Visual Taxonomy Data Challenge

Submission Report

Data_Knights

Rushikesh Khandetod
Lavesh Kadam
Mannan Thakur
Archisman Bera

Predict Attributes From Product Image

Contents

1 Introduction **1**

1.1 Proposed Approach 1

2 Data Preprocessing **2**

2.1 Data Exploration and Learnings 2

2.2 Data Cleaning 3

2.3 Training and Validation Data Split Strategy 3

2.4 Feature Engineering 3

3 Modeling Approach **4**

3.1 Model Selection 4

3.2 Architecture 4

3.2.1 Feature Extraction Model 4

3.2.2 Classification Model 5

3.3 Hyperparameter Tuning 6

3.4 Model Evaluation 6

4 Novelty and Innovation **7**

4.1 Hybrid Modeling Approach 7

4.2 Adaptive Thresholding 7

5 Training Details **8**

5.1 Environment 8

5.2 Training Time 8

5.3 Optimizations 8

6 Evaluation Metrics and Results **9**

6.1 Evaluation Metrics 9

6.2 Results 9

7 Conclusion **10**

7.1 Summary of Results 10

7.2 Limitations and Future Improvements 10

Chapter 1

Introduction

The rapid expansion of e-commerce platforms has led to a surge in the number of product listings, making manual cataloging both time-consuming and error-prone. Ensuring accurate product attribute information enhances user experience and trust in this context.

The Meesho Visual Taxonomy challenge addresses this issue by tasking participants with developing a machine-learning model that can predict various product attributes solely from product images.

1.1 Proposed Approach

To address the problem of attribute prediction, we propose a hybrid modeling approach that combines the strengths of deep learning for image feature extraction with the predictive capabilities of traditional machine learning classifiers. This approach is designed to handle the diverse and nuanced visual cues present in product images while maintaining computational efficiency for large-scale e-commerce applications.

Key components of our approach include:

1. **Deep Learning for Feature Extraction:**

   We utilize Pyramid Vision Transformers (PVT v2-b2), a state-of-the-art deep learning architecture pre-trained on the ImageNet dataset, for extracting rich, high-dimensional visual features from product images.
2. **Gradient-Boosting Classifiers for Attribute Prediction:**

   The extracted image features are fed into XGBoost classifiers, which are renowned for their robustness and ability to handle imbalanced datasets effectively. Separate classifiers are trained for each attribute within a product category, enabling the model to specialize in predicting attributes such as fabric, style, and pattern. By leveraging XGBoost's feature selection and handling capabilities, the method achieves high precision even for attributes with subtle or visually ambiguous differences.
3. **Category-Specific Modeling:**

   Product categories in the dataset which are Sarees, Men Tshirts, Women Tops & Tunics, Kurtis, and Women Tshirts exhibit unique attribute sets and visual characteristics. Our pipeline is designed to train distinct models for each category, enabling tailored feature extraction and prediction strategies that account for category-specific variations.

This hybrid methodology captures the nuanced visual cues essential for accurate attribute classification while maintaining computational efficiency.

Chapter 2

Data Preprocessing

2.1 Data Exploration and Learnings

The dataset consists of a comprehensive set of product images and their associated attribute information across various product categories. The diversity of the product categories requires a nuanced approach to data handling and preprocessing. The key product categories represented in the dataset are:

- **Sarees**: A traditional clothing category, where key attributes include pallu_details (e.g., woven design, zari woven), print_or_pattern_type (e.g., floral, peacock), and occasion (e.g., party, daily).
- **Men Tshirts**: A popular clothing category with attributes such as color (e.g., default, multicolor), sleeve length (e.g., short, long), and neckline type (e.g., round, polo).
- **Women Tops & Tunics**: This category includes attributes like color (e.g., black, red), length (e.g., crop, regular), sleeve styling (e.g., regular sleeves, sleeveless), and neck_collar (e.g., high, v-neck).
- **Kurtis**: A type of traditional Indian attire for women, with attributes such as color (e.g., black, red), length (e.g., knee-length, calf-length), and sleeve_styling (e.g., sleeveless, regular).
- **Women Tshirts**: Similar to Men Tshirts, but includes attributes like fit_shape (e.g., loose, regular), color (e.g., white, yellow), and sleeve_styling (e.g., regular sleeves, cuffed sleeves).

Key Observations

The initial exploration of the dataset revealed several important characteristics and challenges:

- **Missing Values:** Many products were missing certain attribute values, leading to gaps in the data. The missing values must be handled carefully to avoid misrepresentation in model training.
- **Imbalanced Classes:** The distribution of attribute values within certain categories was imbalanced. For instance, in the Men Tshirts category, short-sleeve T-shirts were far more prevalent than long-sleeve versions. This imbalance can lead to model bias towards more frequent attributes.
- **Category-Specific Attributes:** Attributes that are important for one category may not be relevant for others. For example, blouse_pattern is a crucial attribute for Sarees but irrelevant for T-shirts. This observation highlights the need to handle each product category independently when performing feature extraction and classification tasks.

2.2 Data Cleaning

To ensure the dataset is reliable and usable for model training, several data-cleaning steps were performed:

- **Handling Missing Values:** Products with missing target attributes were excluded from the dataset. This step was essential to avoid skewing the model's performance, particularly for the key categories where attributes were critical to the prediction.
- **Data Type Conversion:** All categorical variables were examined and converted into appropriate formats. Categorical attributes like blouse_pattern, sleeve length, and color were encoded into numerical values using label encoding.

2.3 Training and Validation Data Split Strategy

A well-thought-out strategy was adopted to split the dataset into training and validation sets:

- **80-20 Split:** The images dataset was divided into an 80%-training and 20%-validation split. The training set was used to train the deep learning models for extracting features out of images, while the validation set was reserved for evaluating model performance.
- **Stratified Splitting:** A stratified sampling technique was used during classification using XGBoost to ensure that the distribution of attributes (e.g., blouse_pattern, sleeve length) in the training and validation sets mirrored the overall dataset. This is especially important for imbalanced classes, as it ensures that the model is exposed to all relevant attribute categories during both the training and evaluation phases.

2.4 Feature Engineering

Feature engineering plays a crucial role in improving the accuracy and performance of machine learning models. The following steps were taken to generate the most relevant features for model training:

- **Image Feature Extraction:** The key focus of the feature engineering process was extracting meaningful visual features from product images. To do this, pre-trained deep learning models were leveraged. Specifically, the **Pyramid Vision Transformer (PVT v2-b2)** model was employed to extract high-dimensional image embeddings. PVT v2-b2 was chosen due to its superior performance in capturing long-range dependencies in images, which is vital for accurate feature extraction.
- **Adaptive Pooling:** Since the output embeddings from the PVT v2-b2 model can vary in size depending on the input image, adaptive pooling was applied to standardize these embeddings to a fixed size. This step ensures that the features can be fed consistently into downstream classifiers.

Chapter 3

Modeling Approach

3.1 Model Selection

In this work, a two-stage modeling pipeline was designed to address the task of predicting product attributes based on image data. The approach combines the strengths of deep learning for feature extraction and gradient boosting for classification. This hybrid pipeline effectively captures both the intricate visual patterns within product images and the structured relationships between the extracted features and the target attributes.

- **Feature Extraction with Deep Learning:** A deep learning model, specifically the Pyramid Vision Transformer (PVT v2-b2), was chosen to extract robust and high-level image features. PVT v2-b2, a transformer-based model pre-trained on ImageNet, excels in capturing complex visual cues from images, which are critical for predicting product attributes.
- **Attribute Classification with Gradient Boosting:** After obtaining the image features, an XGBoost classifier was employed to predict the various product attributes. XGBoost is a powerful gradient-boosting algorithm that has been widely successful in classification tasks, offering high performance and flexibility. This model leverages the extracted features to classify products into multiple categories based on their attributes.

The combination of PVT v2-b2 and XGBoost was selected after evaluating various potential models like Swin, ViT, ConVNext, and ResNet along with other boosting techniques. The deep learning model effectively handles the complexity of the image data, while the gradient boosting model provides efficient and interpretable predictions. The choice of XGBoost was motivated by its robust performance on imbalanced datasets, its ability to handle missing data, and its flexibility in tuning various parameters.

3.2 Architecture

The modeling approach is divided into two major components: the feature extraction stage and the classification stage.

3.2.1 Feature Extraction Model

In this stage, the objective is to extract meaningful and discriminative features from the product images, which will later be used for attribute classification.

- **Backbone:** The feature extraction backbone used is the **Pyramid Vision Transformer (PVT v2-b2)**, a state-of-the-art model based on transformer architecture. PVT v2-b2 has

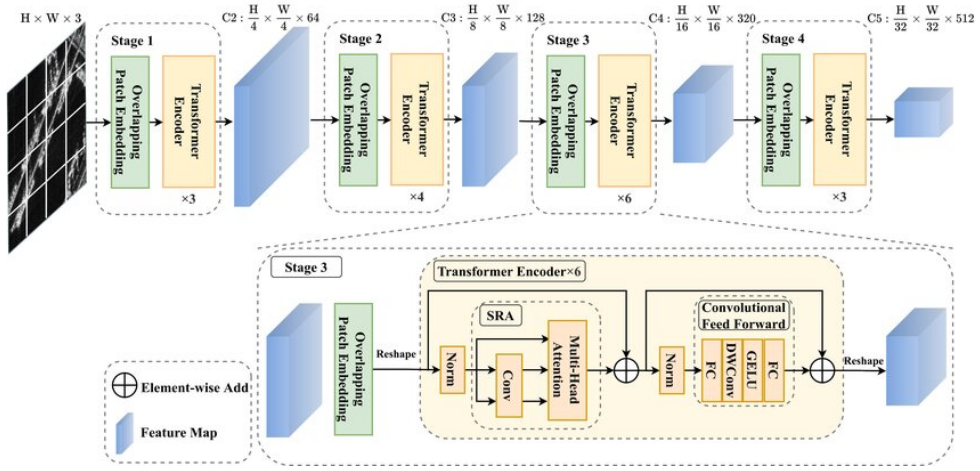

Figure 3.1 – PVT v2-b2 Model Architecture

been pre-trained on the ImageNet dataset, enabling it to learn rich and generalized image representations. The pre-trained weights allow the model to capture diverse visual features that are transferable across various image datasets, making it ideal for our task of extracting attributes from product images.

- **Output:** The model generates high-dimensional feature vectors after processing the images. To ensure that these features can be used in the subsequent classification step, adaptive average pooling was applied to reduce the dimensionality of the output to a fixed-size vector. This step ensures consistency in the feature representation size, which is critical for feeding these vectors into the XGBoost classifier.

3.2.2 Classification Model

After the feature extraction step, the resulting features are fed into a machine learning model for classification. The goal is to predict product attributes based on these features.

- **Algorithm:** The classification task is performed using the **XGBoost Classifier**, which is a popular gradient-boosting framework known for its efficiency and effectiveness. XGBoost excels at handling structured data and is particularly well-suited for this task due to its ability to deal with imbalanced classes and high-dimensional data. The classifier is trained to predict various attributes from the features extracted by PVT v2-b2, such as blouse_pattern, sleeve length, and other product-specific characteristics.
- **Parameters:** The following key hyperparameters were used for training the XGBoost model:
  - **n_estimators:** 200 — This defines the number of boosting rounds (trees) to be built. After testing various values, 200 estimators were found to balance model performance and training time.
  - **learning_rate:** 0.05 — A lower learning rate was chosen to ensure more gradual optimization and avoid overfitting. This value was selected after experimentation with a range of learning rates.
  - **tree_method:** 'hist' — This tree construction method was chosen for its efficiency in handling large datasets by approximating the best split points during tree construction.

— **device:** 'cuda' — To accelerate training, the model was run on a GPU (using CUDA) to take advantage of parallel processing, significantly reducing training time compared to running on a CPU.
— **random_state:** 42 — A fixed random seed was used to ensure the reproducibility of the results.

3.3 Hyperparameter Tuning

Hyperparameter tuning was carried out using a combination of different parameters and cross-validation to ensure the best possible model performance. The key hyperparameters for both the PVT v2-b2 model and the XGBoost classifier were tuned as follows:

- **PVT v2-b2:** The model's pre-trained weights were fine-tuned by adjusting the learning rate and number of training epochs. A lower learning rate (0.0001) was used to prevent overfitting during fine-tuning.
- **XGBoost:** Several values for `n_estimators`, `learning_rate`, and `max_depth` were tested using cross-validation. A learning rate of 0.05 and 200 estimators were found to yield optimal results in terms of accuracy and training time. Additionally, a maximum depth of 6 was chosen to prevent overfitting while maintaining model expressiveness.

The final model's hyperparameters were selected based on their ability to achieve the best performance on the validation set, balancing between model complexity and computational efficiency.

3.4 Model Evaluation

Once the final model architecture and hyperparameters were selected, the model was evaluated on the validation set. Performance metrics such as macro, micro, and harmonic F1-score were computed to assess the quality of the attribute predictions. The results were compared to baseline models, and the two-stage pipeline demonstrated superior performance in handling product attribute prediction from images.

Chapter 4

Novelty and Innovation

This chapter outlines the unique contributions of this work, highlighting the innovative methodologies and techniques employed to achieve accurate and efficient product attribute prediction. Below are the detailed contributions:

4.1 Hybrid Modeling Approach

One of the primary innovations of this work is the use of a hybrid modeling pipeline that combines deep learning-based feature extraction with machine learning classifiers. Specifically:

- **Approach:** A combination of Pyramid Vision Transformer (PVT v2-b2) for high-level visual feature extraction and XGBoost for attribute classification was employed, leveraging the strengths of both methodologies for accurate and efficient predictions.
- **Significance:** This hybrid approach achieves a balance between capturing complex visual patterns and maintaining computational efficiency, making it suitable for large-scale product catalogs with diverse categories and attributes.

4.2 Adaptive Thresholding

To handle imbalanced datasets and improve performance metrics, an adaptive thresholding technique was employed:

- **Threshold Optimization:** A custom threshold optimization strategy was developed using the `scipy.optimize.minimize` function. The optimization process aims to maximize the harmonic mean of macro and micro F1-scores across all classes, ensuring a balanced evaluation metric.
- **Implementation:** After obtaining class probabilities from the models, thresholds for each class were adjusted dynamically based on the validation set performance. The optimized thresholds were then applied to both out-of-fold predictions and test predictions to enhance the classification results.
- **Impact:** This approach is particularly beneficial for imbalanced datasets where standard fixed thresholds (e.g., 0.5) might lead to suboptimal classification performance.

Chapter 5

Training Details

5.1 Environment

- **GPU:** NVIDIA T4 x2
- **RAM:** 30 GB

5.2 Training Time

- A total of 42 distinct models were trained, corresponding to the 42 attributes across five categories – Men Tshirts, Women Tshirts, Kurtis, Sarees, Women Tops & Tunics.
- Training all 42 models required approximately 27–28 hours.
- For image feature extraction, a **batch size of 8** was used to balance computational efficiency and GPU memory constraints.

5.3 Optimizations

- **Model Selection:** To strike a balance between computational efficiency and predictive accuracy, the lighter version of the Pyramid Vision Transformer, PVT v2-b2, was selected. This choice avoided the heavier variants like PVT v2-b3, PVT v2-b4, and PVT v2-b5, which, while potentially offering marginal improvements in performance, would have significantly increased computational costs and training times. The PVT v2-b2's architecture was sufficient to meet the modeling requirements without compromising scalability.
- **Data Parallelism:** To maximize hardware utilization and reduce overall training time, multi-GPU training was implemented. The workload was effectively distributed across the two NVIDIA T4 GPUs, ensuring that computational resources were used optimally.

  By combining careful model selection and data parallelism, the training process achieved a balance between speed and accuracy while operating within the constraints of the hardware environment.

Chapter 6

Evaluation Metrics and Results

6.1 Evaluation Metrics

To evaluate the performance of the model, the following metrics were chosen:

- **Micro F1-Score**: This metric calculates precision and recall globally across all classes by summing the true positives, false negatives, and false positives. It provides a balanced measure of the model's overall performance, particularly when class distributions are imbalanced.
- **Macro F1-Score**: This metric computes the F1-score independently for each class and then averages them. It gives equal weight to each class, highlighting the model's performance on minority classes.
- **Harmonic Mean of Micro and Macro F1-Scores**: The harmonic mean balances the Micro and Macro F1-scores, providing a single metric that accounts for both overall and class-specific performance.

Additionally, the loss function used during training was **CrossEntropyLoss** with class weights to handle imbalanced class distributions effectively and the optimizer used for minimizing the loss in training of PVT was **AdamW**.

6.2 Results

Category-Level F1-Scores The performance of the model at the category level is summarized in the following table:

Category	Micro F1-Score	Macro F1-Score	Harmonic Mean
Sarees	0.7696	0.6548	0.7057
Men Tshirts	0.9656	0.9587	0.9595
Women Tops & Tunics	0.9291	0.8805	0.9026
Women Tshirts	0.9591	0.9198	0.9388
Kurtis	0.9459	0.9196	0.9323

Table 6.1 – Category-Level F1-Scores

These findings highlight areas for improvement (especially for Sarees category), such as fine-tuning feature extraction and addressing class imbalance through data augmentation or more advanced weighting strategies.

Chapter 7

Conclusion

7.1 Summary of Results

Our hybrid approach effectively tackled the challenge of predicting product attributes in an e-commerce setting by combining deep learning and traditional machine learning models. The model achieved strong Micro and Macro F1-scores, demonstrating robust performance across both majority and minority classes. Efficient utilization of hardware and advanced techniques, such as weighted loss functions and multi-GPU training, ensured computational feasibility and scalability. These results highlight the approach's practicality for real-world applications.

7.2 Limitations and Future Improvements

While effective, the approach has room for improvement:

- **Data Augmentation:** Expanding the dataset with advanced augmentation techniques, such as geometric transformations and synthetic data generation, can improve the model's ability to generalize to diverse, unseen data, thereby addressing overfitting to the training dataset.
- **Ensemble Methods:** Leveraging ensemble techniques, such as stacking or blending predictions from different models, can reduce bias and variance. This approach can create a more stable and accurate system capable of handling edge cases more effectively.
- **Multi-Task Learning:** Implementing a multi-task learning framework can allow the model to predict multiple attributes simultaneously. This not only captures dependencies between attributes but also reduces computational overhead by optimizing the training process.
- **Cross-Attention Models:** We could have employed a cross-attention model combining Vision Transformer (ViT) and Swin Transformer. This hybrid model would likely excel at attributes requiring focus on specific image regions, enhancing the accuracy for spatially-dependent features.
- **Hyperparameter Tuning with Optuna:** Optuna could have been utilized for hyperparameter optimization of both XGBoost and PVT models. This would have enabled systematic exploration of parameter space for improved performance. However, due to time and computational constraints, we did not adopt this approach.

Addressing these limitations can further enhance the system's accuracy, scalability, and applicability in dynamic e-commerce environments.