Parcel Classification

William Swain
Virginia Tech
williamswain@vt.edu

Helen Zhu
Virginia Tech
helen19@vt.edu

Zain Mirza
Virginia Tech
zainmirza@vt.edu

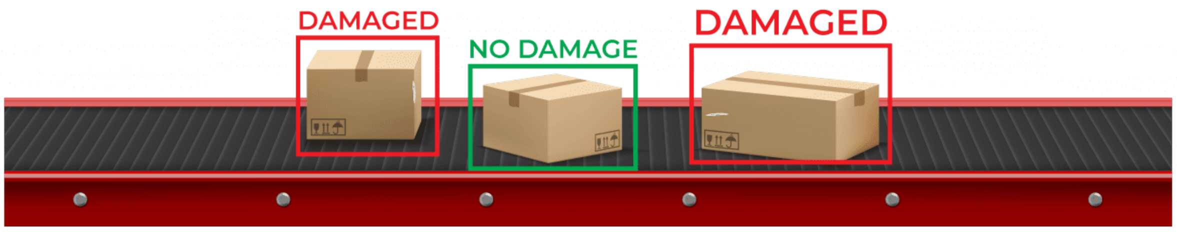

Figure 1: Damage detection in a commercial setting. Source: [8]

ABSTRACT

We set out to solve the problem of identifying damaged packages in hopes that such a system will be implemented in a distribution center to automatically identify and mark damaged packages as they pass through. As part of our research, we explored various CNNs and Visual Transformers to determine which machine learning model would yield the best results. We found that the VGG model with a custom top layer resulted in the best performance, with a decently high accuracy, and a very high recall. Recall is especially important in the warehouse environment where marking damaged packages as undamaged is a less favorable outcome compared to marking undamaged packages as damaged.

KEYWORDS

datasets, neural networks, damage detection, text tagging

1 INTRODUCTION

1.1 Problem Description

Parcel damage is a common problem that compromises the safety and can often lead to the damage of the contents of a parcel. Approximately 10% of parcels arrive at their destination damaged. Globally this equates to $15 billion in lost revenue. In most cases, the shipping company is liable for a replacement of the damaged item. This is a significant expense for shipping companies that can be mitigated through identification of where and what type of damage occurred. For our capstone project, we will train four different models, which are ResNet50, VGG-19, YOLOv5, and VisualBERT, to determine whether or not a package has been damaged, and compare the performance of all models. We anticipate that the best machine learning model can be incorporated into shipping warehouses to record where in its journey a particular package was damaged to identify patterns of unsafe handling along the distribution network[1].

1.2 Motivation

Sending out damaged parcels to customers will require companies to reship the products, and this unnecessary trip will cost the company extra expenses. This classification will also serve as evidence to determine liability and battle cases of fraud. Classifying whether a parcel is good before it leaves the warehouse is critical to cost-efficient shipping.

1.3 Challenges and Solutions

Before we can train any of the models, we first must collect over one thousand images of parcels that are either undamaged, crushed, folded, burned, leaking, or ripped. Collecting a dataset of this size comes with many challenges to overcome.

(1) (a) Challenge: A parcel can be damaged in many ways, so collecting a diverse data set with a balanced distribution is necessary.
    (b) Solution: William used GoogleImageCrawler from icrawler to gather data, splitting data into two main categories and five sub-categories under the bad package category.
(2) (a) Challenge: Manually clean a data set of one thousand images.
    (b) Solution: Three of us manually deleted irrelevant and under-standard images. Zain and Helen used the ImageHash library to clean the duplicated images.

Challenges with training models:

(1) (a) Challenge: If an algorithm has a high number of false negatives, it will mean many damaged parcels will continue down the line.
    (b) Solution: When evaluating models, prioritize recall to mitigate false positives.
(2) (a) Challenge: During training of the YOLOv5 series, the loss encountered NaN values or during testing the P/R/MAP values may all be zero.
    (b) Solution: Commented out check_amp and assign amp to False directly in train.py. The principle behind this solution is to convert the half-precision floating-point data of the graphic card to single-precision floating point data for calculation. Although this increased the precision, it also leads to a longer training time and higher memory usage.

2 RELATED WORK

2.1 Literature Survey

(1) The first article used a convolutional neural network, MobileNet, to detect and classify damaged packages. The accuracy, recall, and specificity of the model classification measured on the test set were all above 80%. This article also compared MobileNet with other models. They performed additional experiments with VGG16 and ResNet50 under the same conditions. The conclusion is that MobileNet is the optimal solution[3].

(2) The second paper used the pre-trained residual neural network ResNet-50 model for binary classification in the medical field. The authors improved the generality of the algorithm by adding a dropout layer and a fully connected layer with two output neurons. The article's algorithm architecture and implementation section introduces these contents, which is beneficial for our project[6].

(3) This third one uses the ResNet model for garbage classification. The authors adopted operations such as reducing the residual block of the pre-trained ResNet model and adding it to the batch normalization layer. Finally, they formed an improved residual network model (AResNet26). The final recognition accuracy of this model can be as high as 91.81%. Our project can refer to the improved approach taken in this article[7].

(4) This paper describes how researchers analyzed the effect of convolutional neural network depth on the accuracy of image recognition in a large-scale setting. The study was conducted by steadily increasing the depth of the network by adding more convolutional layers, which is feasible thanks to the small convolutional filters (3x3). The use of small 3x3 convolutional filters in convolutional networks is referred to as VGG (Visual Geometry Group), which may be handy for our project[4].

(5) This article uses VGG to extract features from bird species for accurate image classification. The researchers then compare the classification accuracy of different models that are using these extracted features. The Support Vector Machine (SVM) method was the most accurate model, outperforming K=Nearest Neighbor and Random Forest[2].

(6) Li et al. discuss the advantages and applications of VisualBERT. It serves as a guide for the usage of the framework and describes it underpinning to give readers a better understanding of VisualBERT. In essence, VisualBERT combines computer vision and natural language processing to enable reasoning capabilities for a visually-informed system. The name is indicative of the design, as BERT is a Transformer-based model for natural language processing. This technique can make downstream applications easier to develop, as it provides human-readable information[5].

2.2 Limitations of Existing Approaches

Classification problems such as this one are highly dependant on good quality data. The CNN needs to be fed an image extremely representative of what it will encounter in the test data to attain an acceptable accuracy. As such, the strength of our model will be limited by the quality of the training data, in this case images, that we are able to obtain.

3 PROPOSED APPROACH

3.1 Problem Definition (Mathematically)

Classification is a supervised learning task in which we aim to learn a mapping function from input features to a set of discrete output labels. In the case of classifying good and bad packages, the input features might include various attributes of the package such as weight, size, fragility, and so on. The output labels would be binary, with one label indicating a good package and the other indicating a bad package. We collected relevant training data in the form of images by scraping images that matched our keyword searches from the internet. Then we manually cleaned the data by removing duplicates and inaccurate images.

In particular, the project will involve training 4 different machine learning models: ResNet50, VGG-19, YOLOv5, and VisualBERT, to identify whether or not a package has been damaged. These models are all well-known deep learning models that are commonly used for computer vision tasks such as image recognition.

Once trained, the models will be evaluated and compared to determine which one performs the best in terms of accurately identifying whether or not a package has been damaged. The best-performing model will then be incorporated into shipping warehouses to record where in its journey a particular package was damaged, which can help identify patterns of unsafe handling along the distribution network.

Mathematically, the problem involves developing a set of binary classifiers, where each classifier takes an image of a package as input and outputs a binary value indicating whether the package is damaged or not. The performance of each classifier can be measured using metrics such as accuracy, precision, recall, and F1-score. We prefer to have more false positives than false negatives, because it's better to pull out an intact package than to let a damaged one pass. To reduce the number of false negatives, we can compare different models and prioritize those with a higher recall rate. Therefore, we decided recall rate as our main evaluation metric. The best-performing classifier can then be chosen based on these metrics while prioritizing recall.

3.2 Data Pre-processing/Feature Engineering

The image data set will be 1000 images split evenly between good packages and bad packages. The bad packages category contains five subcategories: burned packages, leaking packages, crushed packages, ripped packages, and folded packages. To maintain a balanced data set, 50% of our data will be images of good packages, 10% of ripped packages, 10% of burned packages, 10% of leaking packages, 10% of crushed packages, and 10% of folded. We will then have 500 images of good packages and 500 images of bad packages-100 images for each subcategory of bad packages. These images will be manually cleaned, removing irrelevant data and duplicate data. They will then be formatted as needed for each model.

3.3 Architecture

The pipeline illustrated in Figure 2 serves as the framework for this project.

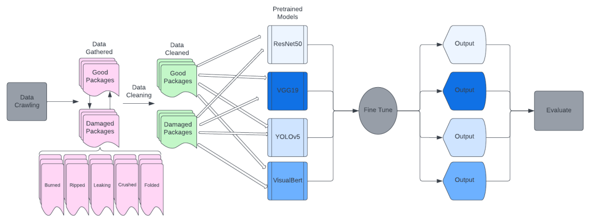

Figure 2: Pipeline

The data is scraped, cleaned, split, and plugged into each of the models. The models each have their own unique structure.
The structure of ResNet is shown in Figure 3.

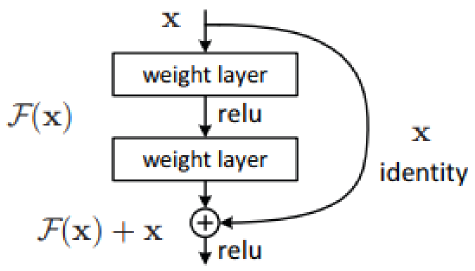
x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 3: A single block of a ResNet architecture
Source: [9]

ResNet50 has 50 of these blocks, each using both its input and output as input for the next block in the structure.
VGG19, however, is far more linear:

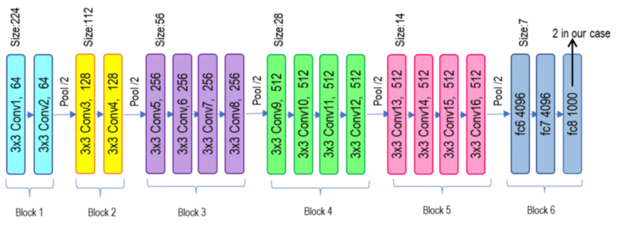

Figure 4: A VGG19 structure
Source: [10]

VGG19 uses 16 convolution layers, which are grouped into 5 blocks. After each one of these blocks, there is a pooling layer that reduces the size of the image and increases the number of filters of the convolution layer. This increase in filters gives it far more trainable parameters than the other models.

Figure 5 displays the architecture of YOLOv5.

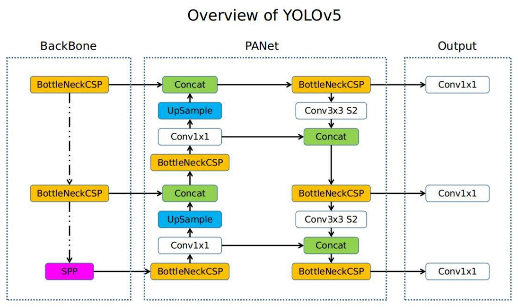

Figure 5: The structure of Yolov5
Source: [11]

The network structure of YOLOv5 consists of three parts: the backbone, neck, and head. In the yolov5s.yaml file, the author combined the head and neck layers. YOLOv5 is a fully convolutional network, meaning it consists solely of convolutional layers, batch normalization layers, etc., without fully connected layers. The backbone's main role is to extract features and continuously reduce the feature map's size. The backbone comprises several main structures, including the Convolutional (Conv) module, the Cross Stage Partial (C3) module, and the Spatial Pyramid Pooling (SPP) module. The neck structure is a feature pyramid network (FPN) that combines shallow graphic features with shallow semantic features. The head layer is the Detect module, with a simple network structure consisting of only three 1*1 convolutions, corresponding to three detection feature layers.

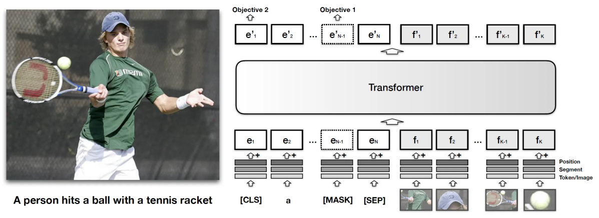

Figure 6: The structure of visualBert
Source: [12]

VisualBERT utilizes visual embeddings on top of the standard BERT model components. These embeddings correspond to bounded regions in an image obtained via an object detector. Visual, segment, and position components are summed, then passed to a multi-layer transformer. Self-attention enables the discovery of implicit relationships between language and visual features.

4 EXPERIMENTAL EVALUATION

4.1 Dataset Exploration Showing Observed Patterns

Collecting the custom dataset required scraping images from Google using Google's iCrawler. In order to mitigate irrelevant data, multiple search keywords and dates were used with each crawl. The purpose of this was to mix up what images the iCrawler would retrieve, as a single keyword with a single date would result in many duplicate images. After multiple queries were made, we then manually cleaned the data by scanning through the images and deleting all images that were irrelevant or duplicate. With the data cleaned and the final image count being 500 for good packages, 100 for burned packages, 100 for ripped packages, 100 for folded packages, 100 for crushed packages, and 100 for leaking packages, the data set was balanced and ready to be processed.

4.2 Model Demonstrating

(1) ResNet50: We used a pre-trained ResNet50 model for image classification. To normalize the images according to the model architecture, we referenced the preprocess_input function from the ResNet50 library. Our test generator had a batch size of 1, while the training generator had a batch size of 32. We defined a base model equal to ResNet50, setting the include_top parameter to "false" since we only had two classes of images and needed our own dense layer as output. To obtain a transfer learning model from the imagenet dataset, we set the weight parameter to "imagenet". The output variable took the output of the base model, and we added new layers to the model, including a GlobalAveragePooling layer and a dense layer as the last layer. By applying the dense layer, we created a transfer learning model that utilized our own classes for prediction. Next, we defined a model that took the input from ResNet50 and the output from the prediction on the final layer. Since the pre-trained weights were already advanced, we set our model to be non-trainable. We trained the model for 10 epochs, obtaining a total loss, and then saved the model.

(2) VGG19: The approach used in ResNet50 was also used for VGG19. The preprocessing input was taken from the VGG19 library and used to create a data transformer for which all training and testing data would be run through. We removed the top layer of the pretrained model with the imagenet weights and again added a GlobalAveragePooling layer and two dense layers. The weights of the base model layers were again frozen, so that the only layers trained were the ones added on top. The model then trained for 10 epochs and saved.

(3) YOLOv5: We utilized the pre-trained YOLOv5 model yolov5s.pt from GitHub for our project. In order to apply YOLOv5 to our specific dataset, we made the decision to split it into three separate sets - training, testing, and validation - using a 6:2:2 ratio. To label our dataset, we opted to use graphical image annotation tool LabelImg. As for training the model itself, we created a package.yaml file and then proceeded to edit the arguments found in the train.py file.

(4) VisualBERT: For our visual transformer model exploration, we used the pre-trained visualBERT model from the Hugging Face transformers Python package. We placed a linear layer on top of the last hidden state of the [CLS] token using PyTorch to tailor the model to our data set. We utilied a training/validation/test split of 70/20/10. We trained for 3 epochs with a batch size of 4 and a learning rate of 2e-5. Preliminary results showed a training loss ranging between 0.1 and 0.02 for epoch 3, as well as 1.0 accuracy.

4.3 Final Result Analysis

The models yielded the following results:

Final Results				
Metric	ResNet50	VGG19	Yolov5	visualBERT
Accuracy	0.888	0.95	0.9032	0.465
Precision	0.875	0.9	0.80036	0.036
Recall	0.897	1.0	0.73545	1.0
Specificity	0.878	0.909	0.8667	0.454
F1-Score	0.886	0.947	0.76637	0.070

Currently, VGG19 has the highest accuracy, recall, and F1-Score of all the models. These are the highest priority evaluation metrics, so VGG19 is the best model so far for the task of classifying parcels. Although ResNet50 has a higher specificity score than VGG19, it is not an evaluation metric that holds much importance so this high value is insignificant. This is due to the specific task of classifying good and bad packages, where it is much better to have a false positive and pull a package off the line than to let a damaged one continue. Thus, it is better to mitigate false negatives by prioritizing recall than to mitigate false positives by prioritizing specificity.
YOLOv5 behaves alright since our goal is to reduce the false negatives on the premise of high accuracy. There can be various reasons why the recall rate of YOLOv5 may not be very high. One reason could be the size and complexity of the dataset used to train the model. If the training data does not represent the entire spectrum of objects or situations that the model is expected to encounter, it may not perform well in certain scenarios, leading to a lower recall rate. We only have 1000 images and maybe this is one reason. The other reason is that it's possible that the recall rate of YOLOv5s is lower compared to deeper models due to its simpler architecture. Finally, the model's hyper-parameters, such as learning rate, and batch size, can also affect the recall rate. The visualBERT model showed very poor performance and further investigation is needed to determine the cause. We suspected dataset size and model generalizability as possible causes. Changing the training/testing split to match the other models and analyzing training loss and accuracy did not reveal the issue.

5 FUTURE WORK

If we were to continue with this project, we would likely shift from binary classification to multiple classification since we have various types of broken parcels. Additionally, we could expand our dataset to ensure it encompasses the full range of parcels.
To increase the performance:
For YOLOv5: To increase the recall rate, we might still need to continue to find the optimal set of hyper-parameters that balances

the trade-off between precision and recall. Besides, we could try more complex models like YOLOv5m or YOLOv5l to acquire more precise result.
For VGG19 and ResNet50, their performance was satisfactory, but could benefit from a larger dataset and further experimentation with hyperparameters.
For visualBERT we would further investigate the poor results and experiment with different hyperparameters and dataset sizes. Reimplementating the model with increased control over its setup will likely yield better results.

6 CONCLUSION

According to the results obtained, VGG is currently the best model for the specific task of classifying good and bad parcels, outperforming all the other models in every evaluation metric except for specificity. This can be attributed to the structure of the model. The architecture of the VGG19 model is quite shallow, as it only has 16 convolution layers as opposed to the 50 that Resnet50 has. This may be part of why VGG performed so well, as its shorter structure was better suited for the smaller dataset of only 1000 images. Despite having fewer layers, VGG has far more filters than every model, giving it far more trainable parameters than the others. We assumed that this would lead to overfitting, but VGG's evaluation metrics suggest that VGG's extraction method using more parameters is best for the task of parcel classification. Something was definitely wrong with the implementation of visualBERT and it cannot be assumed that the evaluation metrics returned are the best possible results. For this reason, we cannot conclude on whether visualBERT is the worst model for this specific task, but we also cannot rule out that it could potentially be the best. More research and development are required to determine visualBERT's ability to classify parcels and how it compares with the other models tested.

7 REFERENCES

[1] B. H. Team, "The Hidden Costs of Packaging Damage," Black-Hawk Industrial Supply - The #1 Choice For Manufacturers, 02-Jun-2020. [Online]. Available: https://www.bhid.com/hidden-costs-packaging-damage. [Accessed: 21-Mar-2023].

[2] Islam et al. (2019, July 27). Bird Species Classification from an Image Using VGG-16 Network: Proceedings of the 7th International Conference on Computer and Communications Management. ACM other conferences. [Accessed: 21-Mar-2023].

[3] Kim et al. (2022, March 3). Image classification of parcel boxes under the underground logistics system using CNN mobilenet. MDPI. [Accessed: 21-Mar-2023].

[4] K. Simonyan et al. (2015, April 10). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv.org. [Accessed: 21-Mar-2023].

[5] L. Li et al. (2019, August 9). VisualBERT: A Simple and Performant Baseline for Vision and Language. arXiv.org. [Accessed: 21-Mar-2023].

[6] Meng et al. (2023, February 6). Artificial intelligence based analysis of corneal confocal microscopy images for diagnosing peripheral neuropathy: A binary classification model. MDPI. [Accessed: 21-Mar-2023].

[7] Yi et al. (2022, August 1). Research on garbage image classification and recognition method based on improved ResNet Network Model: Proceedings of the 2022 5th International Conference on Big Data and internet of things. ACM Other conferences. [Accessed: 21-Mar-2023].

[8] Everything You Need to Know About Packaging Inspection with Image Recognition, Visionify, https://visionify.ai/everything-you-need-to-know-about-packaging-inspection-with-image-recognition/. Accessed 13 Mar. 2023.

[9] Yawar, Md. (2023, March 16) "ResNet Architecture." Code Studio, https://www.codingninjas.com/codestudio/library/resnet-architecture.

[10] Khattar, Anuradha. (2022, September) "VGG-19 Architecture." Research Gate, https://www.researchgate.net/figure/VGG-19-Architecture-39-VGG-19-has-16-convolution-layers-grouped-into-5-blocks-Afterfig5359771670.

[11] (2020, July)"Overview of Model Structure about Yolov5." GitHub, https://github.com/ultralytics/yolov5/issues/280.

[12] Tsang, Sik-Ho. (2022, May 28) "Review-Visualbert: A Simple and Performant Baseline for Vision and Language." Medium, Medium, https://sh-tsang.medium.com/review-visualbert-a-simple-and-performant-baseline-for-vision-and-language-616ca0da698.