ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10448146

ROBUSTNESS EVALUATION OF MACHINE LEARNING MODELS FOR ROBOT ARM ACTION RECOGNITION IN NOISY ENVIRONMENTS

Elaheh Motamedi [†], *Kian Behzad* [†], *Rojin Zandi* [†], *Hojjat Salehinejad* [‡⋆], *Senior Member, IEEE, and Milad Siami* [†], *Senior Member, IEEE*

[†]Department of Electrical & Computer Engineering, Northeastern University, Boston, MA, USA
[‡]Kern Center for the Science of Health Care Delivery, Mayo Clinic, Rochester, MN, USA
[⋆]Department of Artificial Intelligence and Informatics, Mayo Clinic, Rochester, MN, USA

ABSTRACT

In the realm of robot action recognition, identifying distinct but spatially proximate arm movements using vision systems in noisy environments poses a significant challenge. This paper studies robot arm action recognition in noisy environments using machine learning techniques. Specifically, a vision system is used to track the robot's movements followed by a deep learning model to extract the arm's key points. Through a comparative analysis of machine learning methods, the effectiveness and robustness of this model are assessed in noisy environments. A case study was conducted using the Tic-Tac-Toe game in a 3-by-3 grid environment, where the focus is to accurately identify the actions of the arms in selecting specific locations within this constrained environment. Experimental results show that our approach can achieve precise key point detection and action classification despite the addition of noise and uncertainties to the dataset.

Index Terms— Franka Emika robot arm, deep learning, key point extraction, noisy environment, robot arm action recognition.

1. INTRODUCTION

Robotic systems are widely used in diverse sectors such as healthcare, manufacturing, and automation [1, 2]. Central to the performance and utility of these systems is the accurate determination of the robot's spatial orientation, commonly known as pose estimation, in an uncertain environment. Typically, pose estimation involves using regression models to detect the key points of a robot, which are the joints that make up its skeleton. Accurate pose estimation is crucial for various tasks, ranging from object manipulation to navigation and interaction with the environment [1, 3, 4].

With the advancement of machine learning, significant progress has been made in the field of robot pose detection [5–8]. As an example, convolutional neural networks (CNNs) have been widely developed to address the pose estimation problem of robot arms in ideal environment [9–13]. However, such solutions often overlook the challenges of noisy and real-world environments, leading to models that excel theoretically but struggle in everyday scenarios.

In this paper, a pretrained ResNet-50 [14] is utilized as a standard tool for pose recognition. The output of this model is a time series of pose locations prone to noise. A CNN is proposed for robot arm action recognition from the noisy time series and its performance is compared with the state-of-the-art models such as transformers [15] and Rocket [16]. The dataset for the experiments was collected using a Franka Emika [17] robot arm. This study makes a significant contribution to the field of robot action recognition by introducing a model for robot arm action recognition with minimal error margins in noisy environments. The insights gained from this study have the potential to inform the development of more reliable robotic systems for various applications in noisy environments.

2. METHOD

The proposed model leverages a pre-trained ResNet-50 [14] on ImageNet [18] for robot arm pose recognition followed by a CNN for robot arm action recognition in noisy environments, as depicted in Figure 1. Each step is described as follows.

2.1. Robot Arm Pose Recognition

To capture the movement of the robot arm, our initial step involves extracting precise arm poses using J cameras, where each camera j outputs a video stream $\mathbf{V}^{(j)}$ of length T with

This material is based upon work supported in part at Northeastern University by grants ONR N00014-21-1-2431, NSF 2121121, the U.S. Department of Homeland Security under Grant Award Number 22STESE00001-01-00, and by the Army Research Laboratory under Cooperative Agreement Number W911NF-22-2-0001. The views and conclusions contained in this document are solely those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security, the Army Research Office, or the U.S. Government.

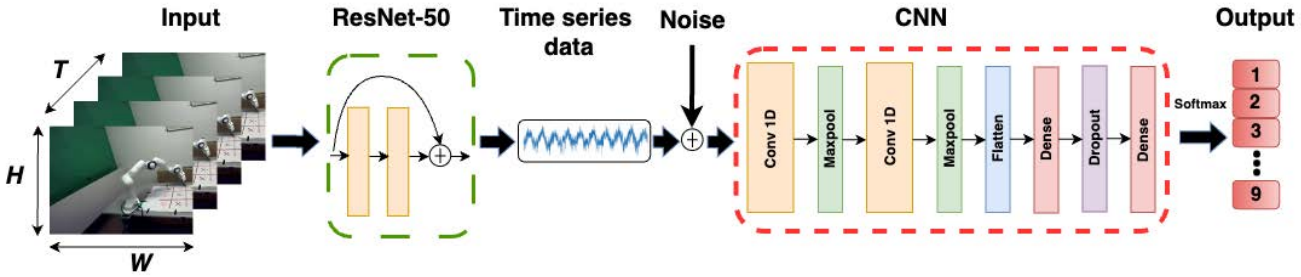

Fig. 1: Architecture of robot arm action recognition in noisy environments. Noise is applied after the estimation of the key points using ResNet-50 on the robot arm in the environment.

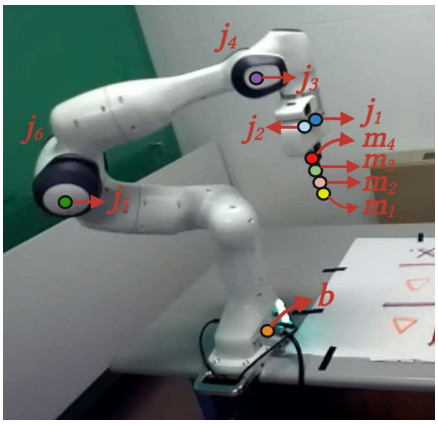

Fig. 2: Colored circles highlight detected key points, except for j_4 and j_6, which are not visible in this frame due to visibility constraints.

frame size $H \times W$. Each video stream is processed by a pre-trained ResNet-50 to estimate the robot arm poses in the X-Y plane over time as $\mathbf{A}^{(j)} \in \mathbb{R}^{K \times T}$, where K is the number of key points $(m_1, \ldots, m_4, j_1, \ldots, j_6, b)$ as in Figure 2.

A significant advantage of this approach is its effectiveness with a limited number of labeled frames, attributed to the pre-training of ResNet-50 on ImageNet [18].

2.2. Robot Arm Action Recognition

The output of ResNet-50 model for camera j, denoted as $\mathbf{A}^{(j)}$, represents the evolving trajectory of key points on a robot arm over time. It is assumed that a noisy environment can alternate $\mathbf{A}^{(j)}$ and affect the robot arm action recognition. To tackle this challenge, a CNN is proposed as depicted in Figure 1.

The proposed CNN model has a 1-D convolution layer with 32 filters where each filter is of size 3. After max-pooling with window size 2, another 1-D convolution layer is applied with 64 filters of kernel size 3. The resulted features after max-pooling with window size 2 are flattened and passed to two dense layers for action recognition. The number of possible target classes is 9, which represents different locations of

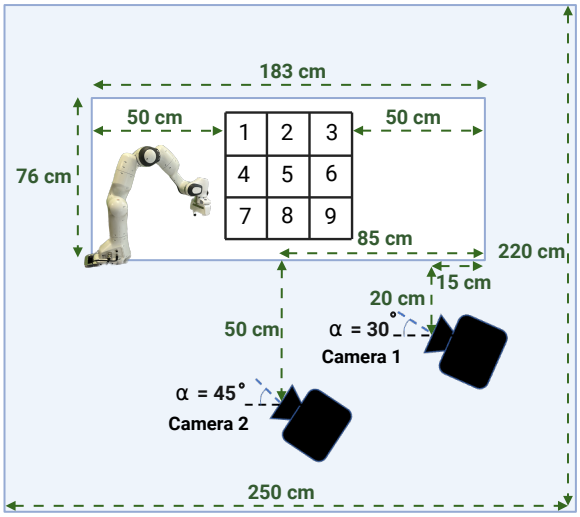

Fig. 3: Data collection using one robot arm and two cameras.

the Tic-Tac-Toe game as illustrated in Figure 3.

3. EXPERIMENTS

3.1. Data Collection

To evaluate our method, we utilized $J = 2$ cameras positioned at varying locations within our laboratory. These cameras recorded the movements of the robot arm on Tic-Tac-Toe board, as depicted in Figure 3.

Each recorded video has $T = 481$ frames captured over a 16-second duration, with an original image size of $H = 1280$ by $W = 720$. To ensure the diversity of our dataset, we gathered 50 samples for each of the nine action classes. Figure 4 shows some samples of the actions. We run experiments using a dataset consisting of 100 annotated frames.

3.2. Training Setup

The network was trained for up to 10^5 iterations with a batch size of 8, and early stopping was utilized. To select the opti-

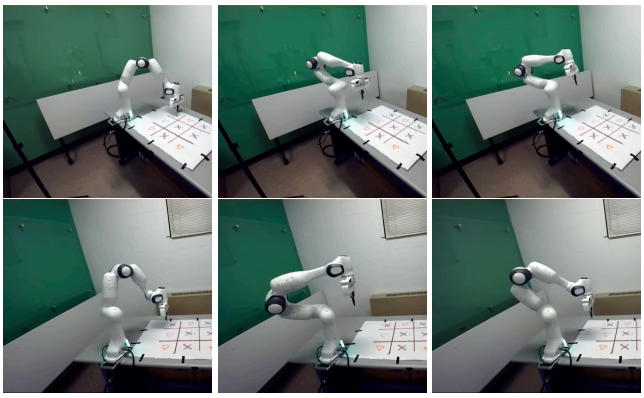

Fig. 4: Dataset samples: First row captured by camera one, second row by camera two as depicted in Figure 3.

mizer and learning rate, we conducted a grid search involving learning rates from the set $\{0.001, 0.01, 0.1\}$ and considered three optimization algorithms: SGD, RMSprop, and Adam. The categorical cross-entropy loss function along with the Adam optimizer [19] were configured with a learning rate of 0.001. The split percentages for all experiments are consistently set to 80%, 10%, and 10% for the training, validation, and the testing set, respectively.

3.3. Result Analysis

Our proposed model achieved high accuracy, with only a few misclassifications observed, between labels 8 and 5, as well as between labels 1 and 4. The proximity of these positions on the Tic-Tac-Toe board contributed to the confusion. In roughly 50 to 60 epochs, our model attains a testing accuracy of approximately 98%, with a validation accuracy of around 97.7%. To assess the classification model's performance, we employ two distinct models, transformer and Rocket, for classifying robot arm movements. We then compare their results with our proposed method, particularly evaluating their performance in a noisy environment.

3.3.1. Baseline Models

We utilized a transformer model including 4 attention heads, a point-wise feedforward network with a depth of 128, and 4 transformer encoder blocks, complemented by dense layers. The activation function used in the dense layers was ReLU. In addition, we incorporated 1-D global average pooling to compress the output generated by the transformer encoder module. Within approximately 80-90 epochs, our model achieves a testing accuracy of around 94.2%, a validation accuracy of approximately 93.7%. The Rocket model achieved a testing accuracy of around 95.6%, a validation accuracy of approximately 93.8%.

3.3.2. Evaluation in Noisy Environments

In this section, we perform a comparative analysis of three classification models by applying cut-out, salt and pepper, and Gaussian noise to time series data that represent the robot arm's trajectory. To understand the influence of noise, we designed and executed three experiments: the first employs noise augmentation only during training; the second tests the models on noisy data after training them on noise-free data; and the third both trains and tests the models on noisy data. The study compares these experiments with models trained without any noise. The main aim is to assess the model's ability to generalize to noisy data, simulating situations like camera malfunctions where frames are missing, resembling cut-out noise. Over the course of these experiments, we systematically increase the level of noise from 10% to 50%, enabling us to explore the model's resilience to escalating levels of noise and its subsequent ability to maintain accuracy and robustness. In the following, we provide detailed descriptions of three distinct types of noise applied to the dataset.

Cut-out Noise: In this approach, random rectangular regions within the time series are masked by setting their pixel values to zero. This technique encourages the model to understand context and adapt to missing or distorted information. The region size is 10, and masking is through a randomized probability selection process.

Salt and Pepper: Adding salt and pepper noise to time series data means inserting random spikes (salt) or drops (pepper) to mimic extreme values, making the data more resilient and ready for unexpected changes. We use parameters like density (how many data points get noise) and intensity (how strong the noise is). Density ranges from 10 to 50 in five steps, and intensity depends on the data's minimum (for pepper) or maximum (for salt) values. We decide to apply salt or pepper noise randomly to each data point based on probability.

Gaussian Noise: The critical hyperparameter in applying Gaussian noise is the noise level or standard deviation, which we have explored across a range of values, from 0.1% to 0.5%.

Average and standard deviation (STD) of accuracy for transformer, Rocket and CNN models are presented in Tables 1, 2 and 3 after 5-fold cross-validation. Additionally, Figure 5 provides a comprehensive analysis of how each type of noise impacts the performance of the three models, facilitating their comparison. In general, these models may display sensitivity to data noise, with variations depending on their architectures and the data's characteristics. Our study, conducted within the context of time series data and our specific application, illustrates how the sensitivity of these models to noise is influenced by their architectures and the unique features of the data. In summary, the CNN model emerges as the most promising performer, demonstrating exceptional capability in excelling even in noisy environments. On the other

Table 1: Performance of the **transformer-based model** with noisy data. Noise and accuracy results are in %.

Noise	Noise type	Train	Test	Train & Test
10	Cut-out	93.0±1.6	91.1±1.9	91.3±1.1
	Salt & Pepper	89.0±1.6	82.3±2.1	78.4±2.9
	Gaussian	90.6±2.7	95.0±1.1	96.0±2.1
20	Cut-out	93.3±2.1	88.8±2.7	91.0±1.7
	Salt & Pepper	82.6±1.8	75.9±3.5	74.3±4.4
	Gaussian	91.1±1.6	91.4±1.7	94.0±1.7
30	Cut-out	90.8±1.5	91.1±4.8	90.6±4.9
	Salt & Pepper	71.6±3.9	57.0±6.5	65.1±2.7
	Gaussian	87.5±2.2	97.7±2.6	93.0±2.3
40	Cut-out	88.8±4.1	89.2±2.1	85.7±4.3
	Salt & Pepper	70.2±2.6	55.1±4.6	44.7±3.9
	Gaussian	86.5±1.8	93.3±1.6	91.6±3.8
50	Cut-out	85.0±3.6	80.0±3.6	80.1±5.8
	Salt & Pepper	65.3±2.9	53.5±3.6	46.9±5.1
	Gaussian	86.8±4.0	90.5±1.1	86.4±1.5

Table 2: Performance of the **Rocket model** with noisy data. Noise and accuracy results are in %.

Noise	Noise type	Train	Test	Train & Test
10	Cut-out	79.2±6.2	58.4±4.4	95.8±3.7
	Salt & Pepper	73.5±4.3	34.4±3.8	78.3±6.3
	Gaussian	89.8±1.4	51.5±2.7	92.2±1.7
20	Cut-out	80.2±5.9	47.4±4.8	91.8±1.4
	Salt & Pepper	69.1±3.8	30.9±5.4	71.8±3.7
	Gaussian	83.2±4.6	48.2±5.3	93.1±2.6
30	Cut-out	73.8±4.3	35.5±3.7	94.1±4.6
	Salt & Pepper	62.4±5.2	30.1±4.9	58.4±2.7
	Gaussian	76.4±2.4	39.1±4.8	89.4±4.9
40	Cut-out	76.9±3.3	28.0±6.3	94.5±4.8
	Salt & Pepper	57.6±3.9	28.3±3.7	53.6±4.0
	Gaussian	69.4±3.3	32.7±3.9	89.1±4.1
50	Cut-out	60.7±4.8	21.3±5.2	92.1±5.3
	Salt & Pepper	47.1±3.4	26.4±4.5	48.8±5.7
	Gaussian	53.4±6.1	27.2±5.5	87.4±3.2

hand, Rocket exhibits the lowest level of effectiveness, with a notable drop in accuracy when exposed to noise. However, when noise is introduced into both training and test datasets, the Rocket model's performance remains consistent with that in a noise-free environment. It is worth noting that among all types of noise, salt and pepper noise has the most detrimental impact on accuracy across all models.

4. CONCLUSION

This study focuses on vision-based robot action recognition in noisy environments, where our model performs real-time, accurate 2-D pose estimation of a robot arm from video inputs. In contrast to studies focused on ideal environments, our research evaluates the performance of state-of-the-art neural networks like transformer and Rocket in noisy environments. Our findings show robustness of the model in robot arm action recognition in various noisy environments.

Table 3: Performance of the **CNN model** with noisy data. Noise and accuracy results are in %.

Noise	Noise type	Train	Test	Train & Test
10	Cut-out	98.1±1.2	98.0±1.2	98.0±1.6
	Salt & Pepper	95.5±1.8	91.1±1.8	95.6±1.9
	Gaussian	98.1±2.4	98.3±2.2	98.2±3.1
20	Cut-out	98.2±1.7	97.1±2.6	97.8±1.5
	Salt & Pepper	92.8±3.4	73.3±4.5	98.1±2.7
	Gaussian	97.2±1.7	97.6±3.5	97.4±1.9
30	Cut-out	98.0±1.1	98.2±1.7	97.1±1.4
	Salt & Pepper	92.3±2.4	52.5±1.7	97.9±4.9
	Gaussian	95.4±3.6	98.1±3.6	97.5±3.8
40	Cut-out	97.3±1.3	97.0±1.5	97.4±1.9
	Salt & Pepper	91.6±3.9	45.7±4.1	91.1±3.5
	Gaussian	95.7±4.8	98.0±4.7	97.0±4.1
50	Cut-out	97.0±3.4	97.7±2.9	96.8±2.2
	Salt & Pepper	88.0±4.1	31.3±2.9	89.6±3.3
	Gaussian	95.5±5.2	97.7±3.0	97.2±4.6

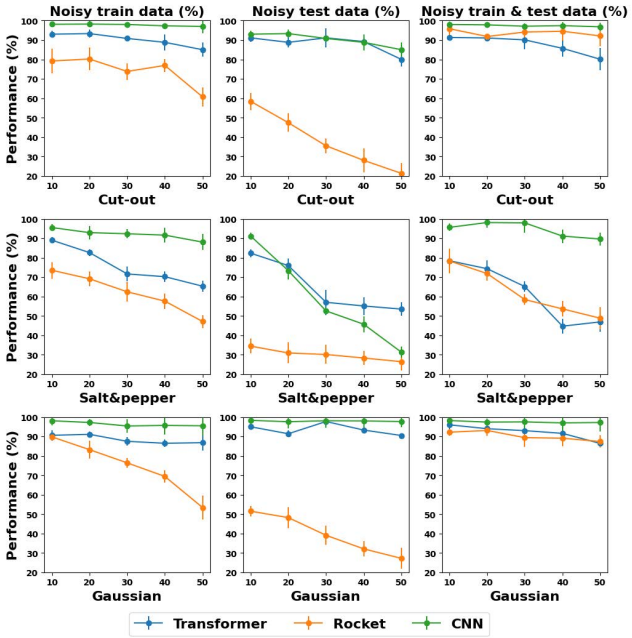

Fig. 5: Comparative analysis of transformer, Rocket, and CNN models in diverse noise scenarios during training, testing, and combined, in percentage. Top row: Cut-out noise; middle row: Salt and Pepper noise; bottom row: Gaussian noise.

5. REFERENCES

[1] Jingpei Lu, Ambareesh Jayakumari, Florian Richter, Yang Li, and Michael C Yip, "Super deep: A surgical perception framework for robotic tissue manipulation using deep learning for feature extraction," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4783–4789.

[2] Timothy E Lee, Jonathan Tremblay, Thang To, Jia Cheng, Terry Mosier, Oliver Kroemer, Dieter Fox, and Stan Birchfield, "Camera-to-robot pose estimation from a single image,"

in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9426–9432.

[3] Gregor Klančar, Luka Teslić, and Igor Škrjanc, "Mobile-robot pose estimation and environment mapping using an extended kalman filter," *International Journal of Systems Science*, vol. 45, no. 12, pp. 2603–2618, 2014.

[4] Chao Sun, Xing Wu, Jia Sun, Changyin Sun, and Lu Dong, "Robust pose estimation via hybrid point and twin line reprojection for rgb-d vision navigation," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–19, 2022.

[5] Christoph Heindl, Sebastian Zambal, Thomas Ponitz, Andreas Pichler, and Josef Scharinger, "3d robot pose estimation from 2d images," *arXiv preprint arXiv:1902.04987*, 2019.

[6] Yang Tian, Jiyao Zhang, Zekai Yin, and Hao Dong, "Robot structure prior guided temporal attention for camera-to-robot pose estimation from image sequence," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 8917–8926.

[7] Jingpei Lu, Florian Richter, and Michael C Yip, "Markerless camera-to-robot pose estimation via self-supervised sim-to-real transfer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21296–21306.

[8] Fan Zhou, Zijing Chi, Chungang Zhuang, and Han Ding, "3d pose estimation of robot arm with rgb images based on deep learning," in *Intelligent Robotics and Applications: 12th International Conference, ICIRA 2019, Shenyang, China, August 8–11, 2019, Proceedings, Part IV 12*. Springer, 2019, pp. 541–553.

[9] Thomas Gulde, Dennis Ludl, and Cristóbal Curio, "Ropose: Cnn-based 2d pose estimation of industrial robots," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2018, pp. 463–470.

[10] Thomas Gulde, Dennis Ludl, Johann Andrejtschik, Salma Thalji, and Cristóbal Curio, "Ropose-real: Real world dataset acquisition for data-driven industrial robot arm pose estimation," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4389–4395.

[11] Christoph Heindl, Sebastian Zambal, and Josef Scharinger, "Learning to predict robot keypoints using artificially generated images," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2019, pp. 1536–1539.

[12] Iago Richard Rodrigues, Marrone Dantas, Assis T de Oliveira Filho, Gibson Barbosa, Daniel Bezerra, Ricardo Souza, Maria Valéria Marquezini, Patricia Takako Endo, Judith Kelner, and Djamel Sadok, "A framework for robotic arm pose estimation and movement prediction based on deep and extreme learning models," *The Journal of Supercomputing*, pp. 1–30, 2022.

[13] Justinas Mišeikis, Inka Brijačak, Saeed Yahyanejad, Kyrre Glette, Ole Jakob Elle, and Jim Torresen, "Two-stage transfer learning for heterogeneous robot detection and 3d joint position estimation in a 2d camera image using cnn," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8883–8889.

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

[15] Sabeen Ahmed, Ian E Nielsen, Aakash Tripathi, Shamoon Siddiqui, Ravi P Ramachandran, and Ghulam Rasool, "Transformers in time-series analysis: A tutorial," *Circuits, Systems, and Signal Processing*, pp. 1–34, 2023.

[16] Angus Dempster, François Petitjean, and Geoffrey I Webb, "Rocket: exceptionally fast and accurate time series classification using random convolutional kernels," *Data Mining and Knowledge Discovery*, vol. 34, no. 5, pp. 1454–1495, 2020.

[17] Sami Haddadin, Sven Parusel, Lars Johannsmeier, Saskia Golz, Simon Gabl, Florian Walch, Mohamadreza Sabaghian, Christoph Jähne, Lukas Hausperger, and Simon Haddadin, "The franka emika robot: A reference platform for robotics research and education," *IEEE Robotics & Automation Magazine*, vol. 29, no. 2, pp. 46–64, 2022.

[18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[19] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.