Code used for success rate testing:

```
# Define a function to evaluate the success rate for a given model
def evaluate_success_rate(model, xtest, ytest):
    correct_count = [0, 0]  # Initialize counters for object one and object two

    # Make predictions for all samples in the test dataset
    predictions = model.predict(xtest)
    predicted_labels = np.argmax(predictions, axis=1)

    # Count correct predictions for each object
    for i in range(len(ytest)):
        if predicted_labels[i] == ytest[i]:
            correct_count[ytest[i]] += 1

    # Calculate success rate
    success_rate = (correct_count[0] + correct_count[1]) / 20

    return success_rate

# Repeat the evaluation process ten times for each technique
success_rates_ann = []
success_rates_cnn = []
success_rates_resnet = []

for _ in range(10):
    # Evaluate success rate for ANN
    success_rate_ann = evaluate_success_rate(model_ann, xtest_normalized, ytest)
    success_rates_ann.append(success_rate_ann)

    # Evaluate success rate for CNN
    success_rate_cnn = evaluate_success_rate(model_cnn, xtest, ytest)
    success_rates_cnn.append(success_rate_cnn)

    # Evaluate success rate for ResNet (or any other transfer learning model)
    success_rate_resnet = evaluate_success_rate(model_resnet, xtest, ytest)
    success_rates_resnet.append(success_rate_resnet)

# Calculate average success rates
avg_success_rate_ann = np.mean(success_rates_ann)
avg_success_rate_cnn = np.mean(success_rates_cnn)
avg_success_rate_resnet = np.mean(success_rates_resnet)

# Compare the success rates of each technique
print("Average Success Rate (ANN):", avg_success_rate_ann)
```

```
print("Average Success Rate (CNN):", avg_success_rate_cnn)
print("Average Success Rate (ResNet):", avg_success_rate_resnet)
```

result:

```
10/10 [==============================] - 1s 7ms/step
10/10 [==============================] - 0s 11ms/step
10/10 [==============================] - 1s 41ms/step
10/10 [==============================] - 0s 5ms/step
10/10 [==============================] - 0s 9ms/step
10/10 [==============================] - 0s 31ms/step
10/10 [==============================] - 0s 5ms/step
10/10 [==============================] - 0s 9ms/step
10/10 [==============================] - 0s 31ms/step
10/10 [==============================] - 0s 4ms/step
10/10 [==============================] - 0s 9ms/step
10/10 [==============================] - 0s 31ms/step
10/10 [==============================] - 0s 4ms/step
10/10 [==============================] - 0s 9ms/step
10/10 [==============================] - 0s 31ms/step
10/10 [==============================] - 0s 4ms/step
10/10 [==============================] - 0s 9ms/step
10/10 [==============================] - 0s 33ms/step
10/10 [==============================] - 0s 5ms/step
10/10 [==============================] - 0s 11ms/step
10/10 [==============================] - 0s 32ms/step
10/10 [==============================] - 0s 4ms/step
10/10 [==============================] - 0s 9ms/step
10/10 [==============================] - 0s 31ms/step
10/10 [==============================] - 0s 4ms/step
...
10/10 [==============================] - 0s 32ms/step
Average Success Rate (ANN): 14.0
Average Success Rate (CNN): 14.75
Average Success Rate (ResNet): 14.25
```

1. **Artificial Neural Network (ANN)**:
    - **Average Success Rate**: 70.0% (14 out of 20 photos)
2. **Convolutional Neural Network (CNN)**:
    - **Average Success Rate**: 73.75% (14.75 out of 20 photos)

3. **Residual Network (ResNet)**:
    - **Average Success Rate**: 71.25% (14.25 out of 20 photos)

Observations and Comparison:

- CNN achieved the highest average success rate among the three techniques, correctly classifying approximately 73.75% of the test photos.
- ResNet follows closely with an average success rate of 71.25%.
- ANN had the lowest average success rate, correctly classifying approximately 70.0% of the test photos.