Summary

[Problem Definition 1](#_Toc100362051)

[Problem Introduction 1](#_Toc100362052)

[Potential Solutions 1](#_Toc100362053)

[Performance Metrics 2](#_Toc100362054)

[Confusion Matrix 2](#_Toc100362055)

[Accuracy 4](#_Toc100362056)

[Precision 4](#_Toc100362057)

[Recall 5](#_Toc100362058)

[F1-Score 5](#_Toc100362059)

[Problem Analysis 5](#_Toc100362060)

[Data Exploration 5](#_Toc100362061)

[Training Dataset Sizes 5](#_Toc100362062)

[Solving Algorithms 5](#_Toc100362063)

[Solution Implementation 5](#_Toc100362064)

[Data Preprocessing 5](#_Toc100362065)

[Solution Refinement 5](#_Toc100362066)

[Postprocessing 5](#_Toc100362067)

[Results 5](#_Toc100362068)

[Conclusions 6](#_Toc100362069)

[References 6](#_Toc100362070)

# Problem Definition

In this section, the problem will be defined, possible solutions will be presented and a discussion about performance metrics for evaluating the solution will be made.

## Problem Introduction

Information always played a great role in management processes, since decisions tend to be as good as the basis upon which they were built. Given the great value of this resource, it was only natural that data utilization would be vastly explored and expanded with time.

One of the main uses of data is the training of several Machine Learning models. Examples of these models can include Natural Language Processing, which allowed the creation of chat bots and translation tools, Computer Vision, responsible for automatic object and person identification and pattern recognition, and many others model types for different purposes.

The problem I intend to tackle is linked to Computer Vision and stock management. Since logistic planning is a hard task by itself, one of the ways data can improve operations is by providing vision to robots used in bin transportation in warehouses. Using Computer Vision algorithms and images provided by Amazon to train a classification machine learning model, we can reduce errors, automate operations and optimize stock placement.

In short, the main problem studied in this project is the creation and training of a Computer Vision model capable of identifying the number of items being carried in transportation bins. That is, to classify images into different classes that are divided between the number of items present in the picture.

## Potential Solutions

Solutions outside the realm of Machine Learning can become very time costly, e.g., human employees checking each transport bin throughout the day.

Taking that into consideration, the potential solutions for the problem can be seen as the ones that actually reduce costs and improve efficiency.

In this academic context, we will only look at solutions related to Computer Vision, that is, the different types of image classification model architectures we can use as base for our model.

These architectures differentiate between themselves by how the convolutional and fully connected layers were employed in order to improve a metric of choice (e.g., accuracy). Additionally, we can modify some sections of these pre-built architectures to fit it to our needs.

As PyTorch will be used as base package for model creation and training, the possible model architectures are the ones contained on Torchvision (PyTorch’s library with useful tools for Computer Vision).

The model architectures available for utilization are:

* AlexNet
* VGG
* ResNet
* Wide ResNet
* ResNeXt
* SqueezeNet
* DenseNet
* Inception
* GoogleNet
* ShuffleNet
* MobileNet
* MNASNet
* EfficientNet
* RegNet
* VisionTransformer
* ConvNeXt

Details on these architectures’ utilization can be found [here](https://pytorch.org/vision/stable/models.html)[1].

## Performance Metrics

Performance metrics are values that give us the measure of how well our model is predicting whatever it was trained for.

There are different metrics for different kinds of problems: if our model is configured for doing regression, the metrics used for classification cannot be used and vice-versa. And since the problem presented previously is a classification into classes, the metrics considered for evaluating the model were:

* Accuracy
* Precision
* Recall
* F1-Score

In the following subsections these metrics will be detailed, but first the concept of Confusion Matrix needs to be introduced.

### Confusion Matrix

In a classification problem, the training process consists in fitting a model to make predictions that will be compared to a label. This label is the target class that we want our model to output.

That idea might seem quite simple when we are dealing with binary classification, e.g., if an image is showing a dog or a cat. For this binary example, we will define that dog predictions are 1 (positive for dog) and cat predictions are 0 (negative for dog, therefore cat).

The possible outcomes in this scenario are:

* Prediction for a dog picture is 1: **true positive (TP)**, the model predicted that the image **was** a dog and got it right.
* Prediction for a cat picture is 0: **true negative (TN)**, the model predicted that the image **was** **not** a dog and got it right.
* Prediction for a cat picture is 1: **false positive (FP)**, the model predicted that the image was a dog when it was a cat.
* Prediction for a dog picture is 0: **false negative (FN)**, the model predicted that the image was a cat when it was a dog.

Having these predictions and labels at hand, it is possible to display the results of true and false positives and negatives as a Confusion Matrix. For this binary case, this matrix will have the following form:

Gráfico

Descrição gerada automaticamente

Figure - Binary Confusion Matrix.

source: <https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826>

In multi-class classification problems, it is not that simple. We will use another example to better show the creation of this matrix for more than two classes.

Consider an image classification problem where we want to predict if an image is of an apple, orange or mango. Our classification method will output one of these classes for each image it analyzes, that is, it will say that an image is **positive for a class**. Consequently, this image is a **negative for the other classes**.

When assembling the Confusion Matrix for this problem, one possible outcome is:

Gráfico, Gráfico de dispersão

Descrição gerada automaticamente

Figure - Multi-Class Confusion Matrix

source: <https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826>

In order to transpose the idea of TP, TN, FP and FN we will take the different classes into consideration.

Looking at the Apple class, we have that:

* The number of TP is 7.
* The number of TN is 8 (5 Orange predictions + 3 Mango predictions). This number includes all the times the model said an image was not an apple and got it right. Since we are looking only at the Apple class, it does not matter whether the predictions for Orange and Mango were correct, the point is that they were **negative** for Apple.
* The number of FP is 17 (8 Orange images + 9 Mango images)
* The number of FN is 4 (1 Orange prediction + 3 Mango predictions)

The method of segregating each class during performance metric evaluation is called One-Vs-Rest (OVR) approach.

It is noticeable that the OVR method will return a value *x* of scores for a given metric, where *x* is the number of classes in the classification problem. This is not ideal, since the objective of performance metrics is to provide a single value that give us the idea of how well the model is doing as a whole.

With that in mind, we use averaging techniques for metrics in multi-class classification:

* Macro: An arithmetic mean of all class metrics. A good choice for problems where the classes are balanced (similar number of samples in each class).
* Weighted: A weighted arithmetic mean where the number of samples serve as weights to account for imbalances between them.
* Micro: A simple division between correct predictions and all the predictions made by the model. It is similar to accuracy.

The concepts of TP, TN, FP and FN for multi-class classification methods will serve as building blocks for the performance metrics presented in the next subsections.

### Accuracy

The accuracy of a model is defined as the percentage of correct predictions in the total number of predictions, that is:

It is the one of the simplest and most intuitive performance metrics and portrays the overall hit rate of the classification method.

The concept of accuracy does not change for binary or multi-class classification problems.

### Precision

Precision focus on showing a number related to the predicted positives output by a model. In mathematical terms, it is translated as:

By optimizing a model for precision, we are saying that the **false positives** have significance on the result aimed by performing predictions. That is, predicting a negative as a positive will have relevant consequences.

### Recall

Recall focus on showing a number related to the actual positives analyzed by a model. Its mathematical form is:

By optimizing a model for recall, we are saying that the **false negatives** have significance on the result aimed by performing predictions. That is, predicting a positive as a negative will have relevant implications.

### F1-Score

F1-Score is a metric that combines both precision and recall scores in order to measure a model. Its mathematical form is:

# Problem Analysis

This section presents information about the dataset used during the development of the problem solution and the planning behind the proposed solution.

## Data Exploration

This subsection aims to present the problem through data visualization and exploration in order to better understand the details surrounding the problem and how to better solve it.

The dataset is divided into 5 classes ranging from the number of items presented in each image.

### Training Dataset Sizes

One important aspect of classification problems is the balancing of classes. To better understand how the classes are divided, the sizes of the training datasets is presented in the following figure and table.

Gráfico, Gráfico de barras

Descrição gerada automaticamente

Figure 3 - Training Sample Sizes (Visual Representation)

Table 1 – Training Sample Sizes (Detailed Values)

|  |  |
| --- | --- |
| Class | Training Sample Size |
| 1-item images | 859 images |
| 2-item images | 1609 images |
| 3-item images | 1866 images |
| 4-item images | 1661 images |
| 5-item images | 1312 images |

Except for the class of 1 item per image, the classes are fairly balanced.

### Images Details

Since we are dealing with image preprocessing and analysis, it is relevant to know how they are formatted and have an idea of how they look like.

As expected, the images in the training dataset are not formatted to be fed directly to the model. The image sizes ranges are:

Table - Image Size Ranges

|  |  |  |
| --- | --- | --- |
|  | Minimum Value | Maximum Value |
| Width | 295 pixels | 1100 pixels |
| Height | 231 pixels | 1171 pixels |

Examples of training images:

* 1-Item class:



Figure - Example of image with 1 item

* 2-Item class:

Uma imagem contendo no interior, cozinha, mesa, pequeno

Descrição gerada automaticamente

Figure - Example of image with 2 items

* 3-Item class:



Figure - Example of image with 3 items

* 4-item class:



Figure - Example of image with 4 items

* 5-Item class:



Figure - Example of image with 5 items

## Chosen Performance Metric

Given the nature of the problem, there is not a considerable difference in importance when false positives and false negatives are compared. The main objective of the developed solution must be the correct overall prediction.

Therefore, **accuracy** was chosen as the target performance metric to evaluate the model. The simplest and most intuitive metric is well suited to show how well the model performs given the project’s objective.

# Solution Implementation

This section aims to show the steps taken in the development of the classification model.

The platform used for these processes was Amazon Web Services (AWS), more specially Amazon Sagemaker and S3.

## Data Preprocessing

The raw dataset ([Amazon Bin Image Dataset](https://registry.opendata.aws/amazon-bin-imagery/)) provided Amazon holds 500,000 images and metadata from bins transported by robots in an operating Amazon Center.

The images are not divided or labelled in any way. In order to use them a prior division of train images would be necessary using the metadata JSON files by Amazon in the dataset S3 folder. That step is not necessary for this project since Udacity has kindly provided a JSON file that labelled 10,441 images into 5 different folders ranging from 1 item to 5 items in each picture. Therefore, it was decided that the files to be used for training, validation and testing would be the ones listed in the forementioned JSON file.

After downloading the data in the JSON files and performing some exploration, the train, evaluation and test datasets were chosen as:

* Train set: 70% of the complete dataset
* Evaluation set: 15% of the complete dataset
* Test set: 15% of the complete dataset

The first step of the process of data preprocessing was then to split the images

Therefore, the first step of preprocessing was to split the data the percentages

## Model Implementation

## Model Refinement

# Results

Afterward, you will collect **results** about the performance of the models used, visualize significant quantities, and validate/justify these values.

## Model Evaluation

# Conclusions

Finally, you will construct **conclusions** about your results, and discuss whether your implementation adequately solves the problem.

# References

[1] *Models and Pre-Trained Weights.* (n.d.). PyTorch. <https://pytorch.org/vision/stable/models.html>

[2] Bex, T. *Comprehensive Guide to Multiclass Classification Metrics* (Jun 9, 2021). Towards Data Science. <https://towardsdatascience.com/comprehensive-guide-on-multiclass-classification-metrics-af94cfb83fbd>

[3] Faria, H. *Project: Inventory Monitoring at Distribution Centers.* (n.d.).

[4]