

Convolutional Neural Network for 3D Object Recognition using Volumetric representation

Xiaofan Xu^{*†‡}, Alireza Dehghani^{*}, David Corrigan[†], Sam Caulfield^{*†} and David Moloney^{*}

^{*}Movidiu Ltd. 1st Floor, O'Connell Br House, D'Olier St, Dublin 1, Ireland.

[†] Trinity College Dublin, College Green, Dublin 2, Ireland.

[‡]Email: xiaofan.xu@movidiu.com

Abstract—Following the success of Convolutional Neural Networks (CNNs) on object recognition using 2D images, they are extended in this paper to process 3D data. Nearly most of current systems require huge amount of computation for dealing with large amount of data. In this paper, an efficient 3D volumetric object representation, Volumetric Accelerator (VOLA), is presented which requires much less memory than the normal volumetric representations. Based on this, a few 3D digit datasets using 2D MNIST and 2D digit fonts with different rotations along the x, y, and z axis are introduced. Finally, we introduce a combination of multiple CNN models based on the famous LeNet model. The trained CNN models based on the generated dataset have achieved the average accuracy of 90.30% and 81.85% for 3D-MNIST and 3D-Fonts datasets, respectively. Furthermore, experimental results show that VOLA-based CNNs perform 1.5x faster than the original LeNet.

I. INTRODUCTION

Object Recognition (OR) is widely used in our daily life for inspection, registration, and manipulation [1]. Generally, OR is performed using colour-based segmentation methods or from grayscale images using Machine-Learning(ML)-based classification methods such as HoG [2]/SVM [3]. Although classical ML techniques are still being used to solve challenging OR problems, they don't work well because they ignore the structure and compositional nature of images. In contrast, Deep Learning (DL) approaches [4], [5], [6], [7], [8], the new generation of ML algorithms, are becoming ubiquitous. DL can model high-level features in data using multiple complex-structure processing layers or multiple non-linear transformations [9]. They are leading great performance in areas such as OR [10] and natural language processing [11]. With the recent advancements in DL, we are now able to solve some of the problems once considered impossible in computer vision, robotics, etc.

CNN techniques [4], as a member of DL family, addresses the gap in traditional ML techniques. CNNs not only perform classification, but they can also learn to extract features directly from raw images, which eliminates the need for manual feature extraction. They produce trainable filters and pooling operations on raw images, which result in multi-dimensional complex features. In addition, CNNs are invariant to pose, lighting, and surrounding clutter [12]. With the availability of large labelled datasets and powerful GPUs, the performance of CNN-based approaches has rapidly grown to over 95% accuracy (GoogLeNet [13], VGG [14], AlexNet [15], etc.).

However, CNNs are extremely computationally intensive requiring up to millions of multiplications per classification. Baidu has achieved the best result to date in the ImageNet classification challenge [16] with custom-built supercomputer called Minwa [17] which contains 72 powerful processors, 144 GPUs, 6.9TB of host memory, and 1.7TB of device memory. These high computational requirements for existing CNN models makes it impossible to deploy CNNs onto mobile devices without modification.

The success of CNNs on OR using 2D images [13], [14], [15], motivated us to extend the framework to process 3D images. Although architectures with volumetric convolutions have been successfully used in 3D video analysis [18], [19] where time acts as the third dimension, the nature of our data is different conceptually. Also, several works have extended CNNs to process 3D object using RGBD data [20], [21]. However, this approach does not contains the full geometric information in data and makes it difficult to combine information between different viewpoints [22]. In contrast, we propose a new CNN-based 3D object recognition approach that can recognise 3D objects from their complete or partial 3D volumetric representations.

There are works on object recognition using 3D points clouds from LiDAR and RGBD sensors. However, point clouds require features with their spatial neighbourhood queries. This can become intractable if there are a large number of points [22]. Unlike other existing 3D CNNs that use 3D point cloud data as training data [22] or RGBD to build 3D CNNs [20], [21], [23], [24], [25], CNNs in this study can be applied directly to recognize the raw 3D volumetric representation of objects. This is the main key characteristic of our approach. Moreover, we introduce new datasets that their structure is more effective for training CNNs than the existing ones. This is the second contribution of this work.

Finally, we introduce a light-weight volumetric representing method, called Volumetric Accelerator (VOLA), that offers substantial memory savings, up to 95% for realistic scenes. Accordingly, VOLA can reduce the computational complexity of CNNs drastically when it applied to 3D object recognition. VOLA makes the multiplications trivial since it represents objects by 1 or 0. It reduces the classification computational requirement incredibly and so it would be much easier to deploy CNN models onto embedded platforms.

The rest of the paper is outlined as follows: Section II

explains the several datasets have been created for this project along with the CNN models developed based on the proposed datasets. Performance evaluation of CNN models including their accuracy and classification run-time will be presented in Section III. Finally, conclusions and future works will be discussed in Section IV.

II. CNNs FOR 3D OBJECT CLASSIFICATION

In this section, a few different newly generated datasets are explained along with the generation procedure. VOLA 3D object representation is introduced afterwards. A few CNN models trained based on the datasets is presented finally.

A. 3D Dataset's Structure

The structure of a few different datasets that have generated to train and test the CNN models based on volumetric representation is explained in this section.

1) **3D-MNIST datasets**: This dataset has been built based on the original 2D-MNIST dataset. As Fig. 1 shows, each 2D-MNIST image is fed into a 3D model generator, e.g. Blender, to generate a 3D .stl model from the input digit with incorporation of no rotation. Then, different rotations along x, y, z axes, randomly selected between 0° to 360° with 1° rotation step, are applied to the 3D digits due to importance of rotations in 3D. This process generates 3 different 3D-MNIST datasets, each contains a total number of 59667 and 9957 training and test 3D digits, respectively. These datasets, called x-3D-MNIST, y-3D-MNIST, and z-3D-MNIST hereafter, are built in order to train the 3D CNN and compare the performance of which with the existing LeNet [4] trained on 2D images.

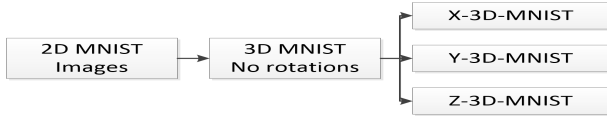


Fig. 1. Structure of the 3D-MNIST dataset.

2) **3D-Fonts dataset**: Since the real world 3D digits are fonts rather than the handwritten, a few 3D-Font datasets have been generated to recognize the 3D digits in the real world scenarios e.g. 3D digit candles. As Fig. 2 shows, the 3D-Font datasets have been created based on the free online fonts (1001freefonts) using Blender. The training and test datasets have been generated using 256 and 79 completely different fonts, respectively. Some examples of the 3D digits are shown in Fig. 3. Similar to 3D-MNIST datasets, different rotations with 10° step along the x, y, and z axes are applied to 3D-Fonts to create x-3D-Font, y-3D-Font, and z-3D-Font datasets, each containing a total number of 91440 and 27000 3D-Font digits respectively. To see the effect of combined rotations along x, y, and z, a dataset of compound x, y and z rotations, called xyz-3D-Font, is also created based on 25 and 5 different fonts for training and test. The rotation step in this dataset is limited to 40° for x, y, and z because smaller values would create a huge dataset with approximately similar performance. xyz-3D-Font dataset contains 144000 and 28800 3D-Font digits for training and test, respectively. As the final dataset, depth

images of different resolutions were captured based on the 3D-Font dataset to provide a means for memory foot-print comparison for VOLA representation.

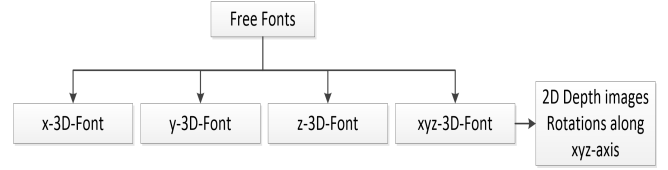


Fig. 2. Structure of 3D-Fonts dataset and 2D-depth images dataset.



Fig. 3. Sample 3D-font digits based on different fonts.

B. Volumetric Accelerator (VOLA)

The existing 3D CNNs use 3D point cloud data as training data [22] or RGB channel along with depth channel for building 3D CNNs [20], [21], [23], [24], [25]. However, VOLA representation is used in this study to represent the 3D objects to CNNs. This software library has been proposed by the authors for creating, manipulating, and visualizing volumetric data (It is under examination as a patent). More specifically, VOLA deals with regular volumetric grids, known as voxels, and stores only a single bit of information to represent each voxel. VOLA uses an octree data structure to store the voxels.

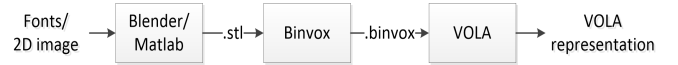


Fig. 4. Required steps for generating VOLA representation.

C. Data Format for CNNs

The 3D digits datasets are fed into CNNs for the training and test purpose. However, their format is not appropriate for the CNNs, which receive image inputs. So, the 3D .stl models should be converted into images first. In order to do this, the volumetric of 3D digits is represented and is then converted into images. We decided to use VOLA for representing 3D digits, rather than the other available 3D volumetric representation methods, because VOLA represents each voxel with a single bit which is more efficient. For example, Binvox requires 1 byte per voxel. However, in order to generate the VOLA representation a few steps are required, as shown in Fig. 4. Specifically, 3D digits with .stl format are converted into .binvox format first using Binvox, which reads a 3D model, rasterizes it into a 3D voxel grid and generates the resulting voxel file. Fig. 5 shows .binvox format of digits "2" and "0" with a few different rotations from z-3D-Font and xyz-3D-Font datasets, respectively.

As Fig. 6 shows, different resolution can be used for binvox, while three different resolutions were tested in the work. Obviously, most of the useful information of Digit 0 is missed in $8 \times 8 \times 8$ volumetric grid. $32 \times 32 \times 32$ volumetric grid provides the best visualizing output for binvox signal. However, it increases the computational requirement for training the CNNs.

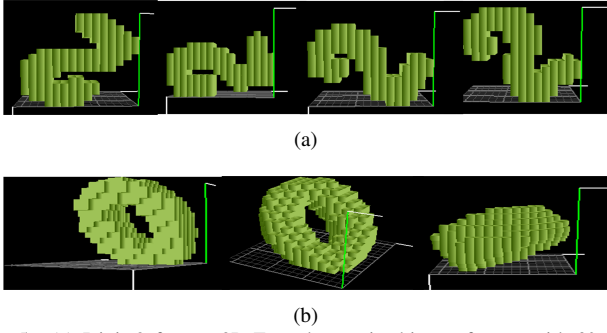


Fig. 5. (a) Digit 2 from z-3D-Font dataset in .binvox format with 0°, 40°, 80° and 100° rotations, respectively; (b) Digit 0 from xyz-3D-Font dataset in binvox format.

In this work, all 3D digits are based on $16 \times 16 \times 16$ voxels resolution because it compromises between the computation and visualization. With $16 \times 16 \times 16$ voxels representation, the binvox contains enough useful information to be recognized, while it moderates the CNN training computational costs.

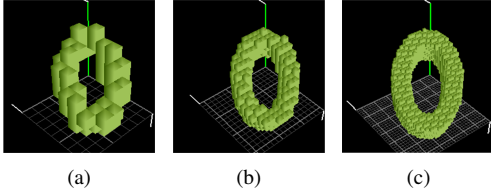


Fig. 6. 3D Same digit 0 with different resolution of binvox: (a) $8 \times 8 \times 8$ voxel; (b) $16 \times 16 \times 16$ voxel; (c) $32 \times 32 \times 32$ voxel.

After converting the 3D digits into .binvox, VOLA generation is applied to generate a single string of 1s and 0s from the input binvox file. VOLA starts the process from point $(0, 0, 0)$ (circle shown in Fig. 7(a)). Then, it moves from right to left along the x axis, bottom to top along the y axis and front to back along the z axis. As the final step, the VOLA single string is reshaped into the binary VOLA image (Fig. 7(b)), ready to be fed to CNNs. Each 16×16 yellow box on image (same size as the x and y dimensions in volumetric grid) represent 1 slice of the binvox in x-y plane along the z axis. It should be noticed that the yellow lines have been used only for visualization so they would not be existed on the images fed to CNNs.

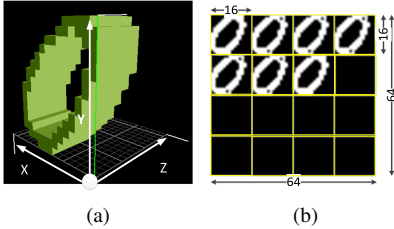


Fig. 7. Binvox to VOLA conversion process: (a) Binvox representation visualization; (b) VOLA representation.

D. CNN on 3D Digits Architecture

According to the proposed datasets in this work, a few LeNet-based CNN models have been designed using Caffe [26], the well-known CNN framework. Fig. 8 shows the CNN model layout used in this work. Unlike the traditional LeNet model that takes in the input of size 28×28 pixels, the proposed

CNN model takes in the input as VOLA image of size 64×64 pixels, based on the binvox of size $16 \times 16 \times 16$ 3D digit, without any resize. The first and second convolutional layers contain 20 and 50 kernels of size 5×5 , respectively. In the Pooling layers, max pooling of 2×2 is applied on each of the feature maps output of convolutional layers. Passing through multiple convolutional, max pooling, and fully connected layers, converts the input 2D VOLA image into a 10×1 vector, which contains the probability of belonging the input to each class and used later for 3D digit classification. All the trainable parameters in each layer are initialized randomly and trained using error back-propagation algorithm [4].

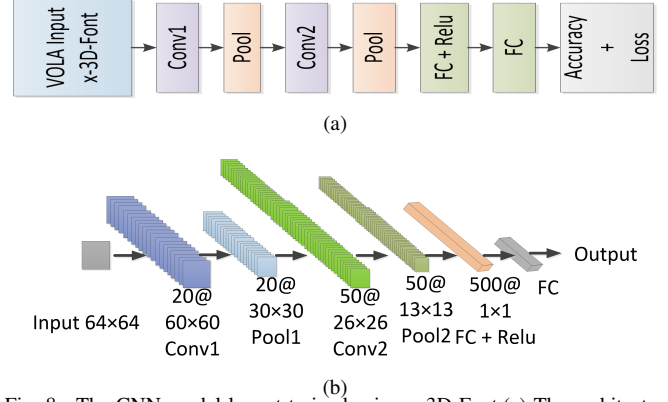


Fig. 8. The CNN model layout trained using x-3D-Font. (a) The architecture contains 2 convolution layers, 2 max pooling layers, 2 fully connected layer; (b) The architecture in details.

E. Combination of Multiple CNNs

Since the 3D digits in real world are associated with different combined rotations, datasets with different individual and combined rotations along x, y, z were presented in this section. Four CNN models with layout of Fig. 8 have been trained using the presented datasets (4 models for 3D-MNIST ones, 4 for 3D-Font datasets). The classification phase, then, is performed based on a Combination of Multiple Experts (CME) [27] approach which combines the output of each individual CNN models. According to Maximum Voting (MV) approach, the CNN model with maximum output will generate the final classification. We decided to use this approach to cover different rotations in real world. Thus, this combined multiple CNN models does select the optimal rotation somehow by MV approach. Experimental results demonstrate that the combined CNNs achieve good prediction on real 3D digit.

III. RESULT

The key findings of this research are portrayed in this section, with the most significant results being highlighted. Accuracies for each CNN model are listed. Furthermore, Classification time for all the CNN models are analysed.

A. Accuracy

In this work Caffe test tool has been used to calculate the accuracy of CNN models. Each trained CNN models is tested against its corresponding dataset. Table I shows the accuracy of CNN models trained based on 3D-MNIST and

3D-Font datasets. The average accuracy of 90.3% and 81.85% are achieved for 3D-MNIST nad 3D-Font trained CCNs. As can be seen, the lowest accuracy happens for datasets with z rotation due to the similarity of digits 6 and 9 when they are rotated 180° in z direction.

TABLE I
ACCURACY FOR 3D-MNIST & 3D-FONT CNNs.

3D-MNIST CNN	Accuracy	3D-font CNN	Accuracy
No rotation	93.75%	X-3D-Font	82.95%
X-3D-MNIST	91.80%	Y-3D-Font	84.70%
Y-3D-MNIST	92.32%	Z-3D-Font	77.33%
Z-3D-MNIST	83.31%	XYZ-3D-Font	82.43%
Average	90.30%	Average	81.85%

Since our proposed algorithm is an alternative to depth methods and also to provide a means for memory foot-print comparison for VOLA, we have trained a few LeNet models based on depth images of xyz-3D-Font dataset with different resolutions. Fig. 9 depicts the accuracy of these networks versus the resolution of depth images (directly related to memory foot-print). Meantime, Fig. 9 shows the accuracy of CNN trained based on xyz-3D-Font dataset with 64×64×1 (4096) bits per images, 82.43%, as a grey dot. Clearly, the depth-based CNN has got a lower accuracy than the 3D-Font when it uses the same memory foot-print. To achieve the same accuracy, depth-based CNN needs roughly 4x memory, which is significantly high. Also, depth-based CNN model can achieve higher accuracy of 90.56% with 32768 bits memory usage, which makes no sense for embedded systems.

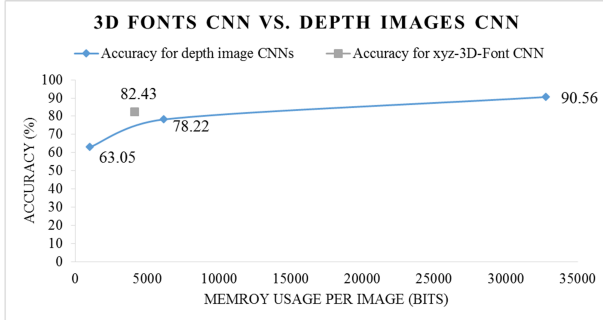


Fig. 9. Accuracy of CNNs based on different resolution depth images vs. memory foot-print.

B. Classification Run-time

In order to compare the classification run-time of CNN models, we have used the Caffe time tool in this work. The whole experiments has been based on Titan-X GPU. Fig. 10 shows the run-time comparison of different CNNs. Due to the similarity, only the run-time of each dataset family with rotation along x axes has been presented as well as the xyz-3D-Font CNN. As can be seen, the classification time for 3D digit datasets is 0.69x of the original 2D-MNIST dataset. It confirms the efficiency of our implementation of CNNs based on VOLA representation. Furthermore, the run-time of different layers of CNN model trained using xyz-3D-Font have been visualized in Fig. 11. Convolutional layers require the most of computational time as there are millions of operations involved in convolutional layers including multiplications and

additions. VOLA representation highly helps to reduce the classification time as multiple 1 or 0 is trivial.

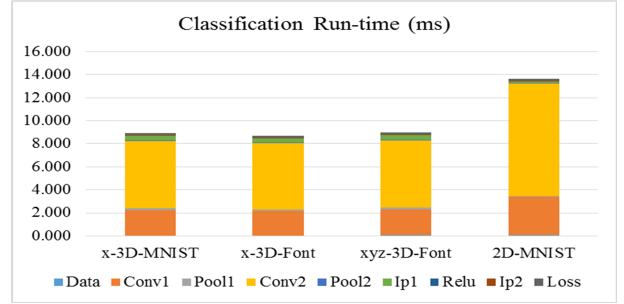


Fig. 10. Classification run-time for CNN models.

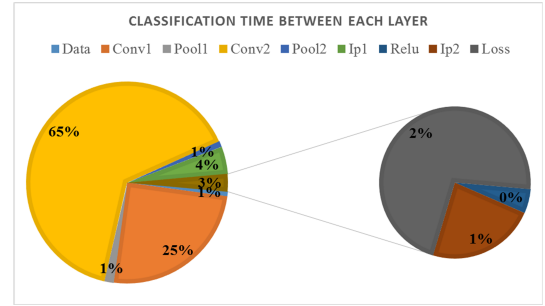


Fig. 11. Run-time of different layers of xyz-3D-Font CNN model.

IV. CONCLUSION

In this paper, we have proposed a 3D object classification approach based on volumetric representation using convolutional neural networks. Firstly, the 2D MNIST dataset as well as a bunch of different online freely available fonts were used in order to create a few 3D digits datasets for training an testing of CNN models. The datasets are generated using Blender and Matlab and cover rotations in x, y, and z directions. A new volumetric representation, called VOLA, has been introduced in order to save the memory footprint and the computational requirement. On this basis, all the 3D digits of datasets have been converted into VOLA representation in order to be fed into the training and testing of CNNs. In order to provide a means for memory foot-print comparison for VOLA representation, depth images of different resolutions were captured based on the 3D-Font dataset. The CNN structure used in this work is based on the famous LeNet model. The trained CNN models based on the generated dataset have achieved the average accuracy of 90.30% and 81.85% for 3D-MNIST and 3D-FonTs datasets, respectively. Furthermore, experimental results show that VOLA-based CNNs perform 1.5x faster than the original LeNet. . In order to evaluate the system against the real world 3D digits, a real world 3D digit dataset will be generated in the future. These 3D digits will be captured using devices such as Kinect.

ACKNOWLEDGMENT

This work has been partially supported by Project Eyes of Things (EoT) Grant n. 643924 from the European Union's Horizon 2020 Research and Innovation Program.

REFERENCES

- [1] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2. Ieee, 1999, pp. 1150–1157.
- [2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.
- [3] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.
- [4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [5] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
- [6] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [7] Y. Bengio, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.
- [8] Y. Bengio, Y. LeCun *et al.*, "Scaling learning algorithms towards ai," *Large-scale kernel machines*, vol. 34, no. 5, 2007.
- [9] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1798–1828, 2013.
- [10] M. A. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. LeCun, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 2007, pp. 1–8.
- [11] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 160–167.
- [12] Y. LeCun, F. J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 2. IEEE, 2004, pp. II–97.
- [13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.
- [14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [17] R. Wu, S. Yan, Y. Shan, Q. Dang, and G. Sun, "Deep image: Scaling up image recognition," *arXiv preprint arXiv:1501.02876*, vol. 22, p. 388, 2015.
- [18] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 1, pp. 221–231, 2013.
- [19] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.
- [20] R. Socher, B. Huval, B. Bath, C. D. Manning, and A. Y. Ng, "Convolutional-recursive deep learning for 3d object classification," in *Advances in Neural Information Processing Systems*, 2012, pp. 665–673.
- [21] L. A. Alexandre, "3d object recognition using convolutional neural networks with transfer learning between input channels," in *Intelligent Autonomous Systems 13*. Springer, 2016, pp. 889–898.
- [22] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 922–928.
- [23] S. Song and J. Xiao, "Deep sliding shapes for amodal 3d object detection in rgb-d images," *arXiv preprint arXiv:1511.02300*, 2015.
- [24] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research*, vol. 34, no. 4–5, pp. 705–724, 2015.
- [25] N. Höft, H. Schulz, and S. Behnke, "Fast semantic segmentation of rgb-d scenes with gpu-accelerated deep neural networks," in *KI 2014: Advances in Artificial Intelligence*. Springer, 2014, pp. 80–85.
- [26] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the ACM International Conference on Multimedia*. ACM, 2014, pp. 675–678.
- [27] L. Xu, A. Krzyżak, and C. Y. Suen, "Methods of combining multiple classifiers and their applications to handwriting recognition," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 22, no. 3, pp. 418–435, 1992.