POTATO CHIPS QUALITY CONTROL

- DEEP LEARNING FINAL PROJECT
- PROF. BEHNAZ BOJD

Group 2

URVI VAIDYA

MEHAK KATRA

MOIZ NAWAB

KENTRICK KEPAWITONO

AYSOUDA VATANPARAST

SHAHIN CHINICHIAN MOGHADDAM

AGENDA

Introduction

Data Summary

Fitting Training Set

Fitting Validation Set

Best Model

Business Implications

Conclusion

UCIrvine | The Paul Merage School of Business

Enhancing Food Safety with Deep Learning in Computer Vision

- Deep learning as a subset of AI: Enhances computer vision for safety evaluation and inspection in the food industry
- Deep neural networks & digital cameras: Capture images of fried potato chips on the packaging line
- Real-time analysis: Sophisticated algorithms analyze images for quality issues
- Surpassing manual inspection: Overcomes limitations such as inconsistency, subjectivity, and high costs
- Automated & reliable evaluation: Ensures product safety and improves overall efficiency in the food industry

UCIrvine | The Paul Merage School of Business

Problem Explanation

Objective:

Develop a Deep Learning model to automatically classify potato chip JPG images as non-defective or defective based on quality control standards.

Outcome Variable and Its Type:

- Outcome variable: Binary classification Label 0/1
- Quality is determined based on factors such as size, shape, color, and presence of defects

Independent Variables (X variables)

- Input data: JPG images of potato chips taken during the production process
- Image features: Width, height, color channels (RGB)

AGENDA

Introduction

Data Summary

Fitting Training Set

Fitting Validation Set

Best Model

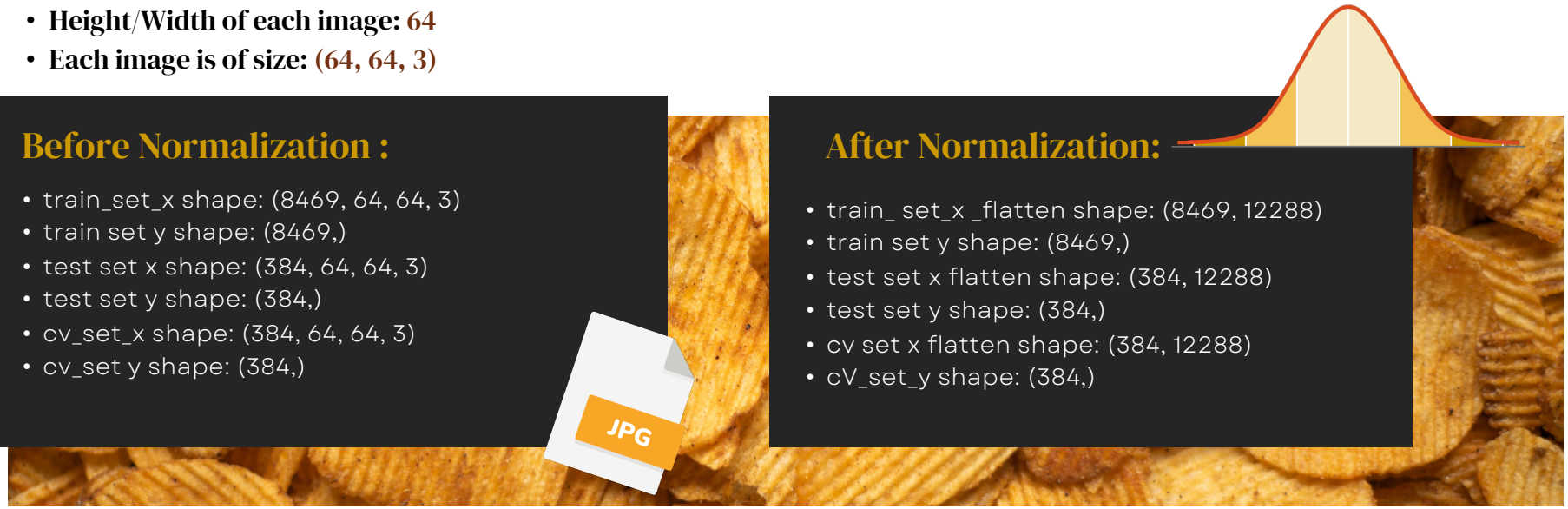
Business Implications

Conclusion

Dataset Summary

Data Preprocessing and Normalization

- **Image resizing: Ensure uniform dimensions for all JPG images**
- **Assign labels for defective (0) and non-defective (1) images**
- **Pixel intensity normalization: Scale pixel values to the range [0, 1] for better model performance**
- **Number of training examples: 8469**
- **Number of testing examples: 384**
- **Number of cross-validation examples: 384**
- **Height/Width of each image: 64**
- **Each image is of size: (64, 64, 3)**



Before Normalization :

- train_set_x shape: (8469, 64, 64, 3)
- train set y shape: (8469,)
- test set x shape: (384, 64, 64, 3)
- test set y shape: (384,)
- cv_set_x shape: (384, 64, 64, 3)
- cv_set y shape: (384,)

After Normalization:

- train_ set_x _flatten shape: (8469, 12288)
- train set y shape: (8469,)
- test set x flatten shape: (384, 12288)
- test set y shape: (384,)
- cv set x flatten shape: (384, 12288)
- cV_set_y shape: (384,)

Data Split

Original data

- **Total:** 961 potato chip images
- **Two categories:**

Defective: 461 JPG images

Non-Defective: 500 JPG images

Augmented Data

- **To create a bigger dataset**
- **Increased Training data**

Dataset	Defective	Non-Defective	Total
Training	4069	4400	8469
Cross-validation	184	200	384
Testing	184	200	384

AGENDA

Introduction

Data Summary

Fitting Training Set

Fitting Validation Set

Best Model

Business Implications

Conclusion

Evaluating Training Performance with Human and Logistic Regression :

a For Metrics, we use binary accuracy to evaluate the training performance

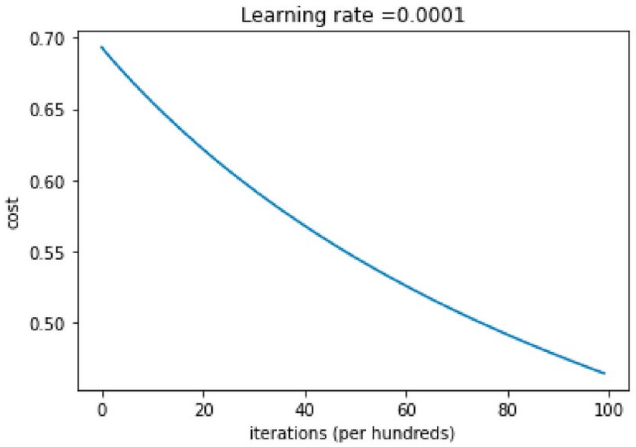
why we chose these:

- The binary accuracy measures the proportion of correctly predicted labels or outputs by the model on the training dataset.

b

Non -Defective Defective

- Considering the minimum difficulty level, human can distinguish easily



Cost Function Graph

c With Logistic Regression Model (no hidden layers), we get accuracy of:
Train Accuracy: 91.81%

Evaluating Training Performance with Bigger Neural Network

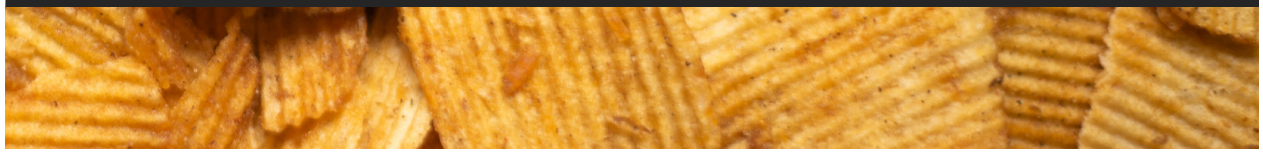
We try different numbers of hidden layers and hidden units

The most accurate model is:
Neural Network with 2 hidden layers:

- 44 Hidden Units in the first layer
- 4 Hidden Units in the second layer

Train Accuracy:

93.88%



Train Accuracy	
Neural Network 1 Hidden Layer 4 Hidden Units	49.83%
Neural Network 2 Hidden Layers 7 & 4 Hidden Units	91.94%
Neural Network 2 Hidden Layers 12 & 4 Hidden Units	88.77%
Neural Network 2 Hidden Layers 44 & 4 Hidden Units	93.88%

Optimizers	Train Binary Accuracy
Adam	85.36%
Adamax	87.59%
SGD	75%

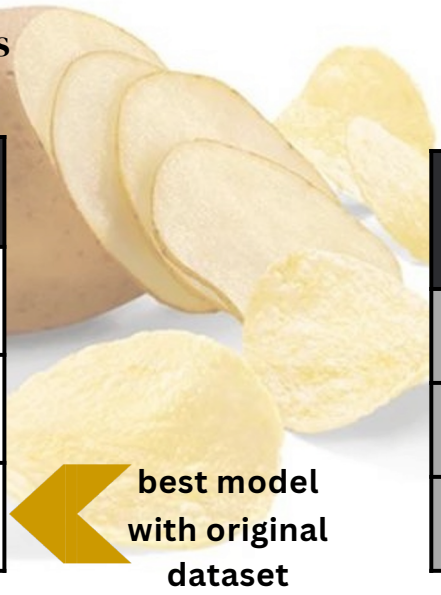
Improving Training Performance with Optimizers

- Adam and SGD cause the model to converge to a suboptimal solution, leading to lower accuracy
- Adamax leads to slightly higher binary accuracy

Improving Training Performance with Epoch & Weight Initialization

Does increasing number of epochs affect training performance?

Epochs	Train Accuracy
10	83.23%
30	85.36%
40	88.27%



best model with original dataset

- As we have more epochs, we get higher train accuracy

Does performing weight initialization affect training performance?

Weight Initialization methods	Train Accuracy
Random Initialization	52.08%
He Initialization	82.29%
Xavier Initialization	75.78%

- Weight Initialization does not necessarily improve our accuracy

AGENDA

Introduction

Data Summary

Fitting Training Set

Fitting Validation Set

Best Model

Business Implications

Conclusion

Validation Data VS Training Performance

Data	Accuracy
Training Accuracy (with best parameters)	98.96%
Test Accuracy (with best parameters)	95.74%
Cross Validation (with best parameters)	93.88%

- Bias is low at around 1-2%
- Variance is moderate (approximately 5%)
- Over-fitting in our model

Our best primitive model is:

- Neurons in the first layer: 44
- Neurons in the hidden layer: 4
- Neurons in the output layer: 1

L2 Regularization VS Dropout Regularization

- Variance - 5%
- Less-to-No Overfitting
- L2 regualrization doesn't improve the performance

λ	Training Accuracy	Test Accuracy	Cross-validation accuracy
0	86%	88%	89%
0.003	81%	85%	83%
0.0005	74%	83%	81%
0.0008	73%	71%	68%
0.0001	86%	83%	82%
0.007	52%	52%	52%

- Dropout regularization doesn't help improve the performance
- It either underfit or overfits the model
- Model needs further adjustments

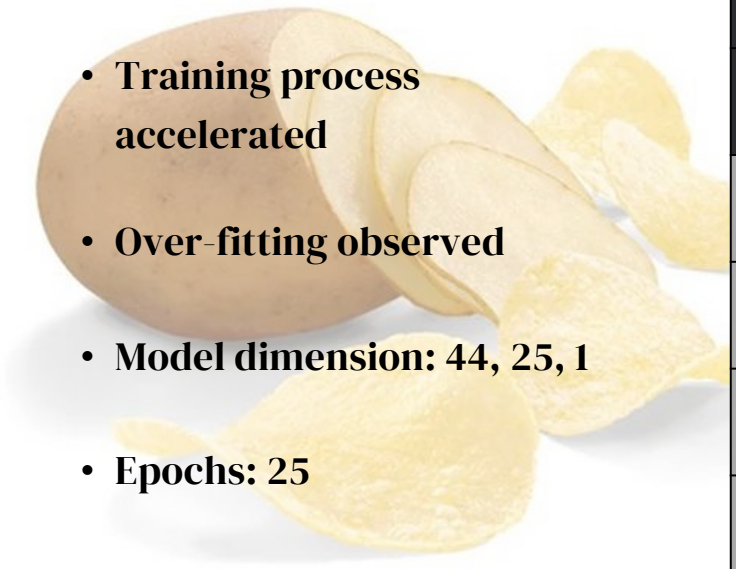
Dropout Probability	Training Accuracy	Test Accuracy	Cross-validation accuracy
0.2	70%	85%	86%
0.4	65%	84%	83%
0.5	71%	66%	66%
0.6	86%	71%	68%
0.8	51%	52%	52%

Validation Performance Improvement: L2 & Dropout Regularization

- Used augmented dataset
- Causes under-fitting/over-fitting
- Not the best approach for our model

λ	Dropout	Training Accuracy	Test Accuracy	Cross-validation accuracy
0.001	0.2	76%	58%	56%
0.0001	0.4	80%	73%	72%
0.0001	0.6	79%	83%	80%
0.0005	0.6	78%	87%	89%
0.0005	0.4	81%	76%	75%

Validation Performance Improvement: Batch Normalization



- Training process accelerated
- Over-fitting observed
- Model dimension: 44, 25, 1
- Epochs: 25

Batch Normalization		Training Accuracy	Test Accuracy	Cross-validation Accuracy
Momentum	Epsilon			
Default	Default	99%	66%	67%
0.9	1.00E-05	99%	88%	83%
0.5	1.00E-06	99%	87%	90%
0.3	1.00E-07	98%	92%	90%
0.6	1.00E-08	99%	92%	91%

Validation Performance Improvement: Batch Normalization & Dropout Regularization

- Over-fitted model
- If dropout probabilities increase, the model performance gets better
- Not much improvement in comparison with base model

Batch Normalization	Dropout	Training Accuracy	Test Accuracy	Cross-validation Accuracy
Default	0.3	96%	53%	50%
Default	0.5	94%	59%	52%
Default	0.6	94%	95%	90%
Default	0.8	92%	95%	90%
Default	0.1	97%	48%	48%

Validation Performance Improvement: Batch-Normalization/L2/Dropout Regularization

- All three techniques together do not have a desired impact on the performance
- Our model works best without these techniques
- Variance ~5% --> acceptable

λ	Batch Normalization	Dropout	Training Accuracy	Test Accuracy	Cross-validation Accuracy
0.003	Default	0.3	88%	59%	54%
0.0005	Default	0.5	88%	52%	52%
0.0008	Default	0.6	87%	76%	77%
0.0001	Default	0.8	88%	65%	60%
0.007	Default	0.1	88%	74%	69%

Validation Performance Improvement: Early Stopping

- Monitored - Validation Loss, Patience = 5
- Early stopping does not improve validation performance
- Base model works best
- These techniques aren't needed in our model to improve performance

Training Accuracy	Test Accuracy	Cross-validation Accuracy
70%	85%	83%

AGENDA

Introduction

Data Summary

Fitting Training Set

Fitting Validation Set

Best Model

Business Implications

Conclusion

UCIrvine | The Paul Merage School of Business

Best Performance

Building the Best Model

- Our best model has hidden layer size [44, 4, 1]
- Learning rate: 0.002674757271187464
- Activation functions: ReLU & Sigmoid
- Optimizer: Adam

Training the Best Model with TensorFlow

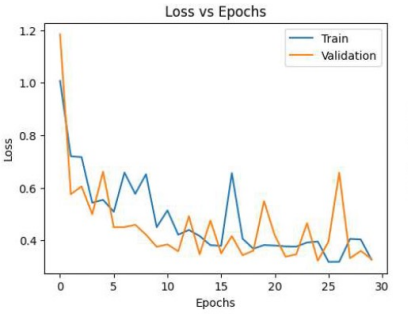
- Sequential model with 3 layers: 44 nodes in the first layer, 25 nodes in the second layer, and 1 node in the output layer
- Compile the model with Binary Crossentropy loss and Binary Accuracy metric
- Train the model with 30 epochs and batch size of 64

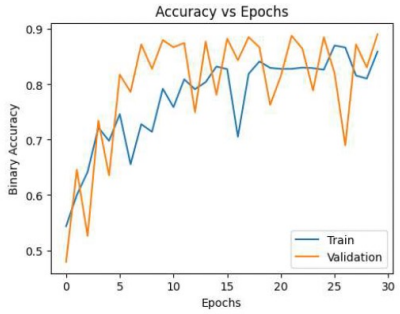
Selection of Best Model: Convolutional Neural Network (CNN)

Comparative Analysis of CNN & NN Performance

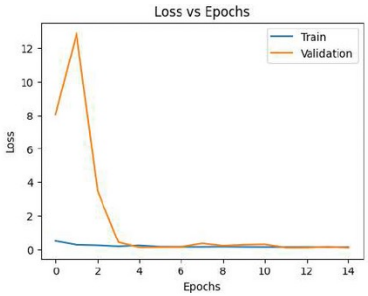
- **Superior Performance:** Outperformed NN in training, validation, and testing stages
- **Accurate Predictions:** Correctly identified shadows, reducing false defectives
- **Ideal for Task:** Superior in processing spatial information, key for image classification

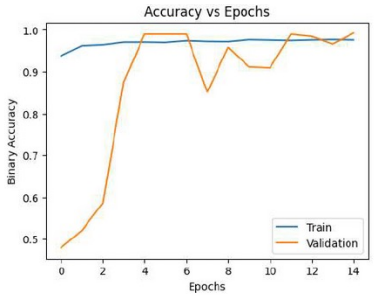
NN Performance:
89%





CNN Performance:
99%





NN Pred: Defective Chip
CNN Pred: Non-Defective chip

AGENDA

Introduction

Data Summary

Fitting Training Set

Fitting Validation Set

Best Model

Business Implications

Conclusion

Business Benefits of the Deep Learning Model in the Potato Chips Industry

UCIrvine | The Paul Merage School of Business

REDUCE CUSTOMER COMPLAINTS: The model's 89% test accuracy ensures that only top-quality potato chips are used for sale.

Time and Cost Savings:

- Time Savings
- Cost Reduction
- Competitive Advantage

Business Benefits:

- Increased Revenue
- Reduced Waste
- Efficiency and Productivity
- Scalability and Adaptability
- Continuous Improvement

Cost Benefit Analysis

Objective: To evaluate the potential savings and advantages of implementing our deep learning model for chip detection from a customer complaints perspective.

The following assumptions were considered:

- Cost of manual inspection per chip: $0.10
- Cost of inspection per chip using the model: $0.03
- Average cost per customer complaint: $70
- Average number of customer complaints per defective chip: 0.2
- Accuracy: 89.1% (assuming a human would have the same accuracy)

Cost of Complaints Assuming No Detection Model

By multiplying the average number of complaints per chip, the number of undetected defective chips, and the cost per complaint, we determined the cost of complaints in the absence of a detection model.

Comparison of Potential Cost Savings

Number of Chips	Potential cost (no model)	Total Cost Manual Inspection	Total Cost Model Inspection	Potential savings Manual	Potential savings Model
384	$ 2,576	$ 319	$292	$ 2,257	$ 2,284
1000	$ 6,720	$ 832	$ 762	$ 5,888	$ 5,958
18000000*	$ 120,960,000	$ 1,800,280	$ 540,280	$ 119,159,720	$ 120,419,720

- Comparing the potential savings from manual inspection and model detection with the cost of complaints assuming no detection model
- Assess the financial benefits and cost reductions that our deep learning model offers

*No of chips assuming $303 mil in sales per month with 12 chips a bag at $2 (https://www.statista.com/statistics/188220/leading-potato-chips-vendors-in-the-united-states-in-2011/)
* split of defective and non- defective chips is 48:52 (we maintain this split in our larger sets for better comparison metrics)
*amounts rounded.

AGENDA

Introduction

Data Summary

Fitting Training Set

Fitting Validation Set

Best Model

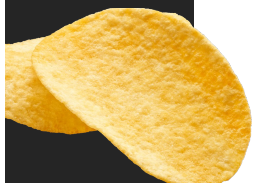
Business Implications

Conclusion

UCIrvine | The Paul Merage School of Business

CONCLUSION

- Implementation leads to ***cost savings*** and ***revenue opportunities***
- Potential monthly savings of **$120,419,719** from reduced customer complaints alone
- Improved **customer satisfaction** and enhanced **product quality**
- Streamlined detection process for operational efficiency
- **Minimized waste** and **lower production costs**
- **Increased profitability** and c**ompetitive edge** in the market



Thank you!

APPENDI

X

Cost Benefit Analysis

PART 2 A: Manual Detection

To assess the cost of manual chip detection, we multiplied the cost of manual inspection per chip by the total number of chips. The number of undetected defective chips was obtained using the model accuracy. We then calculated the cost of complaints for undetected defective chips using the average number of complaints per defective chip with cost. The total cost was determined by adding the cost of manual detection and the cost of complaints for undetected defective chips. By subtracting the total cost of manual detection from the cost of complaints assuming no detection model, we obtained the potential savings from manual detection.

PART 2 B: Model Detection

Similar to manual detection, we computed the cost of chip detection using our deep learning model. This involved multiplying the cost of model inspection per chip by the total number of chips. The number of undetected defective chips and cost of complaints was derived using the same formula as in manual detection. The total cost of model detection was obtained by summing the cost of model detection and the cost of complaints for undetected defective chips. By subtracting the total cost of model detection from the cost of complaints assuming no detection model, we determined the potential savings from model detection.