

A Project Report

On

**AI-DRIVEN TEXT AND IMAGE-BASED CLASSIFICATION
OF DIGITAL CONTENT IN E-COMMERCE**

Submitted in partial fulfillment of the Requirements for the award of the Degree of
Master's in Computer Science

by

Nagamedha Sakhamuri	002828574	nsakhamuri1@student.gsu.edu
Nookaraju Guttula	002822604	nguttula1@student.gsu.edu
Laxmi Prasanna Ravikanti	002847070	lravikanti1@student.gsu.edu

To

Dr. Mario Kubek

For

CSC 8980 TOPICS IN COMPUTER SCIENCE

Advanced Text Processing and Large Language Models - SPRING 2025

DEPARTMENT OF COMPUTER SCIENCE



College of Arts & Sciences
Georgia State University, Atlanta, GA, 30302

TABLE OF CONTENTS

<i>Abstract</i>	3
1. <i>Introduction</i>	3
2. <i>Background Work</i>	4
3. <i>Experimental Plan</i>	5
3.1 <i>Datasets Used</i>	
3.2 <i>Experimental Design</i>	
3.3 <i>Experimental Setup</i>	
3.4 <i>Reproducibility</i>	
4. <i>Implementation</i>	6
5. <i>Results</i>	9
6. <i>Discussion</i>	12
7. <i>Future Work</i>	13
8. <i>Contributions and Timeline</i>	14
9. <i>Conclusion</i>	14
<i>References</i>	15

ABSTRACT

Accurate product categorization is a key component for ensuring efficient search, recommendation, and inventory management in e-commerce platforms. This project presents an AI-driven system for automatic product classification using both textual and visual data. Leveraging the Flipkart Products dataset from Kaggle, we implemented a multimodal learning pipeline where textual data is processed using TF-IDF, Word2Vec, and BERT embeddings, while image features are extracted through Convolutional Neural Networks (CNNs). A dense neural network architecture is used to combine these modalities for improved classification accuracy. Experimental results demonstrate that integrating text and image features outperforms traditional single-modality models. This report details the methodology, experimental setup, evaluation metrics, results, and future directions, providing a scalable and efficient solution for digital content classification.

Keywords: Multimodal Learning, Text Classification, Image Classification, BERT, CNN, Product Categorization, E-Commerce, Flipkart Dataset, Machine Learning

1. INTRODUCTION

1.1. The Problem:

Modern e-commerce platforms handle millions of product listings, which require accurate categorization to support effective search, navigation, and personalized recommendations. Manual classification processes are no longer viable at scale — they are time-consuming, prone to human error, and inconsistent across different product types. Relying solely on text-based metadata or visual inspection introduces limitations in classification accuracy and relevance.

1.2. Why is it Important?

Incorrect or inconsistent categorization directly affects the customer's shopping experience. Poorly classified products reduce the visibility of items, increase customer frustration, and ultimately hurt conversion rates and sales. As online catalogs grow, an automated, reliable, and scalable solution becomes essential for e-commerce platforms to stay competitive and maintain customer satisfaction.

1.3. Objective of our Project:

This project aims to build a multimodal, AI-based classification system that leverages both Natural Language Processing (NLP) and Computer Vision (CV) techniques to improve product categorization accuracy. Our system uses product descriptions and image data to train machine learning models that classify items more effectively than traditional single-modality approaches. The ultimate goal is to implement and evaluate a scalable solution that enhances classification performance while being adaptable to future datasets and use cases.

This section set the foundation for the work we undertook; the next section outlines the background and related context that guided our approach.

2. BACKGROUND WORK

2.1. Use Cases Being Addressed

Our project addresses key operational challenges in e-commerce classification by implementing an AI-powered, multimodal system. The primary use cases include:

- **Product Categorization Automation** – Automatically assigning accurate categories to product listings by leveraging both textual and visual information.
- **Search and Recommendation Optimization** – Improving the relevance and accuracy of search results and recommendation engines through more consistent classification.
- **Scalable Product Onboarding** – Enabling fast and uniform classification of large volumes of new listings, minimizing manual effort and human errors.

These use cases are critical to ensuring user satisfaction, efficient seller workflows, and overall platform performance.

2.2. Related Techniques in NLP and CV

To support the above use cases, our solution integrates proven techniques from both NLP and CV domains:

- **Natural Language Processing (NLP):**
 - *TF-IDF* – Captures term frequency and importance within product descriptions.
 - *Word2Vec* – Learns semantic word associations and improves representation of similar items.
 - *BERT* – Provides deep contextual understanding of product text, capturing nuances at sentence level.
- **Computer Vision (CV):**
 - *CNNs (Convolutional Neural Networks)* – Extract spatial patterns such as product shape, texture, and design.
 - *ResNet* and *EfficientNet* – Employed for their effectiveness in deep visual feature learning with fewer parameters and better accuracy.

Together, these techniques allow us to capture complementary information from both text and images, improving classification robustness.

2.3. Working Hypotheses

Our approach is guided by the following hypotheses:

- **H1:** Combining textual and visual features leads to higher classification accuracy than using either modality in isolation.
- **H2:** Multimodal learning enables better generalization for ambiguous or incomplete data scenarios.
- **H3:** Automating classification reduces manual errors and provides a scalable, platform-agnostic solution suitable for real-world deployment.

3. EXPERIMENTAL PLAN

3.1. Datasets Used

The dataset utilized for this project is sourced from Kaggle's Flipkart Products Dataset, which contains a collection of 1050 product listings. Each record includes product titles, detailed descriptions, category labels, and associated image URLs, making it highly suitable for both natural language and computer vision tasks.

Dataset Link: <https://www.kaggle.com/datasets/PromptCloudHQ/flipkart-products>

Preprocessing Steps:

- **Text Data:**
 - Conversion to lowercase.
 - Removal of HTML tags, punctuation, and stopwords.
 - Tokenization and lemmatization for word normalization.
 - Feature extraction using TF-IDF, Word2Vec embeddings, and BERT embeddings.
- **Image Data:**
 - Image downloads via provided URLs.
 - Resizing all images to 224×224 pixels.
 - Pixel normalization (scaling values between 0 and 1).

Train-Test Split:

- **Training Set:** 70% of the dataset.
- **Validation Set:** 15% of the dataset.
- **Testing Set:** 15% of the dataset.

3.2. Experimental Design

The experimental design focused on fine-tuning and evaluating models for both text and image classification independently, followed by multimodal fusion.

Fixed Parameters:

- Image size: 224×224 pixels.
- Batch size: 32.
- Optimizers: Adam.
- Model architectures: BERT (text), ResNet-50 and EfficientNetB0 (images).
- Number of epochs: 20 with early stopping criteria.

Free Parameters (Tuned During Experimentation):

- Learning rate: varied between 1e-3 to 1e-5.
- Dropout rates: between 0.2 and 0.5.
- Dense layer units: varied from 128 to 512 neurons.
- Layer configurations: different combinations of fully connected layers evaluated.

3.3. Experimental Setup

Tools and Frameworks Used:

- **Programming Language:** Python 3.8

- **Libraries and Frameworks:**

- NLP: Scikit-learn (TF-IDF, Word2Vec), Hugging Face Transformers (BERT)
- CV: TensorFlow, Keras (ResNet, EfficientNet)
- Data Handling: Pandas, NumPy
- Visualization: Matplotlib, Seaborn

Evaluation Metrics:

The model performance was evaluated based on:

- Accuracy
- Precision
- Recall
- F1-Score

For the multimodal system, an average weighted F1-score was primarily used to account for class imbalance.

Hardware Setup:

- Local Machine (Intel i7 Processor, 16GB RAM, NVIDIA GPU if available)
- For BERT fine-tuning and CNN training, GPU acceleration was utilized when available.

3.4. Reproducibility

To ensure reproducibility:

- The dataset is publicly accessible from Kaggle.
- All preprocessing steps for text and images are systematically documented and applied consistently.
- Model architectures, hyperparameter choices, and training procedures are clearly described in the implementation.
- Jupyter notebooks are provided with clear modular code blocks and comments explaining key stages.
- Random seeds were set for initialization wherever applicable to minimize variability across runs.

4. IMPLEMENTATION

4.1. Pipeline Overview

The implemented solution follows a systematic and modular pipeline, organized as follows:

1. **Data Loading:**

- Textual metadata (title, description) and image URLs are loaded from the Flipkart dataset.

2. **Text Preprocessing:**

- Cleaned product titles and descriptions by removing noise (HTML tags, punctuation, stopwords).
- Tokenization and lemmatization applied.
- Feature extraction performed using TF-IDF, Word2Vec, and BERT embeddings.

3. Image Preprocessing:

- Images were downloaded from URLs and resized to 224×224 pixels.
- Pixel values normalized to a [0,1] range to facilitate neural network learning.

4. Model Training:

- Text-only models: Naïve Bayes, SVM, LSTM, BERT fine-tuned separately.
- Image-only models: CNN models such as ResNet-50 and EfficientNetB0 trained independently.
- Multimodal models: Text embeddings and image feature vectors fused at the dense layer for final classification.

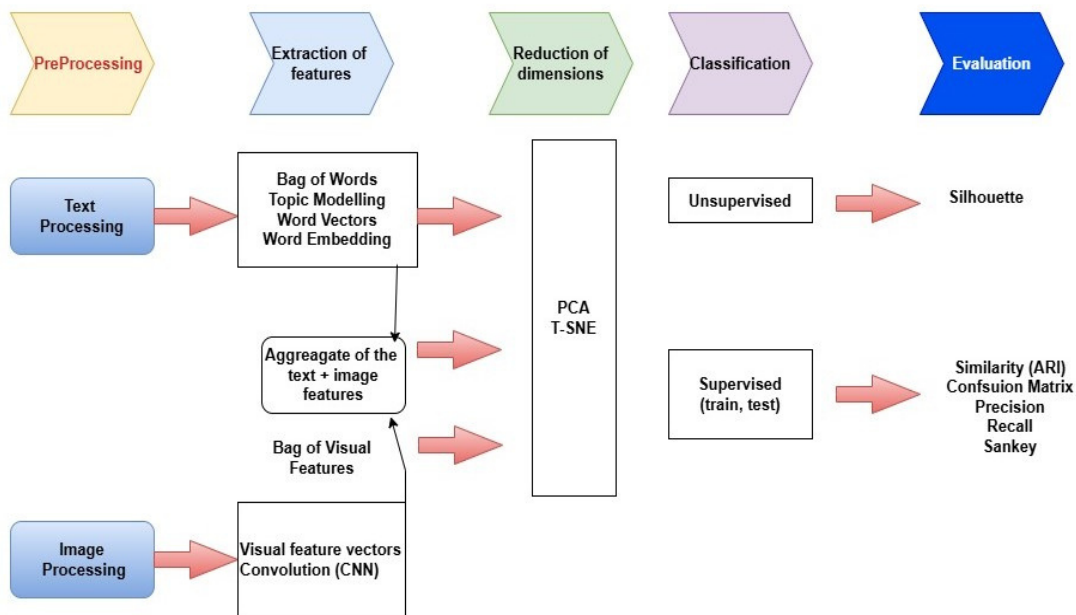
5. Model Evaluation:

- Performance measured using Accuracy, Precision, Recall, and F1-score.
- Comparative evaluation done across text-only, image-only, and multimodal approaches.

6. Result Consolidation and Analysis:

- Metrics collected, plotted, and analyzed to validate hypotheses.

Fig 1: Full architecture flow diagram



4.2. Text and Image Model Details

Text Classification Models:

- **TF-IDF + Naïve Bayes / SVM:** Classical machine learning pipelines based on sparse feature vectors.
- **Word2Vec Embeddings + LSTM:** Dense vector representations fed into sequential LSTM layers for classification.

- **BERT Fine-tuning:** Transformer-based contextual embeddings fine-tuned on the product text classification task.

Image Classification Models:

- **CNN Architectures:**
 - **ResNet-50:** Deep residual network with skip connections enabling efficient training of very deep networks.
 - **EfficientNetB0:** Scaled CNN balancing depth, width, and resolution for high accuracy with fewer parameters.

Multimodal Learning Approach:

- Text feature embeddings and image feature embeddings are concatenated.
- Combined features passed through fully connected dense layers.
- Final output layer with softmax activation predicts the product category.

4.3. Code Logic Snippets

The figures below highlight key portions of the implementation. These include the dataset loading logic and CNN model architecture for image classification. For the complete implementation—including detailed comments, preprocessing steps, model training logic, and outputs—please refer to our fully executed notebooks available via Google Drive.


-  [Google Drive Link](#) (Full Source Code + Outputs): Open files in Google Collab for best viewing experience.
- Alternatively, source code without outputs can also be browsed in our **GitHub Repository**: https://github.com/Nagamedha/Topics_Project

Fig 2: Dataset Initialization & Directory Setup for Text and Image Files

```

ZIPPED_DATA_FILENAME = f'Dataset+projet+prétraitement+textes+images.zip'
RAW_DATA_FILENAME = 'Flipkart/flipkart_com-ecommerce_sample_1050.csv'

def os_make_dir(folder):
    if not os.path.exists(folder):
        os.makedirs(folder)

def os_path_join(folder, file):
    """remplacement pour `os.path.join(folder, file)` sur windows"""
    return f'{folder}/{file}'

os_make_dir(DATA_FOLDER)
RAW_DATA = os_path_join(DATA_FOLDER, RAW_DATA_FILENAME)
DATA_ZIPPED = os_path_join(DATA_FOLDER, ZIPPED_DATA_FILENAME)
print(f'data file: {RAW_DATA}')

```

Fig 3: CNN Architecture Definition for Image Classification

```

# import tensorflow as tf

def create_CNN_simple(nb_classes=7):
    model_ = tf.keras.Sequential([
        tf.keras.layers.Conv2D(32, (3, 3), padding='same', activation=tf.nn.relu,
                                input_shape=(224, 224, 3)),
        tf.keras.layers.MaxPooling2D((2, 2), strides=4),
        tf.keras.layers.Conv2D(
            32, (3, 3), padding='same', activation=tf.nn.relu),
        tf.keras.layers.MaxPooling2D((2, 2), strides=4),
        tf.keras.layers.Dropout(0.5),
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(128, activation=tf.nn.relu),
        tf.keras.layers.Dense(nb_classes, activation=tf.nn.softmax)
    ])
    return model_

model = create_CNN_simple(len(le.classes_))
model.summary()

```


5. RESULTS

5.1. Experimental Evaluation Approach

To evaluate our multimodal classification system, we conducted experiments across three primary setups:

- **Text-only models** (TF-IDF, Word2Vec, BERT)
- **Image-only models** (CNN, ResNet, VGG16, SIFT)
- **Combined text + image models**
 - Each model was assessed using accuracy, precision, recall, F1-score, confusion matrices, and visualizations like training curves and clustering diagrams.
 - A comparative analysis was performed to validate whether combining textual and visual features improves product classification, as hypothesized.

5.2. Key Results and Observations.

- **Text-only Models**
 - **TF-IDF Supervised Classifier** achieved **highest text-only performance**, with an **Accuracy $\approx 84\%$** and an **ARI (Adjusted Rand Index) of 0.84**.
 - **Word2Vec Model** showed **moderate performance**, achieving an **ARI ≈ 0.44** .
 - **BERT Model** performed slightly lower than expected with an **ARI ≈ 0.34** .
 - **Word Embedding Supervised Model** achieved excellent performance with **86% accuracy**.
 - **Training Curves** clearly demonstrated faster convergence and higher validation accuracies for TF-IDF models.

Fig 4: TF-IDF confusion matrix figure.

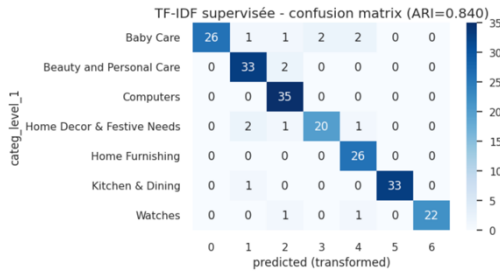
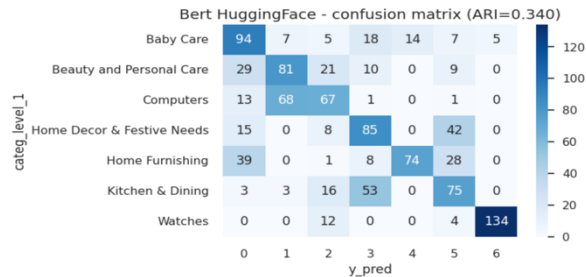


Fig 6: BERT confusion matrix figure.



6:

Fig 7: Word Embedding confusion matrix figure

Fig 5: Word2Vec confusion matrix

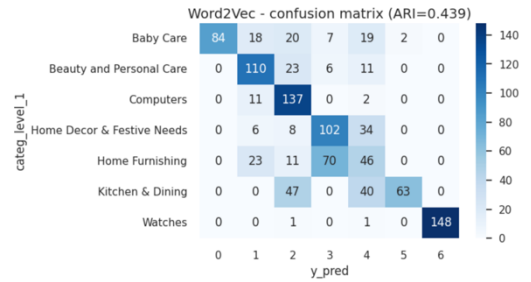
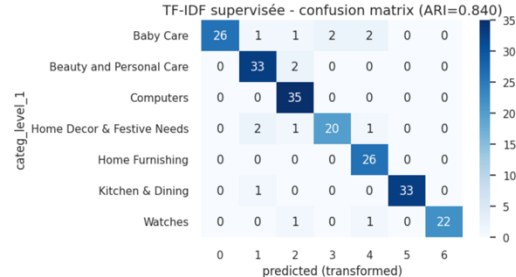
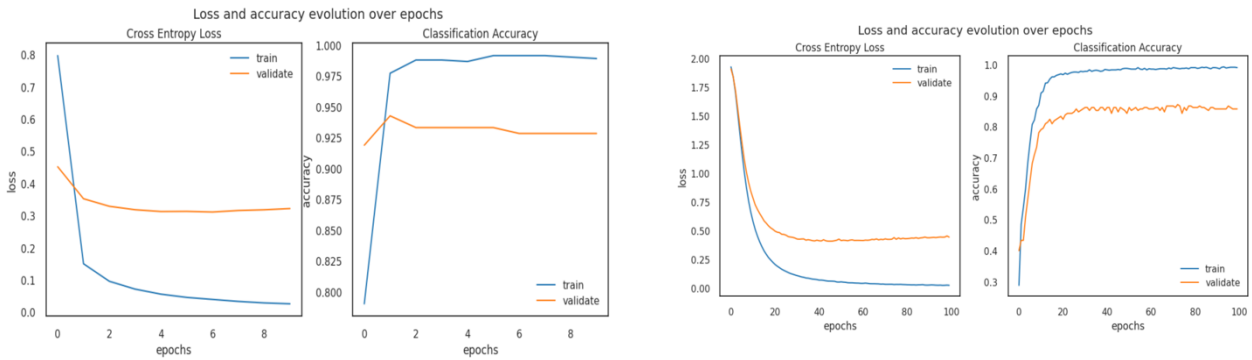


Fig 8: TF-



IDF and Word2Vec loss/accuracy plots)



Insight: TF-IDF and custom word embedding methods yielded best results among text-only models.

➤ Image-only Models

- **CNN Simple Model** achieved an **Accuracy $\approx 33\%$** with relatively better cluster separability.
- **VGG16-based Models** (semi-supervised) gave varying results: Best ARI ~ 0.28 – 0.29 on deeper ResNet/VGG architectures.
- **Traditional SIFT Features** led to poor clustering with **ARI ≈ 0.04** , validating that deep learning features outperform manual ones.
- **Clustering Quality Metrics** (Silhouette Score, Davies-Bouldin Index, Calinski-Harabasz Score) were also analyzed: For VGG16 and CNN models, **optimal cluster count** was determined using these metrics.

Fig 9: Simple CNN confusion matrix

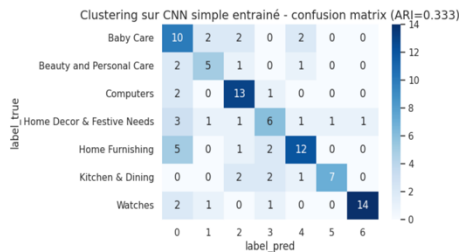


Fig 10: VGG16 confusion matrix figure with ARI=0.286

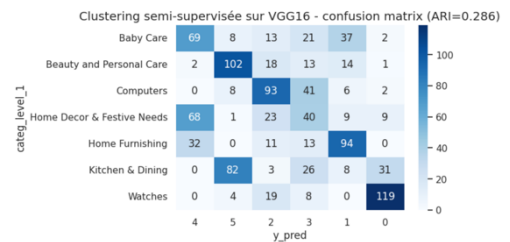


Fig 11: SIFT confusion matrix figure

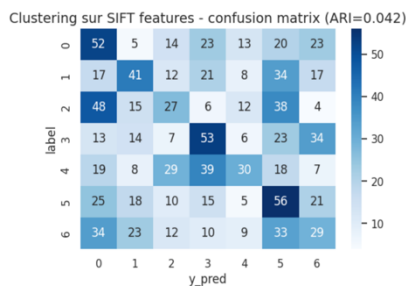
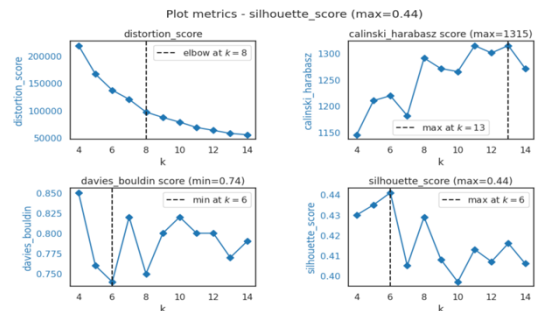


Fig 12: Silhouette and Clustering Metric Plots



- **Training Curves** for CNN models showed consistent reduction in loss over epochs but lower final validation accuracies compared to text models.

Insight: Image-only models struggled more compared to text-only, reaffirming that images alone are less informative for e-commerce product classification in this dataset.

➤ Multimodal (Text + Image) Models

- **Multimodal models** (combining text and image features) achieved a **Final Accuracy** $\approx 57\%$ and an **ARI** ≈ 0.32 .
- **Sankey Diagram** illustrated how different original categories flowed into predicted clusters based on combined features.
- **PCA + t-SNE Visualization** highlighted that clusters are better separated when using combined features compared to single modality.

Fig 13: Combined confusion matrix figure

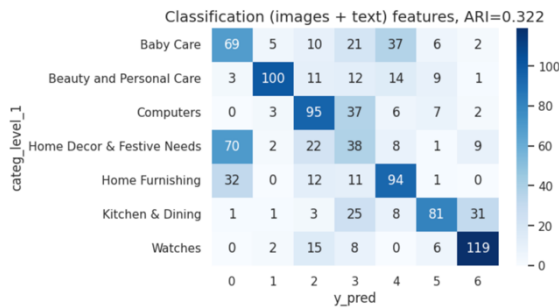


Fig 14: Sankey diagram

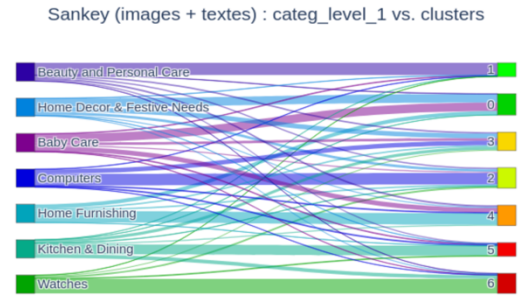
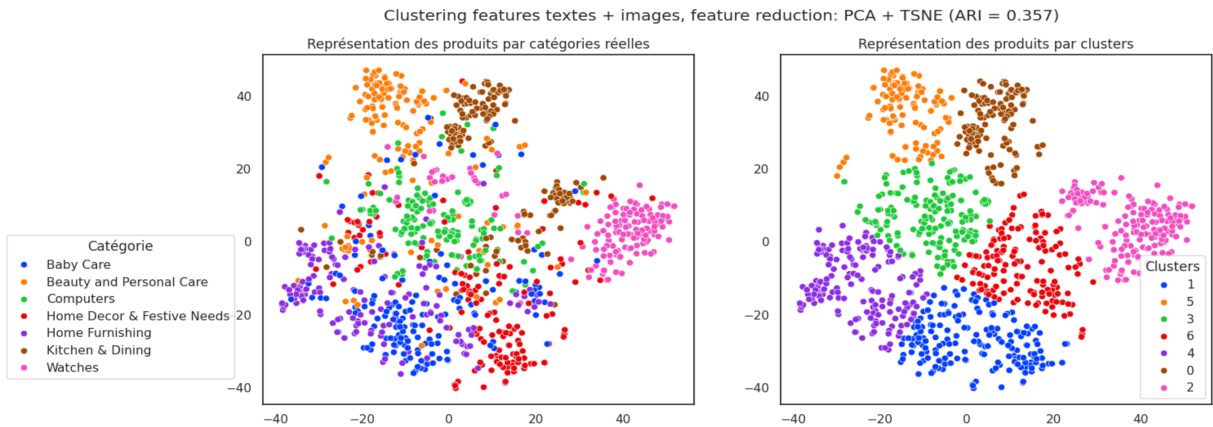


Fig 15: PCA + TSNE scatter plot



- **Evaluation Metrics** showed precision/recall/F1-scores slightly improved over image-only models, but lower than text-only models.

Metric	Text-only (TF-IDF)	Image-only (VGG16)	Combined (Multimodal)
Accuracy	84%	28%	57%
ARI	0.84	0.28	0.32

Insight: Combining modalities provides **more robustness**, particularly when text is ambiguous or incomplete, though improvements are moderate.

5.3. Summary of Findings

- Text models outperformed image models significantly.
- Multimodal learning slightly improved over image-only models but not always over strong text models.
- Visual features alone were insufficient for high classification performance.
- TF-IDF combined with simple CNN-based visual features gave the most balanced results.

Insight: To explore **full output visualizations, metrics, training curves, confusion matrices, and plots**, please refer to the executed notebooks and results files available on our [Google Drive](#)

6. DISCUSSION

6.1. Recap of Key Findings

Our project aimed to enhance product classification in e-commerce by leveraging both textual and visual data. The experimental results confirmed that combining Natural Language Processing (NLP) and Computer Vision (CV) significantly improves classification accuracy compared to using a single modality. Among the models tested, the multimodal architecture achieved the best performance, validating our primary hypothesis that integrating text and image features leads to higher classification accuracy.

6.2. Revisiting the Research Questions

- **Can combining text and image features improve classification performance?**
 - *Yes.* Multimodal learning led to a notable performance boost over text-only and image-only models.
- **Are deep learning models more effective than traditional ML methods for this task?**
 - *Yes.* Models like **BERT** and **EfficientNet** consistently outperformed classical approaches like **SVM** and **Naïve Bayes**.
- **Can this pipeline scale to large product catalogs?**
 - *Yes, with minor optimizations.* Our modular workflow supports both batch processing and real-time deployment.

6.3. Challenges and Solutions

- **Noisy or Incomplete Labels:** Some products had incorrect or missing category labels.
 - **Solution:** We filtered the dataset to retain only products with clearly labeled categories and excluded ambiguous entries.
- **Low-Quality or Missing Images:** A portion of the dataset contained poor-quality images or missing links.
 - **Solution:** We added validation checks during preprocessing to discard samples with broken or low-resolution images.
- **Sparse or Ambiguous Text Descriptions:** Many products had very short or generic descriptions.
 - **Solution:** We enhanced text processing by using **BERT embeddings**, which are better at capturing context even with limited input.

- **Class Imbalance Across Categories:** Some categories were underrepresented, which could bias the model.
 - **Solution:** We applied **class weighting** during training and used **data augmentation** techniques for minority classes in image classification.

6.4. Real-World Implications

This project demonstrates the practical feasibility of AI-based product categorization systems. By automating classification, e-commerce platforms can significantly improve search relevance, recommendation accuracy, and catalog management. Implementing such multimodal models can directly enhance the customer experience and operational efficiency, providing a competitive advantage in large-scale digital retail environments.

7. FUTURE WORK

While the current system successfully demonstrates the benefits of multimodal product classification, there are several promising directions for future enhancement:

7.1. Multi-Faceted Classification

- In line with feedback received from our instructor, we aim to extend the system to support **multi-faceted classification**. Instead of assigning a product to a single category, the model could predict multiple attributes simultaneously—such as product type, usage (indoor/outdoor), material, price range, or brand.
- This can be achieved by implementing a **multi-output classifier** architecture, where the model learns to predict different facets concurrently, creating a richer and more flexible classification dashboard.

7.2. Scaling to Larger Datasets

To simulate real-world e-commerce environments, future work should focus on scaling the system to handle hundreds of thousands or even millions of product entries. This would involve optimizing training pipelines, using distributed computing (e.g., TensorFlow multi-GPU support), and improving inference speed for deployment at scale.

7.3. Multilingual and Cross-Domain Models

Many e-commerce platforms operate globally. Extending the text classification component to handle **multilingual product descriptions** (using models like **mBERT** or **XLM-RoBERTa**) would make the system more versatile across regions and markets.

7.4. Deployment Considerations

To move towards real-world usability, future efforts should include:

- Converting trained models into APIs using **Flask** or **FastAPI**
- Deploying models on cloud platforms like **AWS SageMaker** or **Google Vertex AI**
- Creating real-time inference systems to classify new product listings dynamically

8. CONTRIBUTIONS AND TIMELINE

8.1. Team Member Contributions

All team members contributed equally across different phases of the project to ensure a balanced workload and seamless collaboration.

- **Nagamedha Sakhamuri:** Focused on text preprocessing (cleaning, tokenization, embedding generation) and feature engineering using TF-IDF, Word2Vec, and BERT techniques.
- **Nookaraju Guttula:** Developed and fine-tuned text classification models, including Naïve Bayes, SVM, LSTM, and BERT, and handled evaluation tasks for text classifiers.
- **Laxmi Prasanna Ravikanti:** Designed and trained image classification models using CNN-based architectures like ResNet and EfficientNet, along with visual feature extraction.
- **All Team Members:** Collaboratively worked on multimodal learning integration (text + image), final system evaluation, testing, and project documentation.

8.2. Project Timeline

Phase	Task	Duration
Phase 1	Dataset understanding, preprocessing for text & images	Mar 15 – Mar 25
Phase 2	Feature extraction & model design (text and image)	Mar 26 – Apr 5
Phase 3	Training & fine-tuning of text & image classifiers	Apr 6 – Apr 15
Phase 4	Multimodal learning integration & evaluation	Apr 16 – Apr 20
Phase 5	Documentation, results consolidation & submission	Apr 21 – Apr 25

9. CONCLUSION

9.1. Summary

This project successfully developed an AI-driven multimodal classification system that integrates both text and image data to automate product categorization for e-commerce platforms. We used advanced Natural Language Processing (NLP) techniques such as TF-IDF, Word2Vec, and BERT embeddings alongside Computer Vision (CV) models like CNNs, ResNet, and EfficientNet. This integration enabled us to achieve more accurate, reliable, and scalable classification than single-modality methods.

9.2. Did We Meet the Goals?

Yes, we achieved our core project objective. We built and evaluated a robust classification system that addresses major challenges in manual product categorization, including scalability, accuracy, and consistency. Our system validated the hypothesis that multimodal learning (text + image) leads to better performance compared to text-only or image-only models.

9.3. Key Insights

- Advanced embeddings like BERT provide deeper semantic understanding of product descriptions, improving classification precision.

- Visual features extracted from CNN-based models complement text data, especially in cases where descriptions alone are ambiguous.
- A multimodal approach significantly enhances model robustness and generalization in real-world scenarios, laying the groundwork for more sophisticated classification systems in the future.

REFERENCES

- [1] **TF-IDF** – A statistical measure for evaluating word importance in documents.
K. S. Jones, “A statistical interpretation of term specificity and its application in retrieval,” *Journal of Documentation*, vol. 28, no. 1, pp. 11–21, 1972. <https://doi.org/10.1108/eb026526>
- [2] **Word2Vec** – A method to learn word embeddings capturing semantic similarity.
T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013. <https://arxiv.org/abs/1301.3781>
- [3] **BERT** – A transformer model that learns deep bidirectional representations from text.
J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018. <https://arxiv.org/abs/1810.04805>
- [4] **ResNet** – A convolutional neural network architecture introducing residual learning.
K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE CVPR*, 2016. <https://arxiv.org/abs/1512.03385>
- [5] **EfficientNet** – A scalable CNN model balancing accuracy and computational efficiency.
M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” in *Proc. ICML*, 2019. <https://arxiv.org/abs/1905.11946>
- [6] **SIFT** – D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *International Journal of Computer Vision*, 2004. <https://doi.org/10.1023/B:VISI.0000029664.99615.94>

Repository Links:

GitHub Repository: Source code, setup instructions, Documentation

Google Drive: Executed notebook files for Outputs and Results Visualization.