INSTITUTO TECNOLÓGICO DE BUENOS AIRES - ITBA
ESCUELA DE INGENIERÍA Y GESTIÓN

People Counting Using Visible And Infrared Images

AUTORES: BIAGINI, MARTÍN (Leg. 56343)
FILIPIC, JOAQUÍN (Leg. 56266)

TUTOR: PARISI, DANIEL RICARDO

TRABAJO FINAL PRESENTADO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN INFORMÁTICA

LUGAR: BUENOS AIRES
FECHA: 19/10/2020

Abstract

We propose the use of convolutional neural networks that consider as input four channels images (RGB+IR) for counting and positioning people in images. Our data set was made of images based on photographs taken from a drone using a dual FLIR camera.

Comparison between 3 (RGB) and 4 (RGB+IR) channels are studied for different lightning conditions. The four channel network performs better in all situations, particularly in cases of poor visible illumination that can be found in real night scenarios. The average precision of this network on a testing data set (independent from the training one) is approximately 1 cm in finding the positions of pedestrians (from 15 and 30 m altitude images) and 0.0001% in the relative counting error.

Contents

1 Introduction

Crowd analysis is a labor of known relevance and interest at a global level. It is necessary for its various applications: from the social and political perspective, it is important to obtain data and draw conclusions from demonstrations; from the security point of view, an accurate calculation of densities can facilitate the design of emergency exits and evacuation methods; for informational or statistical purposes, it is a basic task but does not relegate utility.

There are several methods to process images and calculate the total amount of people present [1, 2, 3]. In particular, the use of AI for this work is already widely known, and is addressed by various investigations starting from data sets of crowds images and using convolutional neural networks (CNN) that obtain favorable results [4, 5, 6]. In [7] accurate density maps are obtained as output, commenting that the difficulty lies in particularly dense congregations. However, this study is limited by the simple fact of relying on the visible information provided by the images, so an extra complexity is added when considering dark environments, where the light is scarce and irregular, or where the visible spectrum mix (due to external light sources).

The need to add extra information that helps the network to individually discriminate the position of each person within a crowd comes into play when the visual conditions are not optimal. Starting from [8], in which a method is presented to obtain said positions using a dual-camera (visible + infrared) mounted on an Unmanned Aerial Vehicle (UAV), this study continues and presents the novelty of using a special type of a CNN (U-Net CNN) to evaluate images with their data distributed in 4 channels, 3 for the visible spectrum (RGB) and one for the infrared. The latter, by providing thermal information, will allow a more accurate analysis in these scenarios.

Following the research line of previous works such as [9], which take advantage of the combination of visual and infrared images, the present implementation consists of comparing the precision of a neural network trained with images with the first 3 channels against a trained one with images with the 4 channels (from now on, the first network will be referenced as CNN_3 and the second network as CNN_4), and observing substantial improvements in the obtained results when evaluating cases using the second one. The images that make up the data set are specially selected considering environments with conditions similar to those mentioned, where visual information is scarce, representing aerial shots obtained from a drone (UAV).

2 Methods

2.1 Data

Naturally, both the CNN_3 and the CNN_4 will use the same data sets, and the first thing to mention is the discernment between the training set, the validation set and the testing set [10], which is mutually exclusive from the other two. The main reason for this division is to separate the images that will be used to train the networks from those that will be used to test it.

For the training phase, a data set containing training images and validation images will be used: 90% of the images will be training images, which are the ones provided to the networks to process and update their weights; the remaining 10% (chosen randomly) will compose the validation images, which will be used to check the accuracy of the networks at the end of each epoch (and decide when to complete the training). On the other hand, the testing set has independent images, on which

different metrics will be analyzed and conclusions will be drawn from their results.

Before detailing the composition of each data set, it is necessary to explain how each image was obtained. Real photographs were provided to us, taken by a FLIR DUO PRO R camera attached to a Unmanned Aerial Vehicle (UAV, also known as a multi-rotor drone). These photographs were obtained in the context of a previously on going research project at the "Centro de Agentes Físicos, Biológicos y Sociales": ITBACyT2018-42.

These photographs were taken at 15 m and 30 m altitude of small congregations of people distributed differently in an open field during the day, with full visible light. Each of these images has its IR component, thanks to the technology provided by the camera. Therefore, the following method is used to generate the final images to be used by the networks. First, from the photographs, cut-outs of people and backgrounds are made to use as templates (considering both visible and infrared information). Then, we simulate different crowd configurations (number of people and positions) by performing simulations with the Social Force Model [11]. Different number of simulated pedestrians were generated randomly inside the area, then moving them toward different targets. Positions during the evolution of trajectories were registered, and different configurations and densities were obtained.

Using these positions, we overprint the cut-outs of the people from the real photographs over a given background. Backgrounds were also taken from real photographs and artificially generated by noise. After that, and in order to erase border artifacts of cut-out images we pass a blurring function and add random noise. Blurring is implemented using the technique of a Gaussian function (with a kernel of side $k = 7$ and a standard deviation $\sigma = 1$). The noise added to every image is achieved with a percentage of randomness in each pixel, using as a range $[I_p * (1 - \beta), I_p * (1 + \beta)]$, where I_p is the pixel value and $\beta = 0, 1$.

In this way, representations similar to reality are generated. The obtained images used in the data sets differ from each other in terms of a series of configurable variables (including the positions of agents):

- The attenuation factor for the intensity of the RGB components ζ_{RGB} (the intensity is the pixel value of each channel, an integer within the range of [0, 255]).
- The attenuation factor for the intensity of the IR component ζ_{IR}
- The type of background B (they vary both visually and in temperature).
- The distance A that would represent the altitude at which the photo was taken. Associated with this variable is the amount of people present in the image (depending on the altitude, there is certain amount within a range).
- The people distribution D (their positions in the shown area).
- The composition of the background IR channel T (backgrounds are self-generated with pixel values that correspond to certain temperatures, within a range). This variable is considered only for the formation of the testing set, not for the training set.

Every image has its 4 channels (the RGB in one file and the IR in a separated one), with a resolution of $1024x768$ pixels. Internally, the 4 channels are used together to be processed by the CNNs (in the case of the CNN_3, the last channels is ignored).

Examples of real photographs with different people densities and generated images of different altitude and amount of people can be seen in Fig. 1.

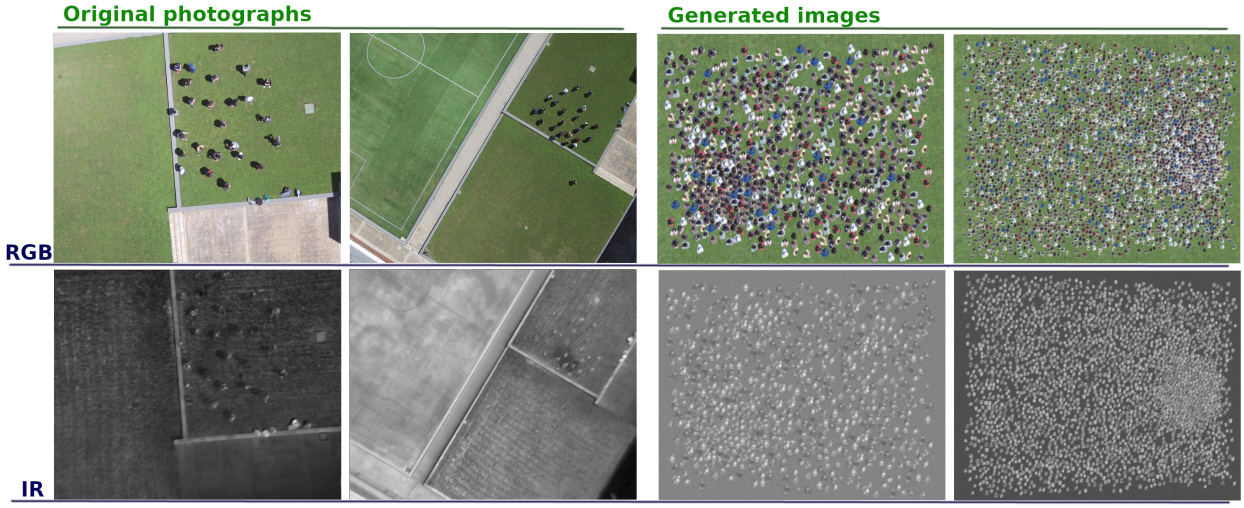

Figure 1: On the left, examples of real photographs taken at 15 m and 30 m altitude, with their corresponding IR channels. On the right, examples of generated images at the same altitudes using the cut-outs from the first ones.

With that being said, on one hand we have the data set for the training and the validation phase of the CNNs. Eq. 1 shows the combinations of the possible values for each variable involved in the generation of this set. Taking into account all combinations between them, we have 1530 images.

The value of the people distribution simply indicates that 5 different instants of the simulation (hence, the people positioned in different places on the ground) were considered for each of the other variable's combinations. For the latter, Table 1 shows their possible values. A few clarifications are made: firstly, cases annotated (*) are repeated twice, and the case annotated with (**) thrice (giving 17); secondly, the total number of people in each image, associated with the altitude, is chosen randomly from the specified range, with a uniform distribution of probabilities

$$S^{tr} = \zeta^{tr}_{RGB} * \zeta^{tr}_{IR} * B^{tr} * A^{tr} * D^{tr} = 17 * 3 * 3 * 2 * 5 = 1530 \quad (1)$$

ζ_R^{tr}	ζ_G^{tr}	ζ_B^{tr}		ζ_{IR}^{tr}	B^{tr}	A^{tr}
1	0	0		0.6	Cement	30 m/[400,2800]
0	1	0		1	Grass	15 m/[100,700]
0	0	1		1.2	Synthetic Grass	
0.01	0.01	0.01	*			
0.05	0.05	0.05	**			
0.2	0.2	0.2				
0.4	0.4	0.4				
0.6	0.6	0.6				
0.8	0.8	0.8				
1	1	1	*			

Table 1: Possible values for each variable of the training and validation set S^{tr}.

On the other hand, as mentioned previously, independent images were used to carry out the prediction tests, obtain results and draw conclusions, once the networks were trained. This testing set is formed combining the possible values of the variables indicated in Eq. 2, obtaining 3456 images. In the same way as before, table 2 shows the possible values for these variables.

Later on, in Sec. 3, the 6 cases divided by the combination of the RGB attenuation factor ζ will be treated separately. As previously said, the reason for an alternate set is to avoid making a prediction with an image that was previously used to update the weights of the networks during the training phase.

$$S^{ts} = \zeta_{RGB}^{ts} * \zeta_{IR}^{ts} * B^{ts} * T^{ts} * A^{ts} * D^{ts} = 6 * 3 * 3 * 4 * 2 * 8 = 3456 \tag{2}$$

ζ_R^{ts}	ζ_G^{ts}	ζ_B^{ts}	ζ_{IR}^{ts}	B^{ts}	T^{ts}	A^{ts}
1	0	0	0.6	Cement	Real/-	30 m/[400,2800]
0.01	0.01	0.01	1	Grass	Generated/[15 °C,20 °C]	15 m/[100,700]
0.05	0.05	0.05	1.2	Synthetic Grass	Generated/[30 °C,35 °C]	
0.1	0.1	0.1			Generated/[45 °C,50 °C]	
0.2	0.2	0.2				
1	1	1				

Table 2: Possible values for each variable of the testing set S^{ts}.

2.2 Model

The implementation carried out consists mainly of the convolutional neural network, which is fed by the images of the data set to be trained first and perform the evaluations later, and a series of functions to transform its output and obtain the desired information.

First of all, it is also necessary to have the exact positions of the people present in each of the images to be used as input. As these images are obtained from the previously described method, we can count on these positions throughout the simulations, since they are recorded at all times. So, for each image we have, in a separate file (called "ground-truth"), the position of each person, represented in a matrix of integers as a coordinate map, with value 0 where there are no people and

value 100 in the center of each of them. Based on this, the CNN, after processing the information of each image, will be able to compare the predictions of the positions with the actual positions.

Considering the neural network, it was trained using the incremental methodology in each epoch, and its architecture is based on an evolved design of normal convolutional neural networks: the U-Net. The use of this design was determined for 2 main reasons. The first one is that it is aimed both at being able to classify the patterns recognized in the inputs and at being able to locate them spatially (exactly what is required in images of crowds of people). The second, thanks to the series of functions that are being applied internally, is that it offers the possibility of using an input image data set that is not necessarily extensive, as if a common neural network would require it.

In particular, the implementation used here is based on the article [12] and the code found in [13], consisting of repeated applications of two $3x3$ convolutions, each followed by a rectified linear unit ReLU ($f(x) = max(0, x)$, with x being the input of the neuron) and a $2x2$ max pooling operation with stride 2 for the downsampling phase. For the upsampling phase, successive techniques are also performed to return the image to its original dimension using a $2x2$ transposed convolution, a concatenation with the corresponding image in the downsampling phase (to combine the information of previous phases) and two $3x3$ convolutions followed by ReLU.

The differences between this implementation and the original model described in the aforementioned work are the application of a batch normalization of the values of each map between the convolutions and the rectification in the downsampling phase, the use of 3 downsampling/upsampling stages instead of 4 and the inclusion of the 4th channel in the input images. This is summarized in the Fig. 2.

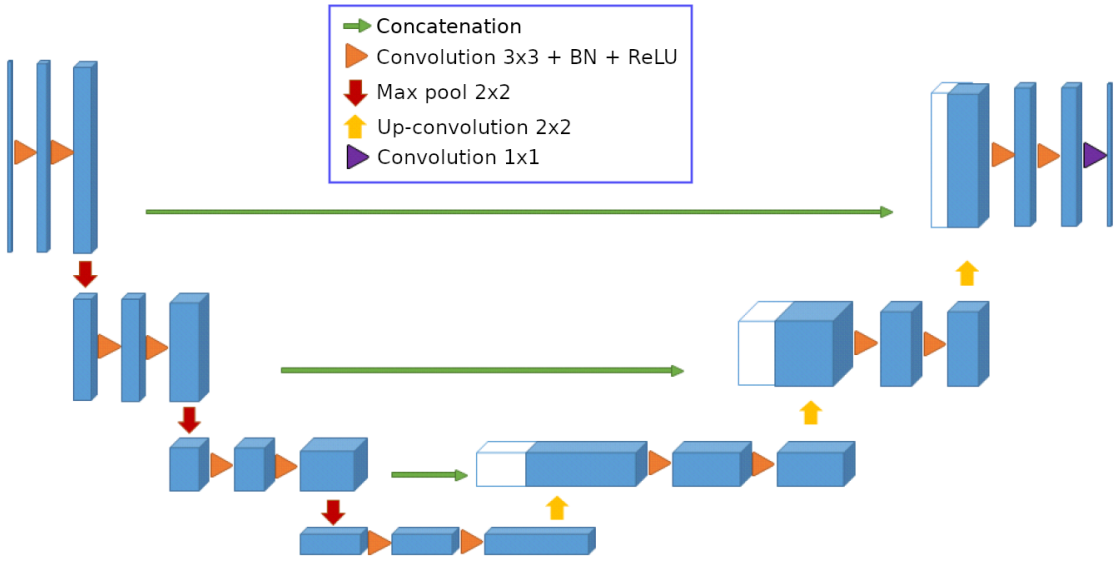

Figure 2: U-Net architecture. First block (top left) is the input image and last block (top right) is the output of the network.

The last thing referring to the network's architecture is the use of the following variables for its training: Nesterov Momentum $\alpha = 0.9$, learning rate $\eta = 0.02$, weight decay $w = 0$, $\gamma = 0.1$ and

step size $s = 10$, the latter being the step of epochs through which η is updated, multiplied by the factor γ.

Regarding the network's output, another matrix of integers is obtained, where the predicted positions of the people are indicated, although further processing is necessary. This output matrix has the same dimension as the resolution of the images that are used as input (which are transformed to similar matrices when feeding the network, but with an additional dimension that contains the information of the RGB and IR channels), that is, $1024x768$.

The necessary processing consists of looping through the matrix, composed mostly of values of 0, which indicate that there are no people present in the element (x, y) where we are looking. When predicting the central position of a person, this is reflected with values greater than 0 in a certain element of the matrix, so we need to identify these elements. However, it is necessary to observe the neighboring elements every time a value other than 0 is found, add their values (since there are cases in which the predicted central position is translated in the matrix as more than one element with value greater than 0) and leave them concentrated in the element where the highest value is found. This unification process can be seen in Fig. 3.

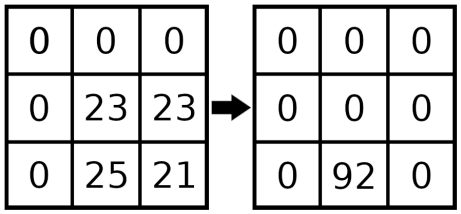

Figure 3: Looping through every element of the network output map (left), neighbours with values higher than 0 must be unified to obtain a final map indicating the predicted positions of the people (right).

Summarizing, the CNN will provide as output the positions of all pedestrians present in the input image.

It is worth mentioning that during the training of the network, the unification process does not take place: to determine the completeness of the training, at the end of each epoch a comparison value is obtained to determine its level of precision. This comparison is made on the images conforming the validation set (differentiating it from the training set, which is used specifically to update the weights of the network). It is calculated as expressed by Eq. 3, which is the same function used by the CNN as its loss function.

$$P_{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{X}_i - \mathbf{Y}_i)^2 \tag{3}$$

where $\mathbf{X}$ is the ground-truth matrix, $\mathbf{Y}$ the network output matrix and n the number of elements in each of them. This is the mean squared error taken between each element of the initial matrix in the ground-truth file (the one that provides the real people positions) and the matrix of the output map of the networks (with no further process, like the left matrix on Fig. 3).

2.3 Metrics

First, while training the network, the best combination of the connection weights will be recorded. Each epoch is compared with the previous one, keeping the one with least P_{MSE}, as described before in Eq. 3. The cut-off criteria is the passage of 30 epochs without registering a minor error.

Prior to calculating network efficiency with the independent testing set, it is necessary to determine the threshold value μ that must be exceeded to be able to discern whether or not a person is detected in a predicted position on the final matrix (after the unification process). This value is found empirically, averaging the absolute mean error calculated between the predicted number of people and the real number of people for each image in the training and validation sets, as noted in Eq. 4

$$N_{MAE} = \frac{1}{n}\sum_{i=1}^{n}|pn_i - rn_i| \tag{4}$$

where pn the predicted number of people, rn the real amount and n the number of images in the set.

Finally, for the analysis of results, there are 2 metrics that refer to the precision of the network processing a single image. The first one is the relative error of the number of predicted people versus the real number. It is shown in Eq. 5

$$E_{num} = \frac{1}{N_r}|N_p - N_r| \tag{5}$$

where N_r is the real number of people and N_p the predicted number.

The second metric is the average error of the distance between the predicted position and the real one (in cm), for each position that has its real pair (that is, excluding people who were not counted and those who were "added"). It is shown in Eq. 6

$$E_{pos} = (\frac{1}{n}\sum_{i=1}^{n}|pp_i - rp_i|) * \psi \tag{6}$$

where n is the number of pairs, pp the predicted positions, rp the real positions and ψ the multiplication factor implying the distance per pixel given the altitude of the image, according to the results in [8]. An extra mention is needed, since the criteria to associate a predicted position to its real pair must be determined.

In order to do this, the algorithm described next is used. Having the list of predicted positions and the list of real positions, for each of the predicted positions the closest real position (euclidean distance) is analyzed. At this point, it is necessary to check if there is another predicted position that is closer to the selected real one, and follow with a similar reasoning to ensure that the correct predicted position - real position pair is chosen. Upon obtaining it, the error between the two is calculated and each one is removed from its respective list, to do the same with the rest. At the end, there may be more predicted positions or actual positions, which will not be taken into account for this metric .

3 Results

3.1 Training and Validation

Since the main objective of this work is to compare the performance of the CNN_3 against the CNN_4, it is taken into account that all measurements were carried out against both of them. The results of the training phase are shown in Table 3. The number of epochs run by each CNN implies the needed epochs to find the lowest P_{MSE}.

CNN	Epochs run	P_{MSE}
CNN_3	29	12.61
CNN_4	19	11.51

Table 3: Training results comparison between CNN_3 and CNN_4: number of epochs run by each network and lower P_{MSE} found on the images of the validation set.

This simply indicates that the CNN_4 needed fewer epochs to obtain a smaller error and perform better at this stage than the CNN_3.

Then, to determine the optimal threshold μ to be used as a criterion in the detection of a person given the output map, the entire training + validation set was run with the CNN_4 and the N_{MAE} was calculated, using different μ (from 10 to 90, with step 5). The results can be observed in the Fig. 4, and the best value for μ is 30 (being $N_{MAE} = 6\ 10^{-4}$). From this results, though, we can say that any value between 25 and 45 can be used to perform similar subsequent evaluations.

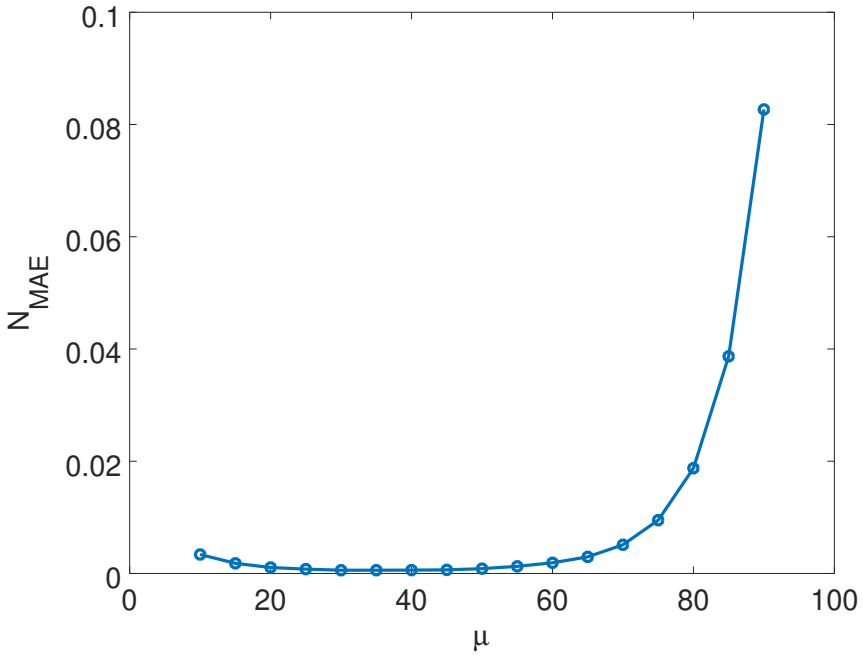

Figure 4: Calculated N_{MAE} of the entire training + validation set S^{tr} for different values of the threshold μ, using the CNN_4.

3.2 Testing

Being able to measure the errors in the predictions of the images of the testing set, using the 2 metrics described in Sec. 2.3, E_{num} and E_{pos}, and the threshold $\mu = 30$, the testing set is divided into the 6 cases mentioned in Sec. 2.1 (based on each combination of the RBG intensity). Basically, this division is made from the amount of light present in the images, represented by the attenuation factor ζ for the R, G and B channels.

In this way, case 1 is composed of monochromatic images ($\zeta_R = 1$, $\zeta_G = 0$, $\zeta_B = 0$), and from case 2 to case 6 they are composed of images with incremental values of the amount of light: respectively, $\zeta_{RGB} = 0.01$, $\zeta_{RGB} = 0.05$, $\zeta_{RGB} = 0.1$, $\zeta_{RGB} = 0.2$ and $\zeta_{RGB} = 1$ for each consecutive case.

Measurements were taken using the CNN_3 and the CNN_4, seeing the corresponding results on Table 4 and Table 5 in which the values of all the images in each case are averaged. For compute E_{pos} the correspondence conversion factor between pixel/cm was calculated taking into account the distance that represents the altitude of each image (reason for which it is specified in that unit). The ratios obtained are 1.25 cm/pixel for height 15 m and 2.5 cm/pixel for height 30 m, based on the calculations in [8].

	E_{num}		$E_{pos}(cm)$	
Case	Mean	Std. Deviation	Mean	Std. Deviation
1	$1\,10^{-3}$	$2\,10^{-3}$	1.1	0.3
2	$2\,10^{1}$	$2\,10^{1}$	-	-
3	$1\,10^{-2}$	$2\,10^{-2}$	2	1
4	$1\,10^{-3}$	$1\,10^{-3}$	1.0	0.4
5	$2\,10^{-4}$	$9\,10^{-4}$	1.0	0.4
6	$2\,10^{-4}$	$8\,10^{-4}$	1.0	0.3

Table 4: Prediction results of the CNN_3: mean and standard deviation of E_{num} and E_{pos} for every image in each case of the testing set S^{ts} (divided by the value of ζ_{RGB}).

	E_{num}		$E_{pos}(cm)$	
Case	Mean	Std. Deviation	Mean	Std. Deviation
1	$1\,10^{-3}$	$5\,10^{-3}$	1.0	0.4
2	$1\,10^{-2}$	$3\,10^{-2}$	1	1
3	$1\,10^{-3}$	$4\,10^{-3}$	1.0	0.4
4	$1\,10^{-3}$	$1\,10^{-3}$	1.0	0.3
5	$1\,10^{-4}$	$5\,10^{-4}$	1.0	0.3
6	$2\,10^{-4}$	$6\,10^{-4}$	1.0	0.3

Table 5: Prediction results of the CNN_4: mean and standard deviation of E_{num} and E_{pos} for every image in each case of the testing set S^{ts} (divided by the value of ζ_{RGB}).

Comparing the results of case 1, it can be observed that the two CNNs show similar values in both metrics. Given that this case is considering monochromatic images, this behavior was as expected, because the information provided by this images can be exploited by both CNNs equally.

The greatest difference is observed, clearly and as expected, between the metrics of case 2, corresponding to the dark images. The CNN_3 is unable to predict decent results and throws out of bounds errors for E_{num}. Regarding E_{pos}, the fact that the error in the people amount is so high prevents us from associating the predicted people with the real ones to calculate the error in their distances.

Logically, it is in these scenarios that the intention to use networks trained with the infrared channel is evident, making an almost exclusive use of it to process the images and make acceptable predictions.

Furthermore, there is a tendency to reduce the error (both in the number of people and in their positions) when comparing the subsequent cases between the two networks (increasing the amount of light in the images). So, the network using four channels CNN_4 does produce better predictions at a general level.

In order to confirm this, we also calculated the relative error in the number of people E_{num} changing the attenuation factor ζ_{RGB}. This calculation involves a larger testing set S_2^{ts}, composed of more number of cases (in addition to the cases of S^{ts}). The comparison between both networks can be seen in Fig. 5, showing a clear advantage of the CNN_4 over the CNN_3, that is wider for dark images but also better for all the range of the studied visible attenuation factor.

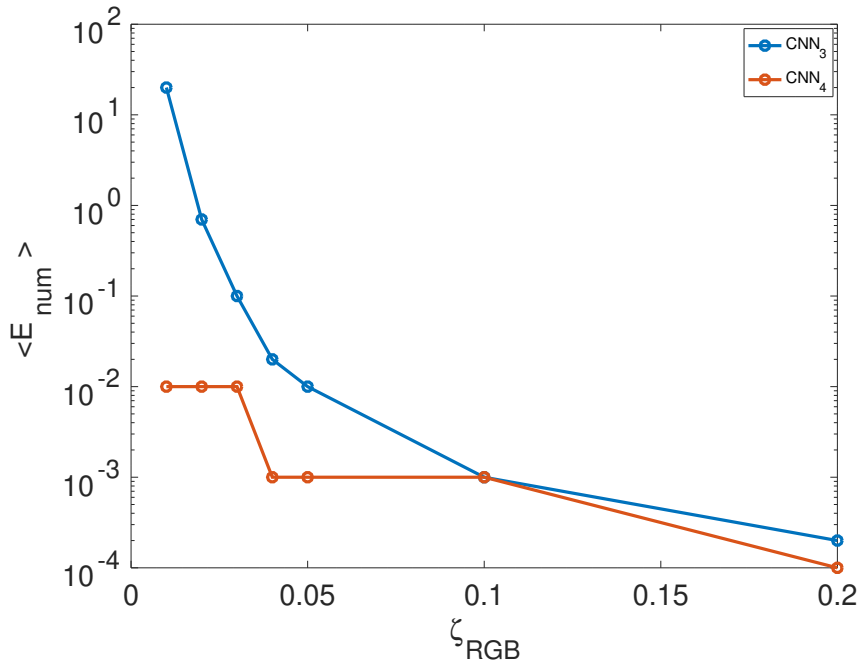

Figure 5: E_{num} mean for each case of the larger testing set S_2^{ts}, depending on ζ_{RGB}.

Finally, to illustrate the performance of the CNN_4, we chose a random image of the testing data set and displayed the predicted positions in Fig. 6, where it can be seen the precision of the neural network counting and positioning people from an image.

Figure 6: On the left, a random image of the testing set S^{ts} with $\zeta_{RGB} = 1$, grass background and 15 m altitude. On the right, the predicted positions of the people on top of the same image, using the CNN_4.

4 Conclusions

Two convolutional neural networks were created and trained, one taking 3 channels images (RGB) as input and the other 4 channel images (RGB + IR), which can count and provide the positions of people in aerial images (in our case, taken from a drone).

The performances of these networks were tested with zenith images of crowds at different altitudes and under different lighting conditions, in addition to combining other variables.

It can be clearly observed that in general, for the studied cases, the use of the fourth channel (IR) improves the precision in counting and positioning people in the images. This improvement is much more evident when the intensity of visible light is low.

Taking all this into account, the four channel network could be used to count people in open spaces, even in dark or changing lighting conditions such as at concerts or other outdoor cultural events.

References

[1] David Ryan, Simon Denman, Clinton Fookes, and Sridha Sridharan. Crowd counting using multiple local features. *Digital Image Computing : Techniques and Applications (DICTA)*, pages 81–88, 2009.

[2] Ya-Li Hou and Grantham K. H. Pang. People counting and human detection in a challenging situation. *IEEE Transactions On Systems, Man, And Cybernetics Part A:Systems And Humans*, 41(1):24–33, 2011.

[3] Ke Chen, Chen Change Loy, Shaogang Gong, and Tao Xiang. Feature mining for localised crowd counting. *BMVC*, 2012.

[4] Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang. Cross-scene crowd counting via deep convolutional neural networks. *CVPR*, 2015.

[5] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. *CVPR*, 2016.

[6] Viresh Ranjan, Hieu Le, and Minh Hoai. Iterative crowd counting. *ECCV*, 2018.

[7] Lokesh Boominathan, Srinivas S S Kruthiventi, and R. Venkatesh Babu. Crowdnet: A deep convolutional network for dense crowd counting. *MM '16: Proceedings of the 24th ACM international conference on Multimedia*, pages 640–644, 2016.

[8] Daniel R. Parisi, Juan I. Giribet, Claudio D. Pose, and Ignacio A. Mas. Set-up of a method for people-counting using images from a uav. *Traffic and Granular Flow*, 2019.

[9] Imran Amin, A.J. Taylor, Faraz Junejo, Amin Al-Habaibeh, and Robert Parkin. Automated people-counting by using low-resolution infrared and visual cameras. *Measurement*, 41:589–599, 2008.

[10] Brian D. Ripley. Pattern recognition and neural networks. *Cambridge University Press*, 1996.

[11] Dirk Helbing, Illés Farkas, and Tamás Vicsek. Simulating dynamical features of escape panic. *Nature*, 407:487–490, 2000.

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *Medical Image Computing and Computer-Assisted Intervention*, 9351:234–241, 2015.

[13] NeuroSYS. Objects counting by estimating a density map with convolutional neural networks. `https://github.com/NeuroSYS-pl/objects_counting_dmap`.