National Technical University of Athens
Data Science and Machine Learning

Machine Learning
Amazon Bin Challenge

1st Salapas Konstantinos
NTUA
kossalapas@gmail.com
03400155

2nd Nanos Georgios
NTUA
nanosgiwrgos1997@gmail.com
03400144

3rd Lymperopoulos Theodoros
NTUA
theolyber@gmail.com
03400140

***Abstract*—The Amazon bucket dataset contains over 500.000 images, labeled with the number of objects in each image. In our project we test different models and find their accuracy to predict the correct number of objects in an image. After training many classical machine learning and deep learning models, we conclude that the best results are produced by the more complex deep learning models, and particularly the Residual Neural Networks. Still, the accuracy of the ResNet models is 50%, which shows the complexity of this task and the poor quality of the dataset. All of our other trained models could not achieve an accuracy higher than (25%), which reveals how much more powerful the CNNs are compared to classical approaches.**

***Index Terms*—computer vision, object counting, deep learning, residual neural networks.**

I. Introduction

In this project we train different models to predict the number of objects in an image. In computer vision this is the well-known "object counting" problem, and has many applications in microbiology (counting bacterial colonies in a Petri dish), surveillance (controlling traffic jam) and many other fields. Our task is to apply different models and object counting techniques to predict the number of products there exist in an Amazon bucket.

Amazon has achieved to distribute and deliver millions of products to more than 100 countries, with the help of logistics, robotics and computer vision tools. But due to the complexity of the task, false decisions may be made and many products are being misplaced between different buckets. It is an imperative need for the company to automate this process and produce better results, in order to improve its services. More specifically, in an Amazon Warehouse, the goal is to have robots with sensors and cameras attached to it, in order to pick up known objects with random poses out of a bucket using a robot end effector. In order to do so, the first thing the robot needs to learn is the number of items inside the box. This is the "Bin Image Count" problem that we address in our project.

The Bin Image Count problem is of course an application of the object counting problem in computer vision. For this problem, deep learning methods provide the state-of-the-art performance, which is a result that we also ended up in our work. After preprocessing the data and applying many of the classical machine learning models, such as Logistic Regression, Naive Bayes, SVMs and Random Forests, with other techniques that could possibly improve their performance (such as normalization, edge detection and the Canny algorithm for edge filter), we concluded that their accuracy was relatively low. The application of deep learning techniques improves significantly the results, even with a simple Multilayer Perceptron. We then study more complex DL models such as the Residual Neural Networks that produce the highest scores (accuracy 50%).

A great number of methods for object counting first addresses the problem of object detection. The methods first detect the desired object in all of its instances in an image, find their positions and then count the instances, returning the count. These methods are applied for example to self-driving cars, since the position of other cars is essential in order for the vehicle to drive its passengers safely. In our problem, the position of a product is irrelevant. It does not make a difference for our task, if product A is placed at the corner or the center of the bucket. We simply want our model to understand that it is a distinct object and count it. Thus, our models are

supervised and are simply trained to predict the correct number of objects in an image. With this approach we gain in simplicity, as long as we do have to answer to the more complex task of object positioning. But we lose in interpretability, since we are not sure what the model has learned to search for. The logic of our method is described below.

Considering the model that extracts the features of the images as a function, the input of it is an image of a bin that contains a certain number of items and the output is the number of items that identifies in the input image. Our aim for the model is to try to connect the label of an image with the number of objects it detects in the image itself.

Of course, the image dataset has to be preprocessed before training the models. Some images of the dataset do not have a corresponding json file with information about the image and are thus neglected. Then, each image is reshaped and converted to gray scale, with a resulting size of 120x120. After that the features of the new image are extracted and fed to a model, with its respective label. This label corresponds to the number of objects in that image.

II. Related Work

For the task of object counting, many different strategies have been followed, making the bibliography particularly rich. Thus, a strong need for grouping the different methods with similar characteristics grew. A set of taxonomies has been developed in order to classify the different approaches into similar categories. A research paper from K.Heinrich et al. [3] from the university of Dresden, sheds light on the different deep learning models that have been used, clustered by similar characteristics. Other methods that do not include deep learning models have been neglected the last years.

In terms of model architecture, there are three different approaches to the particular problem, as defined by the authors. First, the "One-Step ANNs" are systems that are trained as a whole, and only use one step processes, instead of multi-steps processing as we will see in other models below. Those methods typically use local and/or global features to determine the object density per pixel and might also make use of partial and whole images. Second, the "multiple steps that only involve ANN algorithms" are models that first extract partial image features and then use them for training a regression model. This technique is advantageous when objects overlap or differ in size. Some other ANN-based multi step approaches involve sequential models like LSTMs, or parallel architectures with multiple networks that process the data and then decide collectively or feed their results in a "deciding" model to make the final decision (two-step approach). The last category is " multiple ANNs as well as other methods" and includes most of the models that are developped the last years. This category contains models that exploit the preprocessing possibilities of ANNs and combine them with other image processing methods.

In terms of the different counting methods, the authors found a range of different outputs, which is dependent on the nature of the problem. In certain cases the methods make a rough prediction of the number of objects, without discovering their positions in the images. But most methods aim to predict the count of objects in an image (59/99), although without positioning the objects. A small minority of methods take positioning into account and determine a count per pixel. There are two different approaches for that: a) return a bounding box that forms a rectangular shape around the object or b) use centroid coordinates by pixel density.

Many other taxonomies have been developped such as the density of objects (overlapping - non overlapping), the background dynamic (static or dynamic), the number of objects and so on. Each parameter has its own different methods, developed at tackling the peculiarities of the corresponding problems. But we focus on the works of different other scientists tackling the object counting problem for the known datasets or, more precisely, the Bin Image Dataset.

Kaiming He et al. [5] introduced a residual learning framework that can be used to ease the training of deeper neural networks. The network layers are reformulated as learning residual functions with reference to the layer inputs. The deep residual network was tested with low error in the ImageNet Classification Dataset and provides considerable accuracy in increased depths. The residual network (Resnet) can be utilized for object detection and has been included as a classification model for this project.

Zihao Xu et al. [7] utilized Resnet to create models for object enumeration on the Amazon Bin Image Dataset. They compared classification and regression models based on Resnet for the prediction of the number of objects in each image and concluded that the classification models have higher accuracy.

Pablo Rodriquez Bertorello et al. [6] compare the predictions of different models for object enumeration on the Amazon Bin Image Dataset. The different models include logistic regression, classification trees, support vector machines and convolution neural networks (Resnet). The conclusion is that the convolution neural networks give much higher accuracy than the other classification models. A similar conclusion can be drawn from this project.

Fang Zhao et al. [2] introduced a Dynamic Conditional Convolution Network in which a specific convolutional kernel can be dynamically obtained from conditional inputs. The model is used for object counting on Amazon Bin Image Dataset, multimodal image classification, phrase grounding and identity based face generation. The prediction results are higher than in other projects.

Hisham Cholakkal et al. [4] utilized the ImageNet pre-

trained network to create a model for object counting and instance segmentation on the PASCAL VOC 2007 and COCO datasets. They provide predictions for the number of objects in the image and the number of instances of each object. Instance counting is not useful in datasets where there are no specific types of objects.

In summary, different architectures of convolutional neural networks have risen the last years, since it was evident that other machine learning algorithms could not conceive the complex relationships in the image data.

III. Dataset and Features

Amazon has published the Bin Image Dataset [1] which is available in a public S3 bucket. This contains in fact two datasets, the dataset of the images and the dataset of the corresponding json files that has useful information about the images. An image captures an amazon bin, that contains multiple products, such as books, toys, tools and many other products. The number of products of each bin corresponds to an order of a customer and thus varies from a bin to another.

As mentioned before, along with the images, the corresponding metadata -provided for each bin image- include information such as the object category identification, the quantity, the size of the corresponding object, the weight of the bin and so on. The size of bins varies as it depends on the size of objects in it. All this information is irrelevant. We only keep the useful information to us, which can be found in the field of "EXPECTED_QUANTITY" for a given json file. Any other information such as the type of the products contained in the image or the weight of the bucket is not related to the object detection task and, thus, is been neglected.

The dataset has proven to be very challenging. For start, in each image there are tapes wrapping the bins which are used for safety reasons. The lines of the tapes add noise to the data and make the task of edge detection and object identification even more difficult. Furthermore, in images containing many objects, some of them stand in front of the others, and thus only some of their parts are seen. For those reasons, in many cases the objects are unclear and their visibility is reduced. This makes the task of classification extremely hard, even for a human eye.

The dataset consists of over:

1) 500,000 images
2) 500,000 metadata

but there are few metadata files that do not correspond to a specific image. Those files are ignored.

As it is shown in Figure 1, the distribution of the number of objects in each bin image is heavily unbalanced. The majority of bin images contain two to five objects while the number of images that contain over 10 objects is significantly smaller. The method of undersampling would result in a dataset with insufficient data, while the method of oversampling would produce a significantly larger dataset, which would be difficult for a model to train on it. The preferred approach is to ignore classes with a low number of products and then apply undersampling to the rest of the classes. At last, since training the remaining images on the classifiers has heavy cost on the training time and computing power, a smaller percentage of the images is used, which is chosen at random.

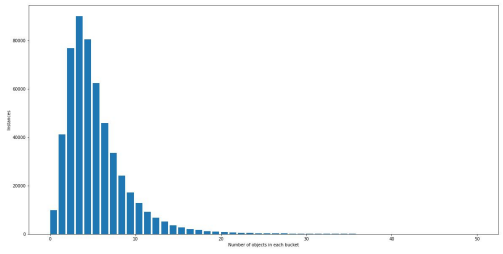


Figure 1: Distribution of quantity of objects in the images

After reading the image and metadata files, all the files that do not have a corresponding image are excluded and a subset of metadata and image files is selected for training. From each metadata file the "EXPECTED_OUTPUT" value is extracted, which represents the label of the image. Also, because of the different sizes of the images, they are reshaped and converted to gray scale with a size of 200x200. The features of the reshaped images are extracted as pixels, they are transformed using histograms of oriented gradients and then standardized.

IV. Methods

For our analysis, we experimented with different methods and models, and compared their accuracy in the validation set in the end. For start, we explored the prediction power of the simple machine learning classifiers in our dataset, namely, GaussianNB, Logistic Regression, Random Forests, SVMs and Multilayer Perceptrons, offered by scikit-learn. Then we applied more complex networks from the pytorch library. More particularly we used Residual Neural Networks.

To train the dataset we use classifiers provided by scikit-learn and convolutional neural networks and compare their results. As shown from previous work, the expected result is that the neural networks will perform better than the other classifiers.

GaussianNB

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. According to Bayes' theorem , the following relationship applies given class variable y and dependent feature vector x:

$$P(y|x_1, ..., x_n) = \frac{P(y)P(x_1, ..., x_n|y)}{P(x_1, ..., x_n)}$$

According to the naive conditional independence assumption , the above relationship is simplified to:

$$P(y|x_1, ..., x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1, ..., x_n)}$$

The probabilities $P(y)$ and $P(x_i|y)$ can be estimated by utilizing Maximum A Posteriori (MAP) estimation ; the former is the relative frequency of class y in the training set. In the case of GaussianNB classifier, probability $P(x_i|y)$ follows a Gaussian Distribution. The classifier is faster than other more complicated methods, but in this specific problem it is expected that the naive approach will result in poor accuracy.

LOGISTICREGRESSION

Logistic Regression is a linear model used for classification, in which the probabilities describing the possible outcomes of the dependent variable are modeled using a logistic function. The logistic function is described by the following equation:

$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}}$$

This model is used to predict the relationship between the dependent class variable and the independent feature variables. It is a fast method, but its disadvantage is that it depends on the linear relationship between dependent and independent variables and it works best on problems where there are two classes for the dependent variable. However, it can be performed in problems where there are multiple classes.

RANDOM FOREST

Decision Trees are a non-parametric supervised learning method used for classification that can create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. The decision trees classifier provides a model that is usually simple to interpret and performs well. The Random Forest classifier combines a number of decision trees on various subsets of the dataset. If the trees are diverse enough, the variance of the model is reduced and the problem of overfitting is prevented. However, the Random Forest classifier inherits a specific problem from decision trees; if the data classes are unbalanced then the model is biased to the dominant classes.

SUPPORT VECTOR MACHINES

Support vector machines (SVMs) are a set of supervised learning methods that can be used for classification. The method that SVMs follow is to map the data to a high-dimensional feature space so that the data points can be categorized. The seperator among the classes can be drawn as a hyperplane, which is the decision boundary that differentiates the classes in SVM. The support vectors are the data points that are closest to the hyperplane and affect its position and orientation. This method of SVM works in problems with two classes for the dependent variable. In the case of multi-class classification, $nclasses*(nclasses-1)/2$ classifiers are constructed and each one trains data from two classes. As a result, the number of classes significantly affects the execution time of the classification method. SVMs are considered effective in problems with high dimensional spaces. SVM algorithms use a set of mathematical functions that are defined as the kernel. The function of kernel is to take data as input and transform it into the required form. SVM method mainly uses different types of kernels RBF, linear and polynomial with the the most used type of kernel function to be RBF, because it has localized and finite response along the entire x-axis.

$$k(x_i, x_j) = exp(-\gamma||x_i - x_j||^2) \quad \text{: RBF Kernel}$$

$$k(x_i, x_j) = (x_i * x_j + 1)^d \quad \text{: Polyonimial Kernel}$$

$$k(x, y) = 1 + x * y + x * y * min(x, y) - \frac{x + y}{2} * min(x, y)^2 + \frac{1}{3} min(x, y)^3 \quad \text{: Linear Kernel}$$

SGD CLASSIFIER

Stochastic Gradient Descent is a simple but very effective approach to fitting linear classifiers under convex loss function such as SVMs and Logistic Regression. In the problem of minimizing an objective function that has the form of:

$$Q(w) = \frac{1}{n}\sum_{i=1}^{n} Q_i(w)$$

the parameter w that minimizes Q(w) has to be estimated. In stochastic gradient descent the parameter w is updated for each specific sample by:

$$w = w - \eta \nabla Q_i(w)$$

The SGD classifier performs a similar method by implementing regularized linear models with SGD learning. The gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule. Generally, SGD is an optimization on the training of a linear model.

MLP CLASSIFIER

Multi layer Perceptron is a supervised learning algorithm that learns a function f by training on a dataset with m dimensions in the input and o dimensions in the output. It can learn a non-linear approximator for classification and between the input and output layers, there are one or more non-linear layers called hidden layers. Each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation followed by a non-linear activation function. MLP trains the model by using some sort of gradient descent such as SGD and the gradients are calculated using Backpropagation. In the case of multi-class classification the output function is Softmax. The main advantage of the algorithm is learning non-linear models.

Convolutional and Residual Neural Networks

Convolutional Neural Networks is a class of Artificial Neural Networks most commonly applied to analyze visual imagery. As described previously, these neural networks consist of an input layer, hidden layers and an output layer. Convolution, pooling and fully connected layers are used to convolve the input, reduce the dimensions of data and connect every neuron between subsequent layers respectively. Each output value is computed by applying a specific function to the input values, which is determined by a vector of weights and bias. Part of training neural networks is separating the training set in similarly sized batches and adjusting the weights for each of them over multiple iterations to fit a suitable model. Residual Neural Networks are similar to convolutional neural networks but utilize skip connections or shortcuts to jump over some layers that contain nonlinearities and batch normalization in between. The purpose of the skip connections is to avoid the problem of vanishing gradients and the accuracy saturation problem, where adding more layers to a deep model increases the training error. The method of adding the output from previous layers to the layers ahead provides the ability to train much deeper networks and faster.

V. Canny Edge Detection

A Canny edge detector is a multi-step algorithm that detects the edges for any input image. It is described below. 1. Removal of noise in input image using a Gaussian filter. 2. Computing the derivative of Gaussian filter to calculate the gradient of image pixels to obtain magnitude along x and y dimension. 3. Considering a group of neighbors for any curve in a direction perpendicular to the given edge, suppress the non-max edge contributor pixel points. 4. Lastly, use the Hysteresis Thresholding method to preserve the pixels higher than the gradient magnitude and neglect the ones lower than the low threshold value.

VI. Results

The accuracy of the model provides a good measure of fitness, but cannot answer to the question "how bad the model is" and "where did it get it wrong". For example, would it be a bad prediction if the model predicted that there are 10 objects in an image with 9? Should that error have the same weight with a prediction of one object for the same picture? The loosen_error adds a small proportion to the accuracy of the model (less than 1), if the absolute value of the difference of the prediction and the labeling is less than 2, and the number of objects in an image is larger that 2 (predicting zero objects in an image with one is a serious mistake) Thus, we use the loosen_error that we defined and the confusion matrix, to have a better idea about the score of the model.

We first applied the GaussianNB and the Logistic Regression, that were used as base classifiers for the rest of our work. We also combined those two classifiers with computer vision algorithms to study their contribution to the task of object detection.

GaussianNB, Logistic Regression

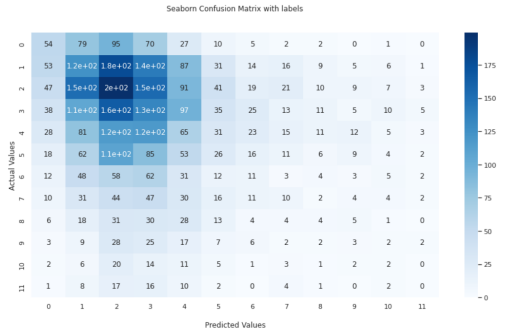
The gaussianNB seems to make its predictions randomly. From the confusion matrix, we can see that most of the time it predicts that there is only one object, and sometimes 7. On the other hand, the logistic Regression is able to distinguish some images with 2-5 objects, but it mush have learned to give that answer to most of the images.

```
sw_corr, corr = loosen_error(predictions, lab_test, 1/2)
print("The accuracy of the model is", corr)
print("The loosen accuracy of the model is", sw_corr)
```

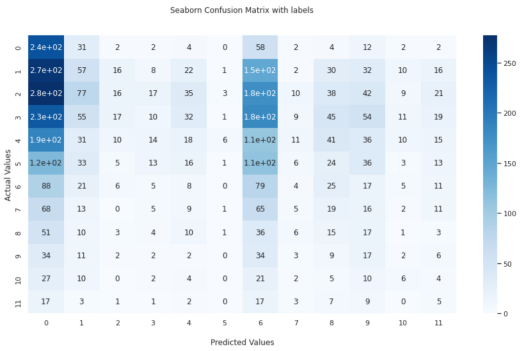
```
The accuracy of the model is 0.1388354186717998
The loosen accuracy of the model is 0.3455245428296439
```

```
sw_corr, corr = loosen_error(predictions, lab_test, 1/2)
print("The accuracy of the model is", corr)
print("The loosen accuracy of the model is", sw_corr)
```

```
The accuracy of the model is 0.05510105871029836
The loosen accuracy of the model is 0.17228103946102022
```



(a) Confusion matrix of the Logistic Regression classifier.

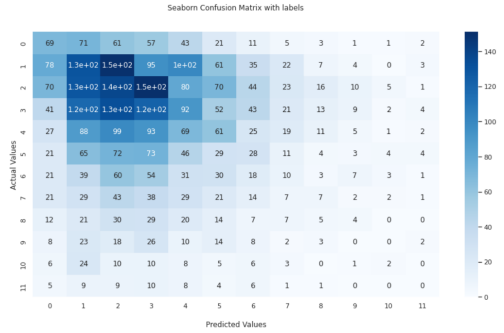


(b) Confusion matrix of the GaussianNB classifier.

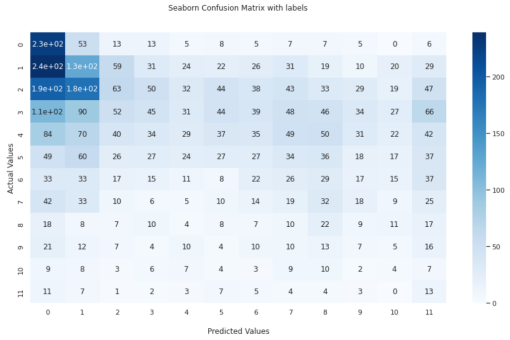
After training these models and calculating their scores, we apply the canny edge detection algorithm to the input images, to test if it helps the classifiers to make better predictions. As we can observe, the application of the algorithm improves the scores of the GaussianNB, but fails to improve the scores of the Logistic Regression model. The Logistic Regression seems to be a more stable and powerful model and thus, as a result, we cannot trust that the Canny algorithm will help to the classification task.

```
The accuracy of the model is 0.12503005530175523
The loosen accuracy of the model is 0.3207501803318105
Text(111.5, 0.5, 'Actual Values')
```

```
The accuracy of the model is 0.09064679009377254
The loosen accuracy of the model is 0.24549170473671555
Text(111.5, 0.5, 'Actual Values')
```



(a) Confusion matrix of the Logistic Regression classifier after applying the Canny algorithm algorithm.



(b) Confusion matrix of the GaussianNB classifier after applying the Canny algorithm

Random Forest, SGD, MLP

The next classifiers that were tested are the Random Forest Classifier and classifiers using stochastic gradient descent. The latter includes the SGD classifier and the MLP classifier using the SGD method. As mentioned previously, a big disadvantage of the Decision Trees is that they tend to overfit the model when there are dominant classes. A method to rebalance the classes is to randomly remove samples from the dominant classes, so that the classes do not have big differences in their samples sizes. This method of undersampling is utilized to compare if there are improvements in the results.

In the case of selecting a subset of the dataset without performing undersampling, the majority of the predictions in the SGD classifier are centered around 2-5 objects for each bin. As expected in the Random Forest classifier, the majority of the predictions is 3 or 4 objects, since these are the most dominant classes in the dataset. The MLP classifier has similar results to the SGD classifier.

After performing the undersampling, the classes are more balanced and this is depicted in the confusion matrices. The classifiers make mostly correct predictions about the samples that do not contain any objects but do not make specific predictions for the rest of the samples. This means that the classifiers cannot draw distinctions between the classes and the method of undersampling does not improve the training process. This method is utilized in other classifiers to test if it improves the results.

```
The overall accuracy is  0.16454954074653116
The loosen accuracy is  0.4275992285568212
```

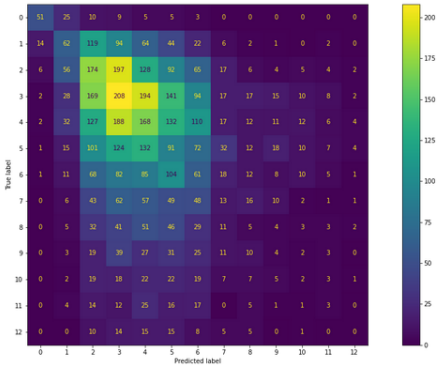
(a) Accuracy of the Stochastic Gradient Descent Classifier.

```
The overall accuracy is  0.1973812780926324
The loosen accuracy is  0.48583945205989404
```

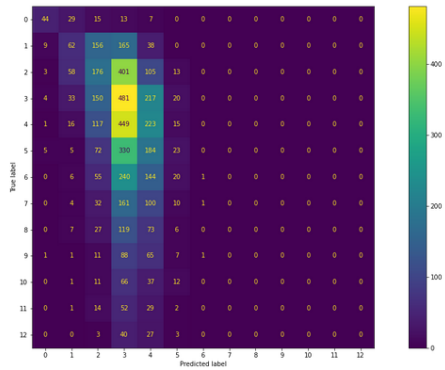
(b) Accuracy of the Random Forest Classifier.

```
The overall accuracy is  0.17119405901895643
The loosen accuracy is  0.4094589350891883
```

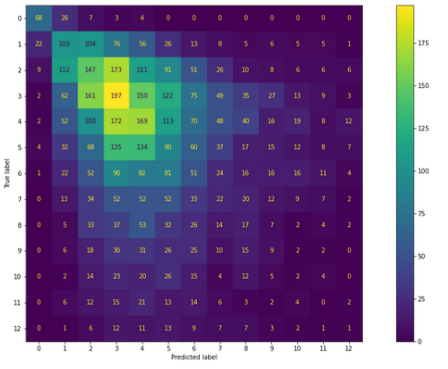
(c) Accuracy of the Multi-layer Perceptron Classifier.



(a) Confusion matrix of the Stochastic Gradient Descent Classifier.



(b) Confusion matrix of the Random Forest Classifier.



(c) Confusion matrix of the Multi-layer Perceptron Classifier.

```
The overall accuracy is  0.16454954074653116
The loosen accuracy is  0.4275992285568212
```

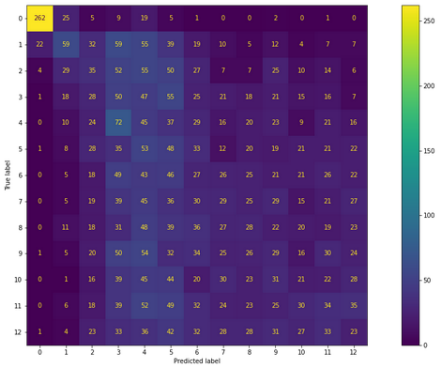
(a) Accuracy of the Stochastic Gradient Descent Classifier after undersampling.

```
The overall accuracy is  0.1973812780926324
The loosen accuracy is  0.48583945205989404
```

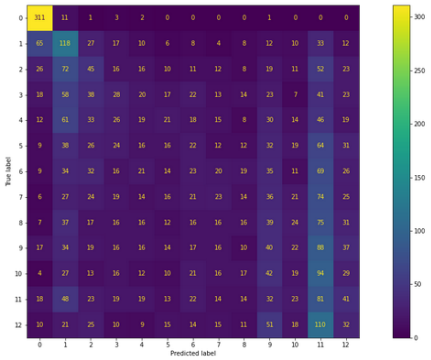
(b) Accuracy of the Random Forest Classifier after undersampling.

```
The overall accuracy is  0.17119405901895643
The loosen accuracy is  0.4094589350891883
```

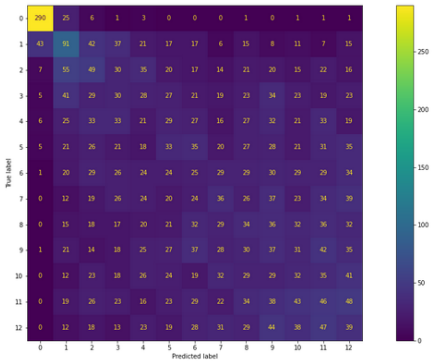
(c) Accuracy of the Multi-layer Perceptron after undersampling.



(a) Confusion matrix of the Stochastic Gradient Descent Classifier after undersampling.



(b) Confusion matrix of the Random Forest Classifier after undersampling.



(c) Confusion matrix of the Multi-layer Perceptron Classifier after undersampling.

Support Vector Machines

We then applied the Support Vector Machine (SVM) methods for the classification. Image classification was performed for 3 SVM kernels, rbf, linear and polynomial. As mentioned before, when we perform an undersampling of the images, the classifiers with rbf and linear kernel behave in the same way as the MLP classifier that we encountered above.

Therefore, they both seem to predict well images containing objects within range 2-6. Meanwhile, the accuracy of the classifiers with kernels rbf and linear fluctuates in the same level as their predecessors with the rbf to achieving a slightly better accuracy than the other two. Furthermore, it is observed that the rbf kernel has a considerable difficulty in locating more than nine objects, in contrast to the linear kernel which achieves slightly better results in this range of objects. On the other hand, the polynomial kernel seem to predict images with a number of objects from 6 to 8, which leads to a quite different confusion matrix from the previous classifiers. Consequently, this seems to be to the main reason for the very low accuracy compared to the other two kernels.

```
svm_rbf = SVC(kernel="rbf", max_iter=500, probability=True)
svm_rbf.fit(X_train_prepared, quan_train)
predictions = svm_rbf.predict(X_test_prepared)
print("Kernel = rbf: " + str(svm_rbf.score(X_test_prepared, quan_test)))
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/svm/_base.py:289: ConvergenceWarning: Solver terminated early (max_iter=500).  Consider pre-processing your data with StandardScaler or MinMaxScaler.
  ConvergenceWarning,
Kernel = rbf: 0.1880476947535771
```

(a) Accuracy of SVM Classifier with kernel: RBF.

```
svm_lin = SVC(kernel="linear", max_iter=1000, probability=True)
svm_lin.fit(X_train_prepared, quan_train)
predictions = svm_lin.predict(X_test_prepared)
print("The accuracy of linear model is: " + str(svm_lin.score(X_test_prepared, quan_test)))
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/svm/_base.py:289: ConvergenceWarning: Solver terminated early (max_iter=1000).  Consider pre-processing your data with StandardScaler or MinMaxScaler.
  ConvergenceWarning,
The accuracy of linear model is: 0.1703013481363997
```

(b) Accuracy of SVM Classifier with kernel: Linear.

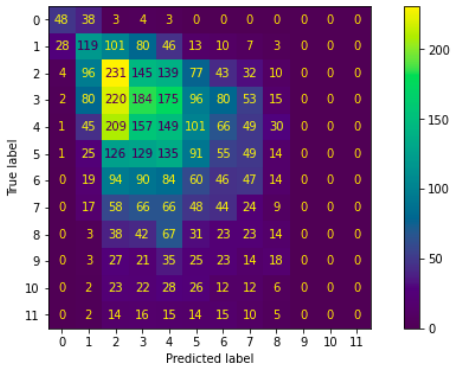
```
svm_poly = SVC(kernel="poly", max_iter=500, probability=True)
svm_poly.fit(X_train_prepared, quan_train)
predictions = svm_poly.predict(X_test_prepared)
print("Kernel = poly: " + str(svm_poly.score(X_test_prepared, quan_test)))
```

```
/opt/conda/lib/python3.7/site-packages/sklearn/svm/_base.py:289: ConvergenceWarning: Solver terminated early (max_iter=500).  Consider pre-processing your data with StandardScaler or MinMaxScaler.
  ConvergenceWarning,
Kernel = poly: 0.06177225182553779
```

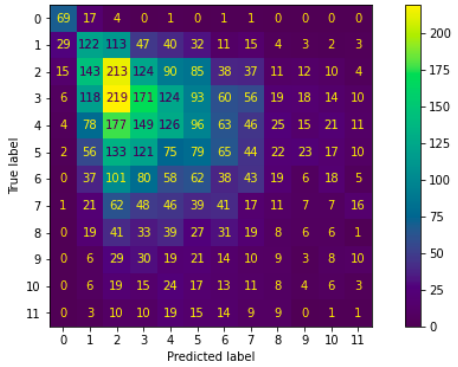
(c) Accuracy of SVM Classifier with kernel: Polynomial.

Residual Neural Network

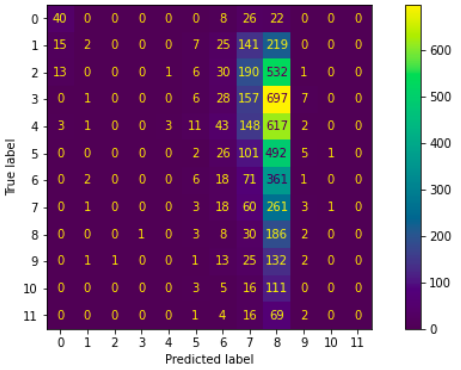
In the case of the convolutional neural networks, we use as a backbone a pre-trained residual neural network that consists of 34 layers, which is provided by Pytorch. Furthermore, the majority of the dataset is utilized for the training and validation processes; a subset of samples is used as the validation set, while the rest of the samples form the training set. After resizing the images, the dataset is split into batches and for each batch of the training set the parameters of the model are updated. Based on the fitted model, predictions are made on each batch of the validation set and an average of the predictions forms the



(a) Confusion matrix of SVM Classifier with kernel: RBF.



(b) Confusion matrix of SVM Classifier with kernel: Linear.



(c) Confusion matrix of SVM Classifier with kernel: Polynomial.

precision of the model. This process repeats for a number of epochs until the precision of the model can no longer be improved. Since the majority of the samples contain 2-5 objects, the subset of samples that contain less than 6 objects has been used for the training of the model.

As it is shown in figure 2, the precision on the validation set improves until the 12th epoch, and then the accuracy stabilizes until the 15th epoch. At the 16th epoch the curve starts to decline as the model tends to overfit. The accuracy of the model is higher compared to other classifiers, but we should mention that significantly more resources have been used. These results agree with the previously mentioned related work.

The confusion matrix displays the prediction at the 16th

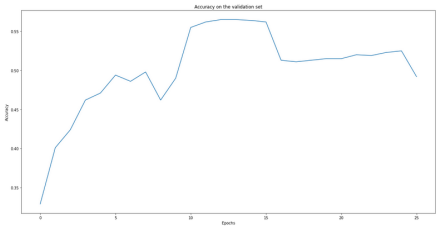


Figure 2: Validation curve on Resnet model.

epoch. As it is shown the majority of samples with no objects have been classified correctly, but as the number of objects increases the accuracy of the prediction declines. However, the majority of the false predictions are 1 tile away from the diagonial of the matrix. Thus, the model would have a better lossen_error accuracy.

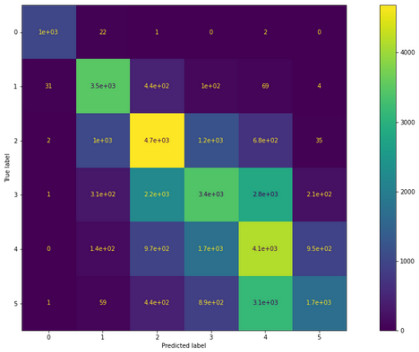


Figure 3: Confusion Matrix on the Resnet Model.

In general, convolutional neural networks work better for image classification as they can tune a large number of hyperparameters in a rather efficient time. Residual neural networks provide even stronger representations and produce faster results with a smaller penalty to the accuracy of the model. Neural networks are also better integrated, so that training in significantly large datasets is not as computationally heavy. High performing classification algorithms such as SVMs and Multilayer Perceptrons are computationally heavy and require tuning a large number of hyperparameters, but scikit-learn does not provide as much support to optimize their execution.

The accuracy of the different classifiers and the neural network is presented in the following table.

Table I: The accuracy of different classifiers

Classifier:	Accuracy(%):	Loosen(%)
GaussianNB	5.51%	34.55%
Logistic Regression	13.88%	17.22%
Random Forrest	19.74%	48.58%
Stochastic Gradient Descent	16.45%	42.75%
Multi-Layer Perceptron	17.11%	40.94%
SVM (rbf)	18.00%	-
SVM (linear)	17.03%	-
SVM (poly)	6.20%	-
CNN (Resnet)	56.5%	-

Our code is in jupyter notebook format and can be found on the following link:

https://github.com/theoOmathimatikos/amzn_bucket.
The models can be found in different notebooks, due to the fact that the project was collaborative and partitioned in three parts: first, the naive models combined with other computer vision techniques (lr-gnb-canny-test.ipynb), second the more powerful machine learning models (sgd-rf-mlp.ipynb), and, at last, the deep learning models (cnn-resnet.ipynb).

VII. Conclusion and future work

The Residual Neural Networks provided higher accuracy to the Bin Image Dataset classification task. The results of the model show that it gained an understanding of the problem, whereas the classifiers provided by scikit-learn seem to make random predictions. The maximum accuracy of the model is 56.5 %. It is important to note that the set of images is not ideal to demonstrate the strengths of the other classifiers. Many of the images are obscure and it is difficult to identify the number of objects on them even for the human eye. The class distribution is also heavily unbalanced, so high accuracy algorithms such as Decision Tree Classifiers tend to overfit.

An interesting concept is separating the images that present very clearly the objects they obtain in order to determine the effect of noise in the training. This escapes the purposes of this project and requires much time since there are hundreds of thousands of images to test. Another idea is identifying the objects in each image, which would help us understand better what the model has learnt.

It is evident that base models are not applicable to the task, and more complex and specific models are needed in order to create more powerful predictors.

References

[1] amazon-bin-images@amazon.com Amazon. Amazon bin image dataset. https://registry.opendata.aws/amazon-bin-imagery/.

[2] Fang Zhao et al. Dynamic conditional networks for few-shot learning. **Proceedings of the European Conference on Computer Vision (ECCV)**.

[3] Heinrich Kai et al. Everything counts: A taxonomy of deep learning approaches for object counting. **Twenty-Seventh European Conference on Information Systems (ECIS2019), Stockholm-Uppsala, Sweden**.

[4] Hisham Cholakkal et al. Object counting and instance segmentation with image-level supervision. **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**.

[5] Kaiming He et al. Deep residual learning for image recognition. **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**.

[6] Pablo Rodriguez Bertorello et al. Amazon inventory reconciliation using ai. **Industrial Manufacturing Engineering eJournal**.

[7] Zihao Xu et al. Deep neural networks for object enumeration. **2018 IEEE International Conference on Big Data (Big Data)**.