Artificial Neural Networks Based Place Categorization

Andreas Kriegler[a,1], Wilfried Wöber[b,2], Mohamed Aburaia[b,3]

[a] *Mechatronics/Robotics, UAS Technikum Vienna, Austria*

[b] *Digital Manufacturing & Robotics, UAS Technikum Vienna, Austria*

[1]krieglerandreas@gmail.com [2]wilfried.woeber@technikum-wien.at
[3]mohamed.aburaia@technikum-wien.at

***Abstract*— Localization using images is a fundamental problem in computer vision for autonomous systems. The information gained enables a high-level understanding of the environment for the mobile robot, which is much more in line with a person's understanding of their surroundings. It is therefore a critical task for human-robot collaboration and enables robots to find places and identify objects applying this a priori information using semantic maps. Customarily, deep learning models such as convolutional neural networks can be trained for location classification from video streams. However, to achieve acceptable place classification results, previous trained neural nets must be adapted to the specific environment. This work tackles this adaptation of an existing system for place categorization including an extension in terms of possible places. The extension is based on classical machine learning models which were trained based on extracted features of a pre-trained neural network. The developed system outperforms the initial system by correctly classifying 90% of images, including places which were unknown to the neural network.**

***Keywords*— place categorization, convolutional neural networks, support vector machine, machine learning, mobile robotics**

I. Introduction

With rapid advancements in technology and science, the level of autonomy in mobile robotics is ever increasing [1, 2]. The exponential increase in computing power in graphics processing units (GPU) has paved the way for the field of deep learning (DL) to emerge [3, 4] – a new approach to solve statistical problems regarding autonomous agents or systems [5]. Convolutional Neural Networks (CNN), introduced in [6], are at the forefront of deep learning when large amounts of visual data need to be processed in a way that enables the overarching system to learn with this data [7, 8, 9, 10, 11, 12, 13]. This has made CNNs a natural addition to mobile robotics systems that are often equipped with multiple cameras to obtain images of their surroundings. The analysis of visual data using CNNs enables mobile robots to extract information of their environment. Based on this information, robots can then answer questions such as Where am I? [14], but also What is the place I am in like? [15]. Images or samples drawn from videos or datasets can be classified based on a known set of existing classes – in this case different semantic place labels such as *hallway* or *kitchen*. Combining place categorization with a map of the robot's environment yields semantic maps [16, 17]. This work focuses on the place categorization aspect, using techniques of supervised learning that shall now be discussed in greater detail.

II. Supervised Learning

Supervised learning is a general term used to describe several machine learning (ML) algorithms that learn a function which maps a given input to an output based on previously learned input-output pairs [18, chapt. 5]. A set of n training samples of the form $\{(\boldsymbol{x_1}, y_1), \ \dots \ , (\boldsymbol{x_n}, y_n)\}$ are given as input to train the algorithm, where $\boldsymbol{x_i}$ is the feature vector of the i–th sample and y_i is its label (or class). The algorithm then approximates a function $g: X \rightarrow Y$ where X is the input space and Y is the output space. These features are often expressed in vectorized form; hence one sample consists of a vector of features and the corresponding class label. All supervised learning tasks therefore require many labelled training examples, which are often difficult and expensive to obtain, requiring either intensive human labelling or accurate measurements. The goal is to obtain a trained model, which correctly determines the class labels for unseen instances, i.e. images that are new to the system. This requires the model to generalize properly [19] – the model must not be overfit [20] on the training samples and can classify new examples with high accuracy.

III. Convolutional Neural Networks

Artificial Neural Networks (ANN) are learning algorithms used to generate the aforementioned input-output mapping functions while making use of the vast computing power available with GPUs. ANNs approximate the mapping-function with a multitude of layers while employing a correction method to tune its weights during training. Figure 1 shows an example of the layout of an ANN with one hidden layer [21]. Every input value is assigned an input neuron $x_1 \dots x_k$ in the input layer forming the input vector $\boldsymbol{x} = (x_1, \ \dots \ , x_k)^T$. The values are then propagated through all the nodes and combined with a weighting vector $\boldsymbol{w}$ in a feed forward manner.

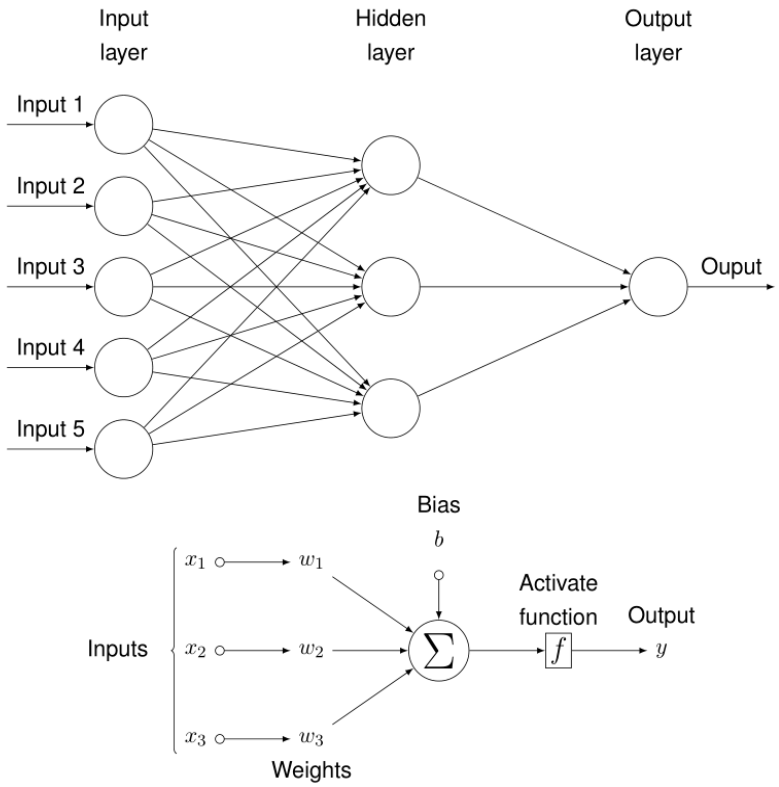

Figure 1: *General layout of an artificial neural network*. The basic layout (top) and the forward pass procedure of a single neuron (bottom) [21].

The activation function is classically the sigmoid function acting as a smooth threshold to map the input to the output following eq. (1).

$$f(\boldsymbol{x}) = sig(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{x}}} \tag{1}$$

Weights influence the steepness of the sigmoid and bias neurons translate the curve along the input axis. The output of one neuron therefore follows eq. (2).

$$y = sig(\boldsymbol{w} \cdot \boldsymbol{x} + b) \tag{2}$$

A convolutional neural network is a class of deep ANNs that has become very popular for extracting all kinds of features from images. The hidden layers in a CNN consist of further convolutional, pooling, normalization and fully connected layers. Figure 2 shows the basic layout of a CNN [22].

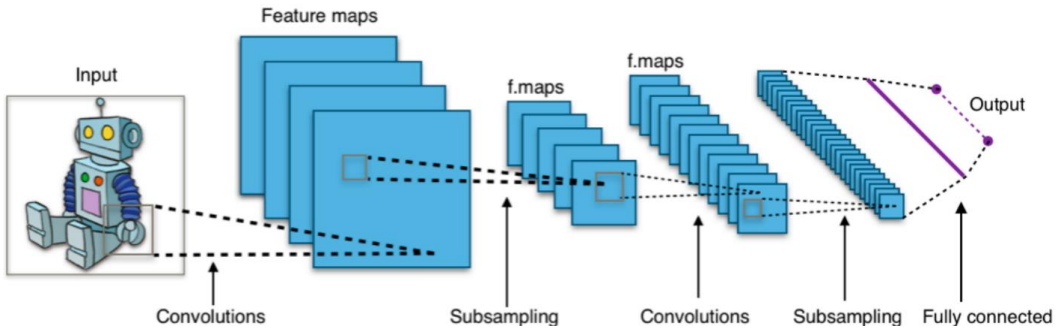

Figure 2: *CNN architecture*. The basic layout and connectivity of the layers in a convolutional neural network [22].

The convolutional layer's parameters consist of a set of learnable filters or kernels with a small receptive field that extends through the full depth of the input volume. During the forward pass, for every filter the dot product between the filter and the input is computed, resulting in a stack of two-dimensional activation maps that form the full output volume [23]. This stack is then passed through pooling and batch-normalization layers before being fed

into another convolutional layer. The end marks a fully connected layer that transforms the two-dimensional activation maps into a single value for every neuron in the output layer, making up the output vector with classification likelihood values [24].

A CNN was used in [15] for place categorization by using a set of images and place labels. Sünderhauf et al. [1 trained a CNN on the Places205 dataset [25] – a dataset featuring images from 205 different place categories including images from places such as *kitchen* and *auditorium* but also *desert* and *igloo*. From the original Places dataset with 476 classes, Places205 is a subset of all 205 classes which contain more than 5000 images per category. Since their network models were trained on such a vast number of different places the classification accuracy can be improved via fine-tuning on a dataset only containing a further subset of these 205 classes. The environment that the mobile robot was to be evaluated in, was expected to include e.g. a kitchen and a hallway, but not a desert, hence the system should be fine-tuned for this area of operations. On the other hand, the mobile robot is expected to encounter new places during its lifespan that might not be part of the 205 classes, therefore new classes should be easily included and trained on to allow classification amongst those as well. To this end, three different machine learning algorithms were trained using the output vector of the CNN as input. The architecture of the Places205-CNN is akin to the popular AlexNet-architecture [6] with a few minor modifications; mainly the last fully connected layer relates to 205 neurons, one for every place class. The last layer after that is a softmax layer [4] used to normalize the classification values of the 205 output neurons to sum up to 1; they can then be interpreted with probabilistic reasoning. While in the work of Sünderhauf et al. [15] this vector is seen as the final result, with the highest value corresponding to the predicted class, in this work this 205×1 vector was instead continuously extracted to create the design matrix $\boldsymbol{m_{205}} \in \mathbb{R}^{u \times 205}$ and corresponding target vector $\boldsymbol{t_{205}} \in \mathbb{R}^{u \times 1}$ where u is the number of analysed frames obtained from the CNN given a specific video as input. This way a new dataset was created, which was then split 75-25% into training and test data respectively. Afterwards, three machine learning models were trained using this dataset.

IV. Implemented Algorithms

The machine learning models were chosen as a way of finetuning the system because of their ease of implementation and to show that classical ML algorithms can work in combination with modern deep learning methods to great success, especially if the problem, such as classification among six classes, is not overly complex. This section discusses these implemented algorithms, namely the Naïve Bayes filter, multinomial logistic regression and a support vector machine.

A. Naïve Bayes Filter

Naïve Bayes filters (NBF) are probabilistic classifiers that are based on the Bayes' rule which relates the conditional probability $p(x|y)$ to its "inverse" $p(y|x)$. The probability density function for every feature for each class is first estimated using the Gaussian distribution. This gives a matrix of kernels $m_{205} \in \mathbb{R}^{K \times 205}$ where K is the number of classes that are expected to be encountered during operation of the mobile robot. The posteriors are then modelled according to the extended form of the Bayes' rule which can be observed in eq. (3). This step is done for every class C_k with $k \in \{1, \ \dots \ , K\}$.

$$\hat{p}(C_k \mid \boldsymbol{x}_1, \dots, \boldsymbol{x}_n) = \frac{\pi(C_k) \prod_{i=1}^{n} p(\boldsymbol{x}_i \mid C_k)}{\sum_{k=1}^{K} \pi(C_k) \prod_{i=1}^{n} p(\boldsymbol{x}_i \mid C_k)} \tag{3}$$

It gives the probability that the current image belongs to class C_k, given the input vector $\boldsymbol{x}$, based on the conditional probabilities $p(\boldsymbol{x}_i|C_k)$, where $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ are the random features of an observation and $\pi(C_k)$ is the prior point-probability that a class index is k. Finally, a decision rule is used that picks the most probable hypothesis; also known as the maximum a posteriori (MAP) rule. The classifier is therefore the function that assigns label $\hat{y} = k$ for some k according to eq. (4).

$$\hat{y} = \underset{k \in \{1,\dots,K\}}{argmax} \ \hat{p}(C_k \mid \boldsymbol{x}_1, \dots, \boldsymbol{x}_n) \tag{4}$$

B. Multinomial Logistic Regression

Multinomial logistic regression (MLR) is an extension of logistic regression for multiclass problems [26]. The log-odds of the probability of a sample belonging to a certain class is a linear combination of independent variables x. In multiclass logit the dependent variable y can take one of multiple discrete outcomes. If the model is parameterized by $\boldsymbol{\theta}$, with N being the number of drawn samples and N^{-1} therefore being a normalization factor, the model can be described with eq. (5).

$$\lim_{N\to+\infty} N^{-1} \sum_{i=1}^{N} \log p(y_i \mid \boldsymbol{x}_i ; \theta) =$$

$$\sum_{\boldsymbol{x}\in X}\sum_{y\in Y} p(\boldsymbol{x}, y) \left(-\log \frac{p(Y = y \mid X = \boldsymbol{x})}{p(Y = y \mid X = \boldsymbol{x}; \theta)} + \log\, p(y \mid \boldsymbol{x}) \right) =$$

$$-D_{KL}(Y \,||\, Y_\theta) - H(Y \mid X) \tag{5}$$

$H(Y \mid X)$ is the conditional entropy and quantifies the amount of information needed to describe the outcome of Y given that the value of X is known. $D_{KL}(Y \,||\, Y_{\boldsymbol{\theta}})$ is the Kullback-Leibler divergence, which measures the information loss using the learned model and quantifies this information loss when the distribution from parameterized model Y_θ is used instead of the ground truth distribution Y. By adjusting the parameters $\boldsymbol{\theta}$ to maximize the log-likelihood of the model (via iterative schemes such as the Newton-Raphson method), the KL divergence is minimized, thus the model which makes the least number of assumptions in its parameters is found; it is closest to the ground-truth.

C. Support Vector Machines

Support vector machines (SVM) try to classify data by separating the data points with a line in the case of two-dimensional data, or a $(p-1)$-dimensional hyperplane for the case of a p-dimensional feature vector. The Hessian normal distance from the group of data-points to the "maximum-margin hyperplane" is maximized by minimizing a loss function (the Hinge loss) that can be observed in eq. (6) [27].

$$\left[\frac{1}{n}\sum_{i=1}^{n} \max\left(0, 1 - y_i(\boldsymbol{w}\cdot\boldsymbol{x_i} - \mathrm{b})\right)\right] + \lambda\left|\left|\boldsymbol{w}\right|\right|^2 \tag{6}$$

λ determines the trade-off between increasing the margin-size and ensuring that $\boldsymbol{x_i}$ lies on the correct side of the margin with n being the number of data vectors. SVMs use the *kernel trick* to generate the nonlinear decision boundary. Kernel functions are similarity functions in the original input space that correspond to the dot products of the individual data points in some expanded feature space. In this expanded space the data points are more easily separated with some linear boundary. Figure 3 gives a visual explanation of the kernel trick [28].

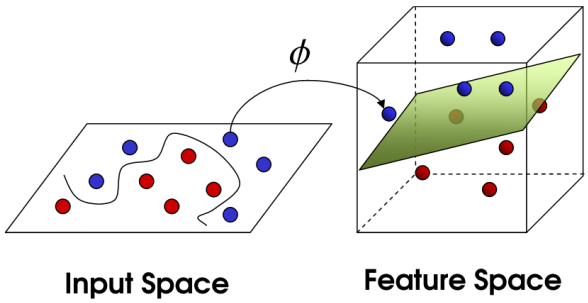

Figure 3: *Kernel trick*. While data points are not linearly separable in the input space, they are in a higher-dimensional feature space [28].

V. Results and Discussion

The system was evaluated on 5 of the possible 12 observable places on university campus: the digital factory resembling an industrial setting as the Places205 class *assembly_line*, a big lecture theater as *auditorium*, a *corridor*, a *kitchen* and an *office*. To show the capabilities of the system to extend the set of classes, data of an additional place was gathered not known to the CNN: a *door*. The videos were captured using the ROS wrapper for Intel Real Sense devices [29] and a BlasterX Senz3D [30] camera. The SVM and other models were tuned on the distributions of the features oftentimes showing correlations between objects that were not common in the original Places205 dataset. This is especially obvious for the frames captured in the digital factory, which resembles partially an office environment as well as an industrial production line, a combination of features that the CNN was not trained on. Table 1 shows the final classification results. Since the number of frames in every video recording were nearly equal, no further weighting had to be done. With the models having correctly labelled 100% of the samples from the *door* class, it is shown, that the classification can easily be extended to environments not part of the 205 places.

TABLE I: CLASSIFICATION RESULTS: TOP-1 CLASSIFICATION ACCURACY OF THE CNN, LIMITED TO 5 OUT OF 205 CLASSES AND THE ML MODELS TRAINED WITH THE INCLUSION OF THE *DOOR* CLASS.

Classification results (recall)				
environment	**CNN-5**	**MLR**	**SVM**	**NBF**
assembly_line	54%	**99%**	97%	92%
auditorium	62%	88%	**89%**	81%
corridor	**100%**	93%	93%	95%
door	-	**100%**	**100%**	**100%**
kitchen	**100%**	89%	87%	83%
office	75%	63%	73%	**79%**
total average	*78.2%*	*88.7%*	***90.2%***	*88.3%*

Figure 4 gives the normalized confusion matrix of the best performing model, the SVM – it shows per-class predictions against the ground truth.

	assembly_line	auditorium	corridor	door	kitchen	office
assembly_line	**0,97**	**0,00**	**0,00**	**0,00**	**0,00**	**0,03**
auditorium	**0,09**	**0,89**	**0,00**	**0,01**	**0,00**	**0,00**
corridor	**0,00**	**0,00**	**0,93**	**0,07**	**0,00**	**0,00**
door	**0,00**	**0,00**	**0,00**	**1,00**	**0,00**	**0,00**
kitchen	**0,13**	**0,00**	**0,00**	**0,00**	**0,87**	**0,00**
office	**0,03**	**0,00**	**0,00**	**0,23**	**0,01**	**0,73**

Figure 4: *SVM confusion matrix*. Normalized confusion matrix for the SVM-model. Rows show the true label vs. predicted labels in the columns.

The recording of the *door* and the *office* were suboptimal, in the sense that the *door* carried large similarities to a closet that was part of the *office*, meaning a fair number of visually similar images were part of two separate classes. This led to misclassifications with the ML-models, while the CNN was not influenced. Most errors therefore come from images from the *office* class being classified as elements of the *door* class (see Figure 4).

As a last experiment, an SVM was trained on a subset of the dataset featuring only three instead of the five classes: an *auditorium*, a *corridor* and a *kitchen*. For this problem the SVM boasted an accuracy of 99.54% showing the ambiguous *office* vs. *door* recordings were the primary cause for misclassifications in the five-class environment; although as the number of classes increases, the prediction accuracy is expected to decrease naturally. The method for classification was One-Against-One [31] – 13, 16 and 9 support vectors were used to create the binary classifiers amongst the three classes *auditorium*, a *corridor* and a *kitchen* respectively.

VI. CONCLUSION

This work introduced an extension for a place categorization system, based on a CNN, using machine learning. These models included a Naïve Bayes filter, a multinomial logistic regression model and a support vector machine. All three models yielded strong classification results between 88% and 99%. These three ML models are a perfect "light-weight" addition to the CNN for fine-tuning. The system, compared to an exclusively CNN-based approach, further makes no strong hardware- or complex software-demands, using a webcam and open-source software. The semantic information obtained is an important enabler of more advanced robotics tasks, especially human-robot collaboration, since humans describe places, rooms and goals not with coordinates but place labels. To this end, the system was successfully tested in an industrial- and office-environment, proving its usability for manufacturers and the commercial service industry alike.

REFERENCES

[1] R. Siegwart, I. R. Nourbakhsh and D. Scaramuzza. *Introduction to autonomous mobile robots*. Cambridge, Massachusetts: The MIT Press, 2011.

[2] Amazone. "BoniRob", *BoniRob – Forschung zur Feldrobotik* [Online]. Available: http://www.amazone.at/1857.asp

[3] R. D. Hof, *Deep Learning.* [Online]. Available: https://www.technologyreview.com/s/513696/deep-learning/

[4] L. Deng and D. Yu. "Deep learning: methods and applications." *Foundations and Trends in Signal Processing*, vol.7(3-4), pp. 197-387, 2014 Jun. 30.

[5] I. Goodfellow, Y. Bengio and A. Courville. *Deep learning*. Cambridge, Massachusetts: The MIT Press, 2016 Nov. 18.
[6] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard and L.D. Jackel. „Backpropagation applied to handwritten zip code recognition." *Neural computation*, vol. 1, pp. 541-551, Dec. 1989.
[7] A. Krizhevsky, I. Sutskever, G.E. Hinton. "Imagenet classification with deep convolutional neural networks." in *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, 2012, pp. 1097-1105.
[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich. "Going deeper with convolutions." in *Proc. IEEE International Conference on computer vision and pattern recognition (CVPR 2015)*, 2015, pp. 1-9.
[9] K. He, X. Zhang, S. Ren and J. Sun. "Deep residual learning for image recognition." in *Proc. IEEE International Conference on computer vision and pattern recognition (CVPR 2016)*, 2016, pp. 770-778.
[10] R. Girshick, J. Donahue, T. Darrell and J. Malik. "Rich feature hierarchies for accurate object detection and semantic segmentation." in *Proc. IEEE International Conference on comuter vision and pattern recognition (CVPR 2014)*, 2014, pp. 580-587.
[11] S. Ren, K. He, R. Girshick and J. Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." *Advances in neural information processing systems*, vol. 1, pp.91-99, 2015.
[12] J. Redmon and A. Farhadi. (2018, April). *Yolov3: An incremental improvement.* (1st edition). [Online]. Available: https://arxiv.org/abs/1804.02767
[13] B. Singh, H. Li, A. Sharma, and L.S. Davis. "R-FCN-3000 at 30fps: Decoupling Detection and Classification." in *Proc. IEEE International Conference on computer vision and pattern recognition (CVPR 2018)*, 2018, pp. 1081-1090.
[14] J. Borenstein, H.R. Everett and L. Feng. (1996, April). *Where am I? Sensors and Methods for Mobile Robot Positioning*. (1st edition). [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.444.5014&rep=rep1&type=pdf
[15] N. Sünderhauf, F. Dayoub, S. McMahon, B. Talbot, R. Schulz, P. Corke, G. Wyeth, B. Upcroft and M. Milford. "Place categorization and semantic mapping on a mobile robot." in *Proc. IEEE International Conference on Robotics and Automation (ICRA 2016)*, 2016, pp. 5729-5736.
[16] A. Pronobis and P. Jensfelt. "Large-scale semantic mapping and reasoning with heterogeneous modalities." in *Proc. IEEE International Conference on Robotics and Automation (ICRA 2012)*, 2012, pp. 3515 – 3522.
[17] S. Hemachandra, M. R. Walter, S. Tellex and S. Teller. „Learning spatial-semantic representations from natural language descriptions and scene classifications." in *Proc. IEEE International Conference on Robotics and Automation (ICRA 2014)*, 2014, pp. 2623-2630.
[18] S.J. Russell and P. Norvig. *Artificial intelligence: a modern approach*. Malaysia: Pearson Education Limited, 2016.
[19] T.M. Mitchell. *The need for biases in learning generalizations.* New Jersey: Department of Computer Science, Laboratory for Computer Science Research, Rutgers Univ., 1980.
[20] S. Geman, E. Bienenstock, R. Doursat. „Neural networks and the bias/variance dilemma." *Neural computation*, vol. 4(1), pp.1-58, 1992.
[21] G. Medina. "Diagram of an artificial neural network." [Online]. Available: https://tex.stackexchange.com/questions/132444/diagram-of-an-artificial-neural-network
[22] Aphex34. „typical CNN architecture." [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/6/63/Typical_cnn.png
[23] ujjwalkarn. "An Intuitive Explanation of Convolutional Neural Networks" [Online]. Available: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/
[24] F.-F. Li, J. Johnson and S. Yeung. CS231n. University Course, Topic: "Convolutional Neural Networks for Visual Recognition." University of Stanford. [Online]. Available: https://cs231n.github.io/convolutional-networks/
[25] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba and A. Oliva. "Learning Deep Features for Scene Recognition using Places Database." in *Adv. in Neural Information Processing Systems 27 (NIPS 2014)*, 2014, pp. 487-495.
[26] W.H. Greene. *Econometric analysis.* India: Pearson Education India, 2003.
[27] A. Ben-Hur, D. Horn, H.T. Siegelmann and V. Vapnik. „Support vector clustering." *Journal of machine learning research*, pp. 125-137, 2. Dec. 2001.
[28] dante. "Kernel trick explanation." [Online]. Available: https://datascience.stackexchange.com/questions/17536/kernel-trick-explanation
[29] Intel Corporation. „ROS Wrapper for Intel RealSense Devices." [Online]. Available: https://github.com/intel-ros/realsense
[30] Creative. "BlasterX Senz3D." [Online]. Available: https://us.creative.com/p/web-cameras/blasterx-senz3d
[31] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.