Ensemble Learning with Computer Version and Natural Language Processing: Multi-Class Prediction of Retail Products

CV & NLP for Retail

Application of convolutional neural networks, word embedding, and classical machine learning algorithms to enhance predictive performance

Sean X. Zhang, MSc

York University, seanxzhang94@gmail.com, ORCID: 0000-0003-3356-4243

Background: Online retail is a lucrative and rapidly expanding industry. There is a need for AI-based solutions to classify products to facilitate both retailer sales and consumer satisfaction. We present an ensemble learning method involving computer vision and natural language deep neural networks to improve predictive performance. **Methods:** 52,567 unique products with a title, description, and 100x100 RGB image, split into 21 categories were downloaded from Kaggle. Models were trained on both image and text data. The final model was an ensemble between various CV and NLP models. **Results:** The mean F1 score of the highest performing CV and NLP models were 0.37 and 0.72, respectively. The mean F1 of the ensemble model was 0.76. Model performance on unlabeled test data was 0.76. **Conclusion:** Combining CV and NLP models improves accuracy while remaining generalizable in multi-class retail products.

CCS CONCEPTS • Applied computing• Computing methodologies

Additional Keywords and Phrases: image classification, natural language processing, ensemble learning, retail

1 INTRODUCTION

In 2020, the lucrative and rapidly-expanding e-commerce industry generated $861 billion US, having increasing 44% in revenue compared to the previous year [1]. Online commerce platforms such as Amazon, Shopify, and Wish take advantage of AI-based recommender systems to sell their products. These online commodities need to be organized by group to facilitate consumer queries. Since millions of unique products are sold online, identifying similar products through manual classification is simply unfeasible. Hard-coded rules are also not flexible enough to account for the sheer diversity of online goods. As such, there is a need for an AI solution to classify products.

Convolutional neural networks have made great breakthroughs in image classification. ImageNet [2], a large online image repository with millions of images and their labels, has borne several state of the art CNN architectures, with many models attaining more than 85% accuracy [3]. As such, these neural networks can also be used to classify retail images.

Performance in natural language processing has also been augmented with the advent of word embeddings such as Word2Vec and GloVe [4], [5]. These allowed for training of extremely large corpuses with billions of unique tokens. Previously, this would have been computationally expensive as unique words were represented as individual dimensions.

However, there has been little research on whether image classification and natural language processing could be combined to improve predictive accuracy. In this project, we demonstrate that learning combining computer vision and natural language processing increases overall performance.

2 IMPLEMENTATION AND RESULTS

The project was implemented with Python 3 in Google Colab Pro. Models were assembled using either *Tensorflow/Keras* or *scikit-learn* and trained with NVIDIA T4 or P100 GPUs.

2.1 Data Description

The retail products dataset was accessed from a Kaggle competition [6]. Each product had a unique identifier, title, short description, a 100 x 100 image, as well as the category to which it belonged to (Table 1). The training dataset contained 52,567 unique products with 21 different retail categories, such as: Electronics, Beauty, Office Products, Baby Products, etc. The classes were balanced, with 2500 images for each category except Electronics, which had 2596. An unlabeled test set of 6367 products was also provided.

Table 1: Data Dictionary

Variable	**Definition**
ImgId	Unique identifier for product
title	Name of the product
description	Brief description of the product
category	Type of retail category

2.2 Data Preprocessing

The dataset contained both image and text features, and as such each were preprocessed independently.

2.2.1 Image Preprocessing

First, 1157 (2.20%) of images were removed, as they did not have a RGB channel (the image array shape was (100, 100) rather than (100, 100, 3)). Removing them did not cause class imbalance as every category remained above 2300 observations.

2.2.2 Text Preprocessing

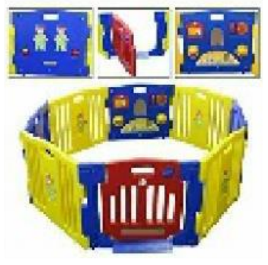
The description texts were converted to lower-case and additional blank spaces were stripped. Non-text, such as symbols and numbers, and stop words [7] were removed. Instances of ‘oz’ were converted to ‘ounce’, ‘S’ to ‘small’, ‘M’ to ‘medium’, and ‘L’ to ‘large’ to account for common abbreviations.

2.3 Exploratory Data Analysis

2.3.1 Images Exploratory Analysis

Image samples of each class were randomly generated. Images were then classified with VGG16 [8] and ResNet50 [9], two pretrained deep convolutional neural networks, to determine baseline data quality. Overall, VGG16 and Resnet50 reasonably identified most images, though some were misidentified (Figure 1).

Figure 1: Example Image Identification with VGG16 and ResNet50

Accurate Identification	**Reasonable Identification**	**Incorrect Identification**
		
Baby Playpen *Baby Products*	**Digital Cordless Phone** *Office Products*	**Sunglasses** *Beauty*
Crib (0.71), *Cradle* (0.12)	*Radio* (0.32), *Modem* (0.32)	*Whistle (0.58), Bib (0.30)*

2.3.2 Text EDA

The cleaned text data were split by category, tokenized, and counted for each time the token appeared. Tokens related to sizing and measurements, such as *ounce*, *inch*, and *small* were shared among several categories. However, the top words for many categories proved an apt descriptor (Table 2).

Table 2: Word counts by selected categories

Category	**Word Count (Top 3)**
Electronics	{*digital*: 1538, *video*: 1172, *cable*: 1123}
Cell Phones & Accessories	{*phone*: 5085, *case*: 1905, *battery*: 1247}
Automotive	{*vehicle*: 655, *easy*: 474, *designed*: 395}
Toys & Games	{*game*: 1687, *new*: 1250, *cards*: 1173}
Tools & Home Improvement	{*inch*: 2149, blade: 1157, *saw*: 813}
Beauty	{*skin*: 2558, *hair*: 1518, *oil*: 786}
Grocery & Gourmet Food	{*flavor*: 726, *tea*: 540, *coffee*: 514}

2.4 Feature Extraction and Modeling

The data was split into 70% train (35987 rows) and 30% test (15423 rows) sets. Models for image and text classification were then trained independently. The target variable was one-hot encoded.

2.4.1 Deep Neural Network Layers

Several deep learning layers were used in this project. For deep neural networks in image classification, the *Conv2D* layer contained small 3x3 kernels to convolve the image, detecting salient features. *Batch normalization* then scales the output The *Maxpool2D* layer reshapes the original input, lowering the resolution for generalizability. *Dropout* layers freeze subsets of neurons to prevent overfitting. Finally, the *Dense* layer outputs the final prediction. For the NLP deep learning models, the *Embedding* layer was used to represent text features as a low-dimensional dense matrix (as opposed to another feature representation of text as a highdimensional sparse matrix, where each word as a feature). A one-dimensional convolution layer, *Conv1D* was also applied. While convolutional has typically been applied to computer vision tasks, they can also be useful in NLP problems. While convolution may reduce the ability to keep word order, they nevertheless can recognize patterns in sentences. Recurrent neural networks such as *LSTM* were also tested, but surprisingly did not perform well enough to be included in the final evaluation. The layers were imported from the *Keras* library [10].

2.4.2 Image Models

The following augmentation parameters were applied to the training set: 10-degree rotation, 0.1 zoom, 0.2 width and height shift, and horizontal flipping. Several convolutional neural networks were trained and only models with maximum validation accuracy (plateau for 3 or more epochs) double that of random chance (2 x 1/21, or 0.095) were chosen for further evaluation. Two custom model architectures were accepted: both custom models received an input shape of (100, 100, 3) and had a softmax output representing 21 different categories. The loss function was categorical cross entropy and was optimized by the Adam optimizer [11]. Model 1 utilized three convolutional–max pooling–dropout designs before flattening the input to a dense output layer (Table 3: IMG1). Model 2 was based off the Xception [12] neural network, which utilizes *depthwise separatable convolutions (SeparableConv)* layers: these perform depthwise spatial convolution, convolving each RGB channel separately (Table 3: IMG2).

Two pretrained models, VGG16 (Table 3: VGG) and ResNet50 (Table 3: ResNet) were also used for transfer learning, as exploratory tests with both pretrained models showed that they could reasonably interpret features in the Kaggle dataset. For use in pretrained models, the images were first preprocessed to model specifications. For both models, the images were converted from RGB to BGR, then zero-centered with respect to the ImageNet dataset, without scaling [8][9].

2.4.3 NLP Models

The text features were trained on both classical machine learning models and deep neural networks. For the classical model inputs, the text was tokenized and each word was represented as a 100-dimensional vector based on Word2vec [4], a deep learning technique which represents words as vectors based on proximity to other words. The word vectors from each retail product description were then averaged to create a ‘document vector’; thus, the final training set was tabular and could easily be used for classical ML models. Two tree-based models, Random Forest [15] (Table 3: RF) and XGBoost [16] (Table 3: XGB) were then trained on the data.

Two simple deep learning neural network architectures were also trained. The first model architecture had an Embedding layer, followed by a Flatten and Dense output layer (Table 3: NLP1). The second model architecture had an Embedding layer, followed by a Conv1D, Global Max Pooling 1D, and Dense output layer (Table 3: NLP2). A another pretrained embedding, GloVe [5], was compared to an Embedding layer trained solely on the retail product descriptions for each model (Table 3: GLV suffix).

2.5 Model Evaluation

2.5.1 Evaluation Metrics

The overall model performance was evaluated based on the mean F1 score. The F1 score measures accuracy based on an average of precision (ratio of true positives to predicted positives) and recall (the ratio of true positives to all actual positives). The mean F1 score pertains to the average of each F1 score by product category.

2.5.2 Model Performance

Four image models (two custom architectures and transfer learning with VGG16 and ResNet50) and six NLP models (Random Forest, XGBoost, two custom models with both a GloVe and untrained embedding layer) were evaluated. The NLP models greatly outperformed image models. The best performing image model was the transfer learning from ResNet50 (mean F1 = 0.37), while the best performing NLP model was a custom-trained Embedding layer without convolution. For image models, the highest-performing categories were: Grocery & Gourmet Food (F1=0.57), Cell Phones & Accessories (F1 = 0.55), and Clothing, Shoes & Jewelry (F1 = 0.53). For NLP, the highest-performing categories were: Grocery & Gourmet Food (F1 = 0.89), Appliances (F1 = 0.85), and Cell Phones & Accessories (F1 = 0.85). There was a small positive correlation between the mean image model F1 and mean NLP model F1 per category ($R^2 = 0.45$). It is unclear whether this was due to differences in data quality between various categories.

2.5.3 Ensemble Modeling

Two ways to ensemble models were included in the final evaluation: 1) average of the predicted probabilities of each class – ensemble averaging; and 2) mode of predicted class (category with highest probability) – ensemble voting. The models used to obtain the highest mean F1 of 0.76 used ensemble averaging (Table 3: ENS AVG), with the models: RF, XGB, NLP1, NLP2, and ResNet. The models resulting in the highest possible F1 using ensemble voting were: RF, NLP1, NLP2, and ResNet with mean F1 = 0.75. Removing poorer-performing models, even the ResNet model (F1 = 0.37) led to a decrease in ensemble F1. This suggests that assembling different types of model architectures increases the generalizability of the predictions.

2.5.4 Kaggle Submission

Each component of the averaging ensemble model was then trained on the full training dataset (51410 rows) and used to predict on an unlabeled test set provided by Kaggle (6367 rows). The mean F1 score was 0.76481, which was consistent the F1 score calculated only with the given train set.

3 DISCUSSION

This project represents a successful implementation of image classification and natural language processing for multi-class retail products prediction, having created models that performed significantly above chance. Some categories in the Kaggle dataset were similar (e.g., All Electronics vs. Electronics, Baby vs. Baby Products, All Beauty vs Beauty). To improve performance, a stacked ensemble model could have been employed, where the output of a multi-classifier could then be fed into a binary classifier trained using transfer learning on the similar classes. Additionally, for poor-performing categories, further data exploration could have been conducted, such as a deep dive into removing problematic or non-sensical words. Finally, the title of the product was not incorporated into any NLP models. It could have been combined with the description or used as its own feature.

	Image Models				*NLP Models*						*Ensemble Models*	
	IMG1	**IMG2**	**VGG**	**ResNet**	**RF**	**XGB**	**NLP1**	**NLP1 GLV**	**NLP2**	**NLP2 GLV**	**ENS AVG**	**ENS MODE**
All Beauty	0.01	0.11	0.35	0.35	0.61	0.53	0.60	0.50	0.57	0.50	0.66	0.63
All Electronics	0.00	0.09	0.17	0.22	0.53	0.46	0.56	0.47	0.55	0.49	0.60	0.57
Appliances	0.14	0.27	0.31	0.38	0.82	0.79	0.85	0.81	0.83	0.75	0.87	0.83
Arts, Crafts & Sewing	0.02	0.11	0.18	0.27	0.71	0.67	0.75	0.70	0.76	0.64	0.80	0.75
Automotive	0.15	0.22	0.28	0.38	0.65	0.61	0.67	0.60	0.61	0.55	0.70	0.73
Baby	0.15	0.29	0.39	0.48	0.72	0.67	0.73	0.66	0.71	0.61	0.76	0.75
Baby Products	0.13	0.20	0.22	0.32	0.61	0.53	0.63	0.55	0.65	0.52	0.67	0.65
Beauty	0.01	0.28	0.15	0.32	0.61	0.52	0.60	0.51	0.61	0.50	0.64	0.62
Cell Phones & Accessories	0.20	0.28	0.51	0.55	0.85	0.80	0.85	0.82	0.86	0.83	0.88	0.87
Clothing, Shoes & Jewelry	0.22	0.39	0.47	0.53	0.73	0.66	0.76	0.68	0.73	0.65	0.80	0.79
Electronics	0.26	0.29	0.42	0.43	0.76	0.69	0.78	0.66	0.78	0.68	0.81	0.81
Grocery & Gourmet Food	0.32	0.41	0.49	0.57	0.89	0.87	0.88	0.86	0.88	0.84	0.91	0.90
Health & Personal Care	0.24	0.34	0.34	0.38	0.62	0.56	0.62	0.53	0.60	0.50	0.67	0.66
Industrial & Scientific	0.00	0.10	0.12	0.27	0.48	0.40	0.51	0.41	0.56	0.43	0.63	0.57
Musical Instruments	0.07	0.31	0.34	0.37	0.81	0.79	0.82	0.76	0.81	0.70	0.84	0.85
Office Products	0.15	0.32	0.35	0.41	0.82	0.78	0.83	0.77	0.81	0.73	0.84	0.85
Patio, Lawn & Garden	0.07	0.11	0.25	0.29	0.72	0.66	0.76	0.67	0.78	0.64	0.81	0.81
Pet Supplies	0.05	0.21	0.23	0.25	0.79	0.72	0.83	0.76	0.84	0.74	0.87	0.86
Sports & Outdoors	0.00	0.03	0.12	0.19	0.54	0.50	0.57	0.46	0.58	0.43	0.64	0.61
Tools & Home Improvement	0.02	0.20	0.29	0.33	0.73	0.64	0.76	0.66	0.75	0.65	0.79	0.79
Toys & Games	0.20	0.41	0.43	0.51	0.80	0.77	0.82	0.78	0.83	0.76	0.84	0.85
Mean F1	**0.12**	**0.24**	**0.31**	**0.37**	**0.71**	**0.65**	**0.72**	**0.65**	**0.72**	**0.63**	**0.76**	**0.75**

Table 3: F1 Scores by Model and Category

Highest F1 scores across models are highlighted

4 CONCLUSION

Computer vision and natural language processing can be combined to enhance predictive power in datasets where both features are available. An ensemble model increased the mean F1 score to 0.76 from 0.72.

ACKNOWLEDGMENTS

The author thanks Karthik Kuber, PhD for his advisory in the York University Machine Learning at Scale course.

REFERENCES

[1] "US ecommerce grows 44.0% in 2020 | Digital Commerce 360." https://www.digitalcommerce360.com/article/us-ecommerce-sales/ (accessed Mar. 18, 2021).
[2] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.
[3] "Papers with Code - ImageNet Benchmark (Image Classification)." https://paperswithcode.com/sota/image-classification-on-imagenet (accessed Mar. 18, 2021).
[4] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," Jan. 2013, Accessed: Mar. 17, 2021. [Online]. Available: https://arxiv.org/abs/1301.3781v3.
[5] "GloVe: Global Vectors for Word Representation." https://nlp.stanford.edu/projects/glove/ (accessed Mar. 17, 2021).
[6] "Retail Products Classification." https://kaggle.com/c/retail-products-classification (accessed Mar. 16, 2021).
[7] "explosion/spaCy," *GitHub*. https://github.com/explosion/spaCy (accessed Mar. 16, 2021).
[8] K. Team, "Keras documentation: VGG16 and VGG19." https://keras.io/api/applications/vgg/ (accessed Mar. 16, 2021).
[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Dec. 2015, Accessed: Mar. 17, 2021. [Online]. Available: https://arxiv.org/abs/1512.03385v1.
[10] K. Team, "Keras documentation: Keras layers API." https://keras.io/api/layers/ (accessed Mar. 17, 2021).
[11] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *ArXiv14126980 Cs*, Jan. 2017, Accessed: Mar. 17, 2021. [Online]. Available: http://arxiv.org/abs/1412.6980.
[12] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," Oct. 2016, Accessed: Mar. 17, 2021. [Online]. Available: https://arxiv.org/abs/1610.02357v3.
[13] "tf.keras.applications.vgg16.preprocess_input | TensorFlow Core v2.4.1," *TensorFlow*. https://www.tensorflow.org/api_docs/python/tf/keras/applications/vgg16/preprocess_input (accessed Mar. 17, 2021).
[14] "tf.keras.applications.resnet.preprocess_input | TensorFlow Core v2.4.1," *TensorFlow*. https://www.tensorflow.org/api_docs/python/tf/keras/applications/resnet/preprocess_input (accessed Mar. 17, 2021).
[15] "sklearn.ensemble.RandomForestClassifier — scikit-learn 0.24.1 documentation." https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (accessed Mar. 17, 2021).
[16] "Python Package Introduction — xgboost 1.4.0-SNAPSHOT documentation." https://xgboost.readthedocs.io/en/latest/python/python_intro.html (accessed Mar. 17, 2021).