# Object Classification with Tactile Sensing

Pranavi Boyalakuntla
Stanford University
pranavib@stanford.edu

Ben Cheng
Stanford University
bencheng@stanford.edu

Kavya Somasi
Stanford University
kavyas18@stanford.edu

## Abstract

*Inspired by the remarkable capabilities of human touch, researchers have been developing artificial tactile sensing systems to imbue machines with similar capabilities. These systems aim to enable machines to interact with the environment in a more intuitive and sophisticated manner, opening up new possibilities for various fields, including robotics, prosthetics, healthcare, and human-machine interfaces.*

*We use data collected from the DIGIT optical tactile senor to perform object classification of different US coins: Pennys, Nickels, Dimes, and Quarters. We generate both real and simulated data to train neural networks for performing this classification. We achieve up to a 96.81% in test accuracy in identifying US coins with our transfer learning approaches, and 25% training from a 2 layer CNN from scratch.*

## 1. Introduction

Tactile sensing is a fundamental aspect of human perception that allows us to interact with the physical world. By using our sense of touch, we can gather valuable information about objects, such as their texture, shape, hardness, and temperature. This ability to perceive and understand tactile stimuli is essential for tasks ranging from delicate manipulation to robust grasping and object recognition.

Tactile sensing is crucial for enhancing the perception and dexterity of machines, enabling them to gather detailed information about the objects they interact with. It plays a vital role in object classification, providing a valuable source of information that complements visual perception. While vision-based approaches rely primarily on visual appearance, tactile sensing offers a unique ability to capture fine-grained details about an object's physical properties. By exploring an object's surface through touch, tactile sensors can gather information about texture, hardness, temperature, and other tactile features that are often crucial for accurate object classification. This multi-modal approach combining vision and touch enables machines to overcome challenges posed by occlusions, variations in lighting conditions, and objects with similar visual appearances. Incorporating tactile information into object classification systems enhances their discriminative power, resulting in more reliable and precise identification of objects, especially in scenarios where visual cues alone may be ambiguous or insufficient. The integration of tactile sensing in object classification holds great promise for numerous applications, including robotics, industrial automation, assistive technologies, and human-machine interaction.Tactile sensing enables machines to perceive and understand the physical properties of objects they encounter, going beyond visual appearance.

Figure 1. Digit Tactile Sensor and Coins

Similarly, contact area estimation is a crucial aspect of tactile sensing with significant implications for diverse fields. In robotic applications, contact area estimation provides essential feedback for grip force control, allowing robots to exert appropriate forces during object manipulation. This information is particularly useful in delicate tasks, where excessive force could damage objects or insufficient force could lead to unstable grasps. In medical applications, contact area estimation is essential for prosthetics, where precise measurements of the pressure distribution between the prosthetic limb and the user's body help improve comfort, stability, and prevent pressure ulcers. Moreover, in material science, tactile sensing assists in analyzing the contact behavior between different materials, pro-

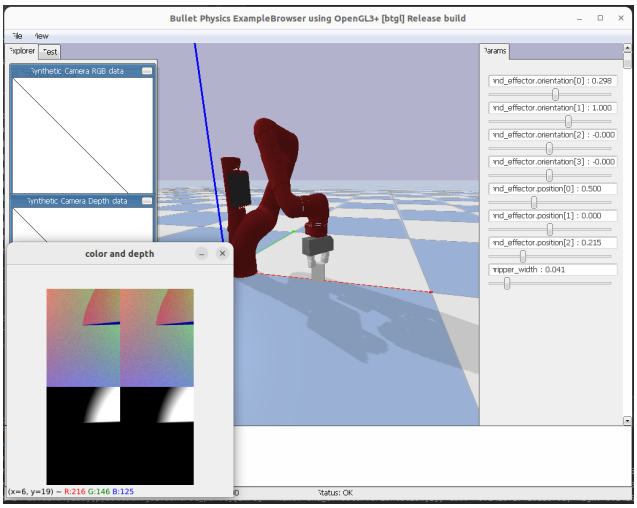


Figure 2. Image from running TACTO Example Code

viding insights into friction, adhesion, and wear, which aids in designing more efficient and reliable interfaces. Thus by accurately estimating contact area through tactile sensing, machines can optimize their interactions with the physical world, leading to improved performance, safety, and reliability in various domains.

The field of tactile sensing faces significant challenges in enabling machines to achieve manipulation performances comparable to those of humans. These challenges arise from the demanding sensing requirements and the need for novel actuation schemes. However, recent advancements have significantly lowered the barrier to entry with the availability of high-resolution, affordable, and compact tactile sensors, such as DIGIT. These sensors have propelled the development of powerful touch sensor simulators, like the TACTO framework, and machine learning frameworks like PyTouch, designed specifically for tactile sensing.

In our project, we aim to achieve a thorough comprehension of the effective integration of tactile sensor data and machine learning frameworks for the purpose of object detection and contact area estimation.

## 2. Related Work

DIGIT [1], an advanced vision-based tactile sensor, stands out among other sensors due to its compact form factor, robustness, and affordability. With dimensions of 20 mm in width, 27 mm in height, and 18 mm in depth, and weighing approximately 20g, DIGIT offers remarkable versatility. Notably, its distinguishing feature lies in the durable gel, which can be easily replaced with task-specific elastomers, enabling tailored tactile sensing for different applications.

TACTO [3], on the other hand, is a simulation library designed to mimic the deformation experienced by DIGIT sensors. It serves as a simulator for vision-based tactile sensors, effectively bridging the gap between physics simulation and the rendering engine. Leveraging various physics engines and ray tracing tools, TACTO offers high simulation quality and flexibility. The simulation process involves configuring the sensor, setting up camera, lights, and gel mesh in the rendering engine, and loading object data into the scene. By conducting necessary calculations for physics simulation, TACTO retrieves poses of each link and captures tactile imprints, ensuring synchronization between objects and sensors in the rendering engine.

Complementing TACTO, PyTouch [2] is an open-source library that processes raw sensor measurements, such as those obtained from TACTO, into high-level features for autonomous decision-making and information acquisition. PyTouch serves as a valuable tool for converting tactile data into actionable insights and facilitating machine learning techniques in tactile sensing research.

In a collaborative study [4] conducted by ETH and NTU, Convolutional Neural Networks (CNN) and Long-Short Term Memory (LSTM) neural network architectures were investigated for object classification using spatio-temporal tactile grasping data. The study compared the performance of BioTac SP and WTS-FT fingertip sensors and observed an accuracy improvement to approximately 94% for both sensor types. Additionally, the study proposed a method to generate more training examples from recorded data. The results emphasize the effectiveness of CNN and LSTM approaches for tactile-based object classification. Our approach shares similarities with theirs as we utilize data from physical sensors and TACTO simulations to train a CNN model for object detection.

In this project, our goal is to conduct a comparative analysis between TACTO, the simulator, and the hardware sensor, DIGIT, specifically in the context of image classification. By examining the performance of these tools, we aim to gain a deeper understanding of how open-source resources, such as TACTO and PyTouch, compare to the actual sensor information that would be available to a robot. Additionally, we intend to train a neural network model to classify images captured by the DIGIT sensor and estimate the contact area of objects in contact with the sensor. Through this research, we aim to enhance our knowledge of tactile sensing and its application in object classification and contact area estimation, paving the way for more advanced and intelligent robotic systems.

## 3. Methods

The main goal of this project is to understand how deformation sensor images can be used to conduct object classification using standard image processing methods. Image classification is can be done by applying Artificial Intelligence (AI) and Convolutional Neural Networks (CNN). We

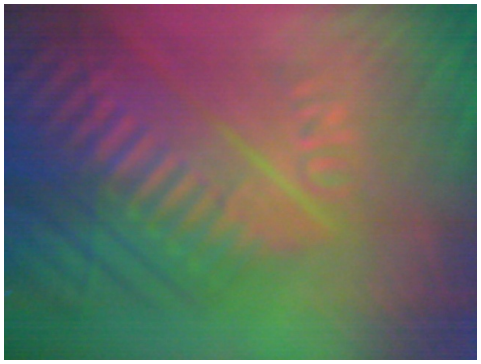

Figure 3. Real Penny Sensor Frame - Back

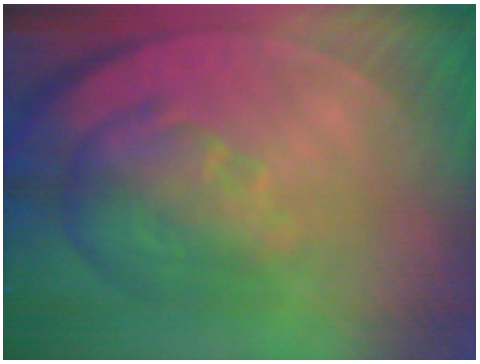

Figure 4. Real Penny Sensor Frame - Front

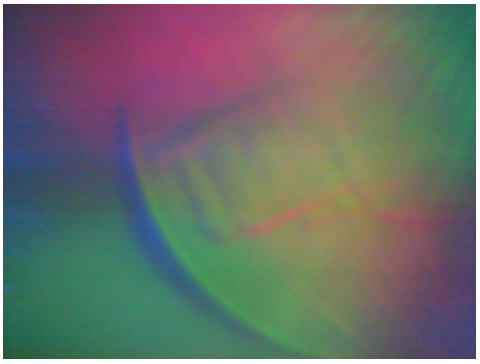

Figure 5. Real Nickel Sensor Frame - Back

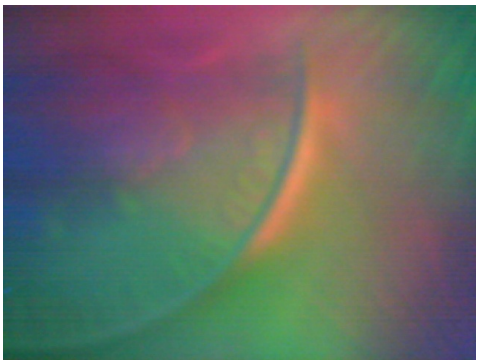

Figure 6. Real Nickel Sensor Frame - Front

aim to apply these same techniques to coin classification here.

There are two sub-goals of this project: understand how accurately we are able to do coin classification using real data produced by the DIGIT tactile sensor and seeing whether or not simulated data could potentially by used to train models for use with real data. Originally, we planned to train a model with simulated data and then test it on data from the TACTO sensor. The goal of this being to see how much the simulated sensor data can be used to train models for autonomy before having to collect any real data. However, since we were only able to find a model for the quarter coin, we decided to do the opposite by training on real data and testing on simulated data.

Before employing this methodology, we need to set up a simulated sensor and a way of getting similarly generated images from the actual tactile sensor that we bought. We purchased the DIGIT tactile sensor for approximately $350 to gather real sensor data.

In order to set up the DIGIT-interface with the real sensor and the TACTO simulator, a bare-metal Linux Machine is required due to rendering errors on other Unix-based operating systems and to avoid streaming issues that we've experienced with virtualization.

To collect the real data, we've configured the DIGIT touch sensor, which uses a OVM7692-RAAA CMOS image sensor, to produce QVGA resolution RGB images (320 x 240) at 30 fps. We press the sensor into the coins on both sides, and varied the rotation, translation, angle, and force placed. We modified the digit-interface examples to enable us to manually, with a keyboard input, to capture and save images.

Collecting simulated data ended up being more challenging than collecting real data for a number of reasons. To collect simulated data, we scoured the internet for models of digital coins that properly capture the geometry of the coins. It is difficult to find usable 3D models of nickels, dimes, pennies from the internet. Usable models that all the fine geometries you'd expect from real coins, surprisingly, need to be purchased. Thus, the only realistic digital coin model that we were able to acquire was a quarter. We scaled and placed the model of the quarter into the TACTO simulator, then captured images of the sensor colliding with the coin at different rotations and translations. The dynamics produced by the collision constraints within the simulator make it difficult to automate and manually capture images, so the simulated images we produced were all manually saved one by one.

We attempt to approximate the realistic scaling of this quarter with respect to the world; however, this is not an

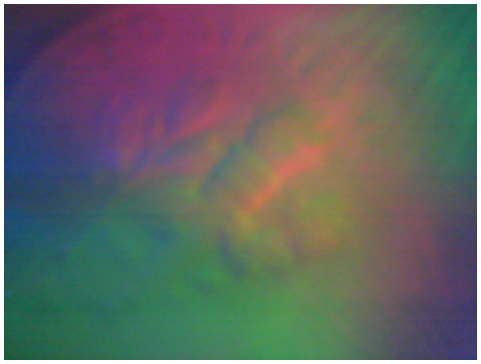

Figure 7. Real Dime Sensor Frame - Back

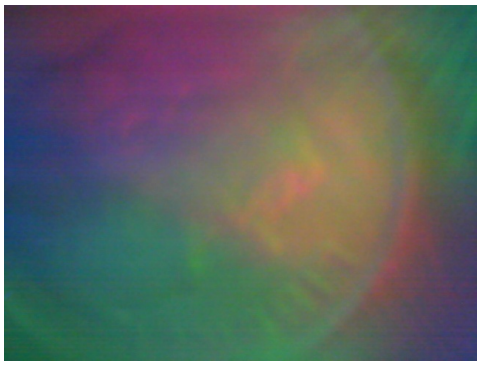

Figure 8. Real Dime Sensor Frame - Front

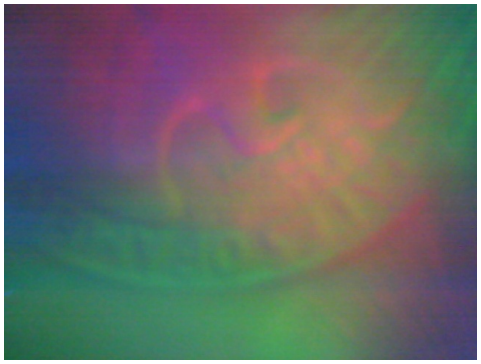

Figure 9. Real Quarter Sensor Frame - Back

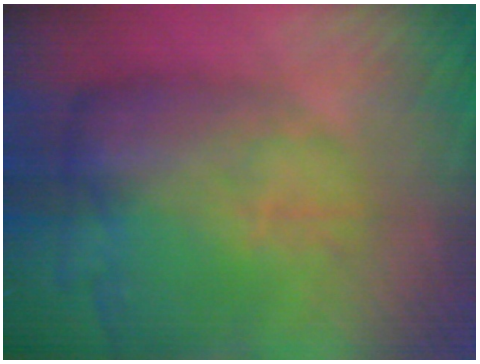

Figure 10. Real Quarter Sensor Frame - Front

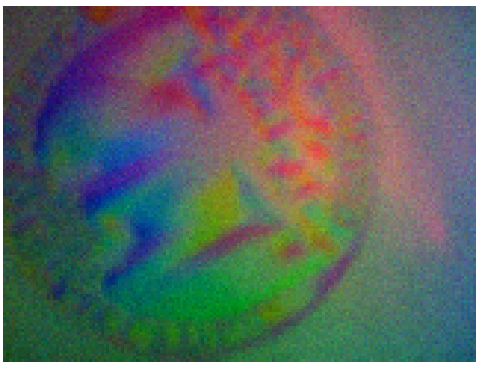

Figure 11. Simulated Quarter Sensor Frame from TACTO - Back

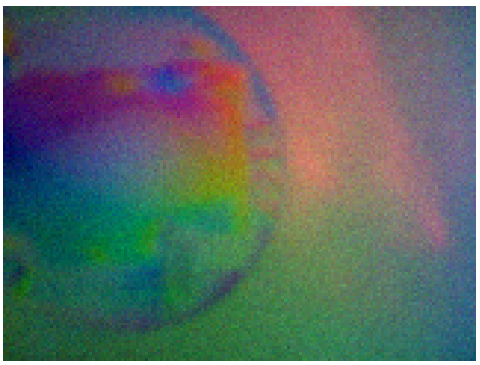

Figure 12. Simulated Quarter Sensor Frame from TACTO - Front

easy procedure since the TACTO interface is currently quite clunky. This introduces a sim-to-real gap. Furthermore, the captured image of the coin can vary quite largely based on how ”pressed into” the sensor it is within the simulator, and the simulator enables the user to ”press in” the coin far more than would be possible in real life, leading to unrealistic images, which makes it more difficult to size the coin correctly.

We’ve collected 5752 frames of real sensor readings and 50 frames of simulated readings in total.

To implement object classification, pretrained models using open-source tooling like PyTouch, PyTorch, and

Table 1. Dataset Image Distribution

| Number of Sensor Images | | |
|---|---|---|
| Coin | Real | Sim |
| Quarter | 1427 | 50 |
| Nickel | 1465 | 0 |
| Dime | 1257 | 0 |
| Penny | 1440 | 0 |
| Nothing | 173 | 0 |

Table 2. Final Pretrained Model Results

| Loss and Accuracy at Epoch 50 | | |
|---|---|---|
| Model | Train Loss | Test Accuracy |
| Resnet18 | 1.84e-3 | 94.61% |
| VGG19 | 4.18e-3 | 95.19% |
| VGG16 | 4.94e-3 | 95.95% |
| Densenet | 1.05e-3 | 96.81% |

OpenCV can be used alongside transfer learning. To do object classification, we use both a 2 layer CNN and perform transfer learning with pre-built CNNs. We compare those performances to what we'd see if we had just randomly guessed a class for all given images.

## 4. Results

The 2 layer CNN had an approximate accuracy of 25% with an approximate constant loss of 1.5. This loss did not decrease across epochs and the accuracy did not improve. The 2 layer CNN performed no better than random guessing. We suspect that this is because we only have 5752 images, and that it is insufficient amount of data for a 2 layer CNN to learn to distinguish between the classes of images.

We tested our data with prebuilt models utilizing transfer learning. There are several common models that are used for image classification including Resnet18, VGG16, DenseNet, and VGG19. We tested our data with these four models to see if one would perform better than others. We added a fully connected layer on top of these pretrained models using the PyTorch library to output one of 5 options: penny, nickel, dime, quarter, or nothing. They all performed very well and similarly across 50 epochs. DenseNet performed the best with an accuracy of identifying the correct type of coin as 96.81%. However, since all of the models perform similarly well, we believe any of these models would work. Transfer learning works significantly better than training our own CNN from scratch because these models have already accumulated the weights and features to differentiate different images even if these images have not been used to train the model in question.

We tested the models we generated on simulated data as opposed to training the models on simulated data and testing them on real collected data due to a lack of accurate models for any coins beyond quarters. The results from this test were that the model identified all of the quarters as pennies. We hypothesize that this is due to the fact that an important feature the model is looking for is the size of the coin in question. The accuracy from this model was consistently 0% since all of the test dataset was quarters and the model identified all of them as pennies.

Currently, the PyTouch library is suffering from a lack of documentation and examples. We attempted to implement contact area estimation using this library. However, there were significant bugs in the code that prevented this from running successfully. We have posted an issue on this on Facebook Research's public Github repository. Once this is resolved, this would be how contact area would be estimated.

## 5. Future Work

The most immediate improvement to this project is the collection of more data to improve accuracy of coin identification. Expanding the dataset to contain additional data from simulated coins such as nickels, dimes, quarters, as well as coins from other currencies, would make data set useful for more than one population. Further improvement of the simulated data by increasing the accuracy of the size and not allowing the user to press models into the digit sensor beyond a certain amount, as well as creating scripts to make it more easily to automate the process of collecting simulated readings. The PyTouch library as explained earlier, once debugged, could help understand what the perceived contact area of the coins are to the sensor, which would be great for doing friction and force estimation. Additionally, exploring the PyTouch library more, specifically, the slip detection feature could prove useful for the coin manipulation problem.

## 6. Conclusion

We have collected a dataset of tactile sensor images of different USD coins: quarters, nickels, dimes, and pennies, as well as, information from the sensor data of nothing being read from the tactile sensor. Both simulated data and the actual data from the sensor was collected using the DIGIT tactile sensor and simulated sensor data using the TACTO DIGIT sensor simulation interface. All of the work was done using Python. We have created object classifiers to distinguish between coins using vision based neural networks via PyTorch using both a 2 layer CNN and several modified pre-built transfer learning networks.

The results using transfer learning were incredibly promising, especially after the low accuracy rate using a Convolutional Neural Network (CNN) that we trained from scratch. The results from testing with simulated data could be improved. It is good that it recognized that all of the quarters were the same coin. However, it recognized them all as the wrong coin. The accuracy of this test would benefit from having a more accurate control over the size of the coin as this seems to be an important feature for detection.

Overall, tactile sensors can clearly be used for coin detection and detection of fine features. This will prove very useful in autonomy applications where recreating the sense of touch is crucial for further improvements.

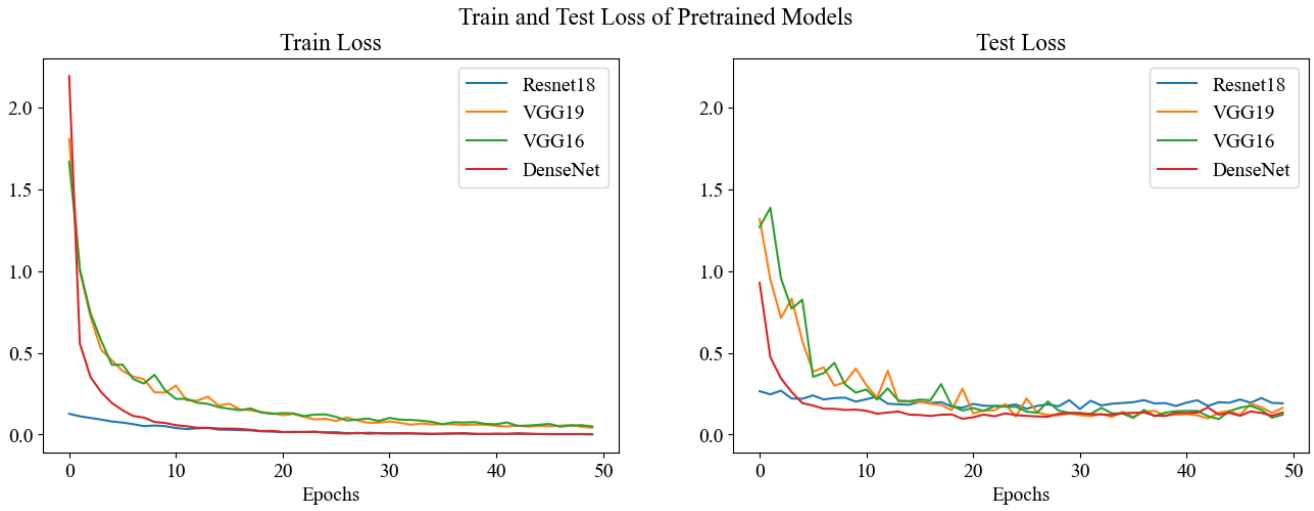


Figure 13. Loss Of All Pretrained Models

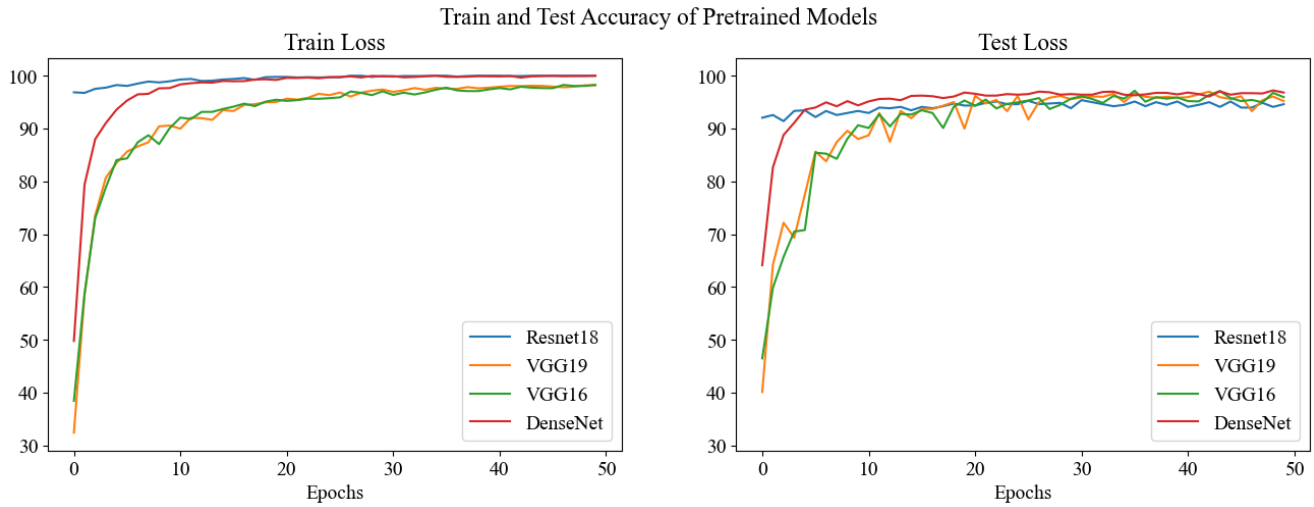


Figure 14. Accuracy Of All Pretrained Models

## References

[1] Mike Lambeta, Po-Wei Chou, Stephen Tian, Brian Yang, Benjamin Maloon, Victoria Rose Most, Dave Stroud, Raymond Santos, Ahmad Byagowi, Gregg Kammerer, Dinesh Jayaraman, and Roberto Calandra. DIGIT: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation. *IEEE Robotics and Automation Letters*, 5(3):3838–3845, jul 2020. 2

[2] Mike Lambeta, Huazhe Xu, Jingwei Xu, Po-Wei Chou, Shaoxiong Wang, Trevor Darrell, and Roberto Calandra. Pytouch: A machine learning library for touch processing, 2021. 2

[3] Shaoxiong Wang, Mike Lambeta, Po-Wei Chou, and Roberto Calandra. TACTO: A fast, flexible, and open-source simulator for high-resolution vision-based tactile sensors. *IEEE Robotics and Automation Letters*, 7(2):3930–3937, apr 2022. 2

[4] Nikesh Lama T.M. McGinnity Wolfgang Bottcher, Pedro Machado. Object recognition for robotics from tactile time series data utilising different neural network architectures. *IEEE International Joint Conference on Neuron Networks*, sep 2021. 2