due to two main reasons. Firstly, SAM tends to segment congested objects of the same category with a single mask. Secondly, SAM is trained with masks that lack semantic class annotations, which could hinder its ability to differentiate between different objects. Nevertheless, further exploration of adapting SAM to the object counting task is still worth studying." (Ma et al., 2023)

The choice of SASNet

The choice of using SASNet was based on three different factors:

1. SASNets fits our need to solve the problem of scale variation in the video material. It was not possible to obtain video material from an approximate orthographic perspective. However, future endeavors for this project might include cameras like those from one of our collaborators PhaseOne who produces approximate orthographic perspectives using high pixel density cameras from high altitudes. SASNet, however, fulfills the requirement to produce relatively high accuracy on an image with perspective for now.

2. SASNet has some of the lowest absolute errors on the ShanghaiTech testing data, according to Gong (2023) which compares at least 45 different crowd counting models and methods.

3. SASNet is open source and licensed under the permissive Apache 2.0 license, which allows us to use it legally and for free in our project. SASNet also publishes the pretrained model weights (trained on ShanghaiTech A and B), which means we can save many compute- and time resources on training the model ourselves.

Finally, SASNet publishes the model weights from the training on either ShanghaiTech part A or part B. For this project, it is deemed more useful to use the

images from part A. While the labeled dataset is smaller in part A than part B, it more closely represents the data (i.e. amount of people, density and perspectives) that will be used in this project.

Activity diagram

Figure 4 illustrates the flow of the program through an activity diagram. The diagram was created to obtain a deeper understanding of the general flow and processes the program goes through during its runtime. The diagram is split into 2 "swimlanes", one lane for our CSMS system and one lane for the implementation of the SASNet model. In broad terms, this can also be specified as a CPU and a GPU lane where our CSMS system handles CPU processing and SASNet focuses on GPU processing.

As depicted in the diagram, section of the activities are divided into partitions to visualise a rough partioning into classes and for better readability throughout the diagram. The input is specified as the path to the pretrained model and the path to the video for it to run on.

Below is a short runthrough of the different partitions. A more indepth description can be found in the implemetation section

1. System Initialization and Input Validation

The CSMS is initialized and the inputs for the model and video path are validated

2. SASNet Model Setup and Construction

The SASNet model is loaded together with VGG16 model which it is based on. This sets the foundation for subsequent image processing.

3. Pre-Processing of Input Data

Preprocessing involves creating a dataset from the video in pyTorch and in turn creating a dataloader from this dataset.

4. Image processing Loop

The images in the dataloader are iterated through and forwarded to the SASNet model until the dataloader is empty. Each image processed through the different layers to prepare them for post-processing

5. Post-Processing & Result Generation

Following the image processing, the CSMS takes over again. Here density information is computed, perspective warping is applied to in turn generate the correct heatmap for further analysis.

6. Output Generation & Reporting

All of the crowd counting data is saved to a CSV filed and the images are stitched together to form a timelapse and a report, summarizing the observations.

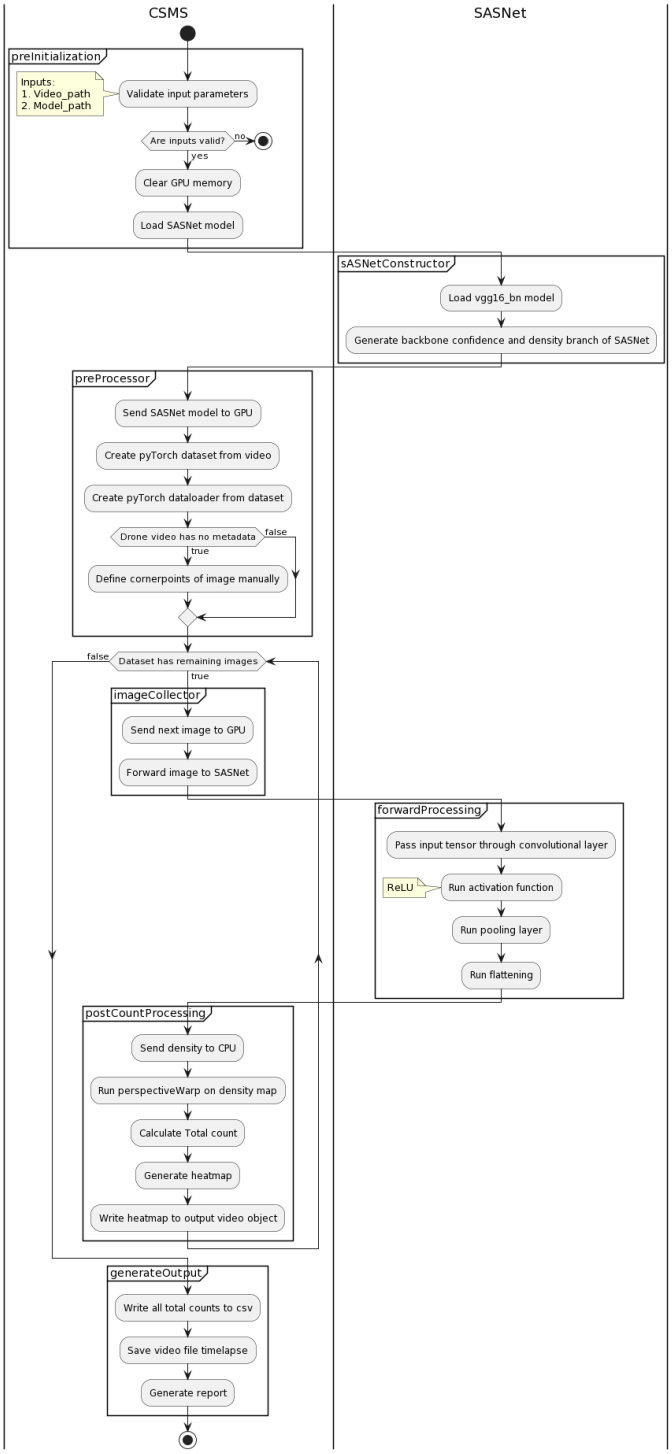


Figure 4: Activity diagram depicting flow in the solution

Implementation

Data Collection and Preparation

To implement this project it was necessary to gather useful evaluation data. For the purposes of this project, that is video security footage and drone footage. Our collaborative partner EventSafety was very helpful in letting us gather this data at their 2 festivals, "Smukfest" and "Grøn". This was done 1-2 months before the project started since this is when the festivals were held. This means that we gathered the data before completing the research on crowd counting and computer vision. It did pose a challenge to collect the data before knowing exactly what type of data was needed. For these reasons, we collected many types of data, including:

1. Drone footage from "Grøn" at an altitude of 25-50 meters from the entrance
2. Drone footage from "Grøn" at an altitude of 25-50 meters of the concerts and end of concerts
3. GoPro footage from "Grøn" at an altitude of 2-5 meters from the entrance
4. Security camera footage from "Smukfest at an altitude of approximately 10-15 meters from the center of the stage.
5. Security camera footage from "Smukfest" at various heights from bars and paths

What turned out to be most useful was option no. 2, drone footage from "Grøn" at an altitude of 25-50 meters of the concerts. Having a perspective from a higher altitude means that more people are clearly visible and also minimizes distortion when warping for perspective correction. Having an even higher resolution from a higher altitude could also increase the accuracy of the predictions. The resolution of the drone footage was 1080p. Having a higher resolution could limit the video compression artifacts. The camera options were discussed with our secondary partner PhaseOne. They have experience with ultra-high-resolution top-down drone footage. This could be an option for future endeavors, which will be discussed in Section section of this report.

While we, with the help of a certified drone pilot, collected the footage from "Grøn" ourselves, the security footage from "Smukfest" was collected by EventSafety. For this, we created a technical specification with our requirements for the footage at the time. This can be read in Figure 5 in Section .

Environment

Setting up an environment for machine learning in Python can pose a bit of a challenge. Is a requirement to have access to a NVIDIA GPU to take advantage of the CUDA toolkit in PyTorch. This opens up GPU acceleration which enables a large performance boost rather than running the CNN models on the CPU. Furthermore, the SASNet open source projects were built for Python 3.6.8 and PyTorch >= 1.5.0 Song et al. (2021).

Google Colab

Because of these strict software and hardware requirements, we decided to use the virtual environment in Google Colab. Google Colab provides a preconfigured Python environment with (free but not unlimited) access to a NVIDIA GPU. Doing this project in the cloud removes the software and hardware limitations on both the users and developers of the project. However, Google Colab has some feature limitations such as it being disallowed to use it for the purpose of hosting web services. Google Drive can then be volume mounted inside the virtual environment making it possible to use and make persisted files.

The Python version in Google Colab is downgraded to match the version needed for SASNet and both PyTorch and the required PIP packages are installed.

UCloud

University of Southern Denmark provides a cloud computing service for faculty members through eScience. Our supervisor applied through this site for a cloud computing resource with 100 GPU hours.

SASNet Model Adaption

The SASNet model is loaded using pre-trained weights and biases for the underlying vgg16_bn model trained on ImageNet. This allows SASNet to more efficiently identify image features with low memory usage. Furthermore, the SASNet weights and bias fine-tunings are loaded into the model which has been trained on the Shanghai Tech Part A dataset. A block size of 32 is being used here.

```
class Args:
  model_path = "<path-to-model-dir>/SHHA.pth"
  video_path = "<path-to-input-video-dir>/DJI_0461_trimmed.MP4"
  block_size = 32
  use_pretrained = True

model = SASNet(args.use_pretrained, args).cuda()
model.load_state_dict(torch.load(args.model_path))
```

A quick note on block size. Block size is important because when loading data into a CNN model (or any model for that matter) minimal data variance is key. For this reason, a block size is added to make sure that each data input is the same length, see example below:

```
Hello SDU
This is a cool project
```

say our input block size was set to 2, this would make sure that each of those lines was truncated to a set length, in this case, a length of 10:

```
Hello SDU 0 0     0       0 0 0 0 0
This  is  a cool  project 0 0 0 0 0
```

Integeration

Optimization and Performance

For two main reasons, it is necessary to limit the compute resource usage:

1. This project is using free and limited GPU and compute resources. Computer vision and CNN's can be very GPU intensive, so it is in the project's to limit the GPU memory consumption as this was found to be the largest resource bottle neck.

2. For the CSMS to be scalable to (near) real-time evaluation performance, it makes sense to limit the use of computing power as much as possible.

According to Contributors (2023), disabling gradient calculation is useful for lowering memory usage when using a model for inference. This fits the use case and greatly reduced the memory usage. This was done using the following way:

```
img = img.cuda()
with torch.no_grad():
  model.eval()
  pred_map = model(img)
pred_map = pred_map.data.cpu().numpy()
```

Here the image tensor is first sent to the NVIDIA GPU using img.cuda(). Gradient calculations are then disabled for the inference, and the model is set to eval mode. These two reduce the GPU memory usage. Finally, after the inference is run and the predicition_map is calculated, the tensor data is sent to the cpu and the tensor is converted to a numpy matrix (an image). This way limits the time that the tensor spends in the GPU greatly, as well as reducing the memory usage.

Error Handling, Security and Privacy

Error Handling

One of the major error-handling implementations made is checking if the data uploaded to the model is present and valid. If either of these checks fail, the code is not run and the program shuts down.

Security

Privacy

Being citizens in and using data of people from an EU country means that we have to abide by the rules of GDPR. This means that security and privacy around the data that we have acquired is crucial. Practically, this means that we have a responsibility in making sure that we only upload the images and videos we

need to run the model on when they need to be there and remove them when not needed. In a perfect world we would run our program locally as to limit the uploading and removal of sensitive data.

Facial recognition vs. Crowd Counting

When reading about this project, one might assume that privacy around facial recognition would be an issue. However, as this is related to heatmaps and crowd density instead of direct recognition of people, the identity of people in our footage is not a major concern, and thus was not featured in our risk management section. The CSMS does not track individuals through time, which could be a privacy concern.

Validation and Verification

Since this project is a proof of concept, unit tests and integration tests are not as favored as user tests and similar. Testing of this system relies on making sure that it is useful and understandable to those who need it. To accomplish these tests, we decided to get the opinions of some EventSafety security personnel about our project and their thoughts.

EventSafety employee questionnaire

User interview nr 1: Question 1 (Usefulness of provided video/heatmap/report) Question 2 (Likelihood of implementing this for post-event evaluation) Question 3 (How could data like this improve your post-event evaluations) Question 4 (Data readability) Question 5 (Other thoughts and notes from interviewee)

User interview nr 2: Question 1 (Usefulness of provided video/heatmap/report) Question 2 (Likelihood of implementing this for post-event evaluation) Question 3 (How could data like this improve your post-event evaluations) Question 4 (Data readability) Question 5 (Other thoughts and notes from interviewee)

Questionnaire evaluation

Runtime and GPU usage

While GPU usage is something already discussed in implementation, runtime and the optimization of this is still crucial to test. While we originally wanted this project to be able to run in real time, we quckly realised that this was close to impossible with our current setup and implementation. When running the model on a video, we originally ran it on a 3 minute clip with an image every 10 seconds being processed. This resulted in a 6 second video and a runtime of the program of around 15 minutes. Not great, not terrible. In the future, if we want to run this live, additional ressources are required. More about this in Section

Discussion

TODO: Add section on possibilities with PhaseOne

Conclusion

Perspective

References

Appendices

References

Chan, A., Liang, J., & Vasconcelos, N. (2008). Privacy preserving crowd monitoring: Counting people without people models or tracking. *Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR.2008.4587569

Contributors, P. (2023). *Pytorch documentation* [Version 2.1.0]. https://pytorch.org/docs/stable/index.html

Feliciani, C., Corbetta, A., Haghani, M., & Nishinari, K. (2023). Trends in crowd accidents based on an analysis of press reports. *Safety Science*, *164*, 106174. https://doi.org/https://doi.org/10.1016/j.ssci.2023.106174

Gong, J.-Y. (2023). Awesome-crowd-counting [GitHub repository].

Lempitsky, V., & Zisserman, A. (2010). Learning to count objects in images. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, & A. Culotta (Eds.), *Advances in neural information processing systems* (Vol. 23). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2010/file/fe73f687e5bc5280214e0486b273a5f9-Paper.pdf

Li, B., Huang, H., Zhang, A., Liu, P., & Liu, C. (2021). Approaches on crowd counting and density estimation: A review. *Pattern Analysis and Applications*, *24*, 853–874.

Ma, Z., Hong, X., & Shangguan, Q. (2023). Can sam count anything? an empirical study on sam counting.

O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *CoRR*, *abs/1511.08458*. http://arxiv.org/abs/1511.08458

Raineri, A. (2004). The causes and prevention of serious crowd injury and fatalities at outdoor music festivals. https://doi.org/10.13140/2.1.3036.0005

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Song, Q., Wang, C., Wang, Y., Tai, Y., Wang, C., Li, J., Wu, J., & Ma, J. (2021). To choose or to fuse? scale selection for crowd counting. *The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)*.

Zhang, Y., Zhou, D., Chen, S., Gao, S., & Ma, Y. (2016). Single-image crowd counting via multi-column convolutional neural network. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Teknisk specifikation til indsamling af data

Syddansk Universitet, Teknisk Fakultet, d. 10/7-2023

Teknisk specifikation til indsamling af data

Formålet med denne tekniske specifikation er at beskrive kravene til indsamling af videomateriale fra en festival til brug i et crowd safety management system. Projektet laves af studerende på Syddansk Universitet. Kontakt oplysninger:

Anton Irvold: anirv20@student.sdu.dk

Christoffer Krath: chkra19@student.sdu.dk

Krav til kameraer

- Kamera skal have tilstrækkelig opløsning til at kunne adskille mennesker fra hinanden (1080P og opad).
- Kamera skal være vidvinkel så fremt muligt.
- Kamera skal optage i 30 FPS så fremt muligt.

Krav til placering af kameraer

- Kameraer skal placeres på strategiske steder, såsom foran scener, indgange, udgange og områder med høj trafik.
- Kameraer skal placeres i en højde på mindst 3 meter eller højere for at undgå at blive blokeret af mennesker.
- Kameraer skal placeres i en vinkel, der muliggør dataindsamling af bredest muligt område.

Krav til dataindsamling

- Data skal gerne ligge på en harddisk eller cloud hvorfra det kan tilgås kort tid efter.

Figure 5: Technical specification document for cameras