**Title: From Pixels to Profit: A Multimodal, Multitask System for Generating and Optimizing E-Commerce Listings**

**(Abstract placeholder)**

**1. Introduction**

The global e-commerce market, valued in the trillions of dollars, hinges on the effectiveness of the digital product listing. This listing is the primary touchpoint between consumer and product, where high-quality imagery and compelling descriptions are critical drivers of engagement and conversion. The manual creation of these listings, however, is a significant operational bottleneck for retailers, being both time-consuming and difficult to scale.

To address this, generative AI has emerged as a promising solution, yet current approaches exhibit fundamental limitations. Many systems treat the constituent tasks of listing creation—attribute tagging, description writing, and price estimation—as a siloed workflow, failing to build a holistic product understanding. More critically, state-of-the-art Vision-Language Models (VLMs), while capable of generating fluent prose, are prone to factual "hallucinations," inventing details not present in the source image (Jiang et al., 2024). In an e-commerce setting, this requires expensive manual efforts to correct and system reliability comes into scrutiny. Furthermore, existing AI tools are almost exclusively *descriptive*; they can detail what a product *is*, but they cannot provide the *prescriptive* guidance needed to inform what a product *should be* for optimal market positioning.

To overcome these challenges, we propose **Opti-List**, an end-to-end multimodal system that generates complete, factually-grounded, and strategically-optimized e-commerce listings from a single product image. Moreover, we try to provide insights to the product designer to increase the product’s perceived value. Our system is built upon a synergistic trio of contributions designed to directly address the aforementioned problems:

* To combat factual hallucinations, we propose a **Hierarchical Generation (HG)** process. Our model first predicts a set of structured attributes and subsequently uses this factual scaffold to constrain a large language model, ensuring the generated text remains faithful to the visual evidence.
* To move beyond siloed models and capture the subtle visual cues that determine product value, we develop a **Holistic Visual Representation (HVR)** using multi-task learning (Ruder, 2017). A single vision encoder is jointly optimized on attribute prediction and price estimation, forcing it to learn a rich, shared representation that benefits all downstream tasks.
* To bridge the gap from description to decision-making, we introduce a novel **Prescriptive Analytics (PA)** engine. This component leverages our trained model to run "what-if" simulations, providing sellers with data-driven suggestions to optimize product attributes for a higher predicted market value.

Our primary contributions are therefore:

1. A **Hierarchical Generation (HG) process** that uses predicted structured data to guide text generation, significantly improving the factual consistency of product descriptions.
2. A **Holistic Visual Representation (HVR)** learned via multi-task learning that jointly models product attributes and price, leading to improved predictive accuracy and model efficiency.
3. A novel **Prescriptive Analytics (PA) engine** that simulates the financial impact of attribute modifications, transforming the generative tool into a strategic business partner.

**2. Related Work**

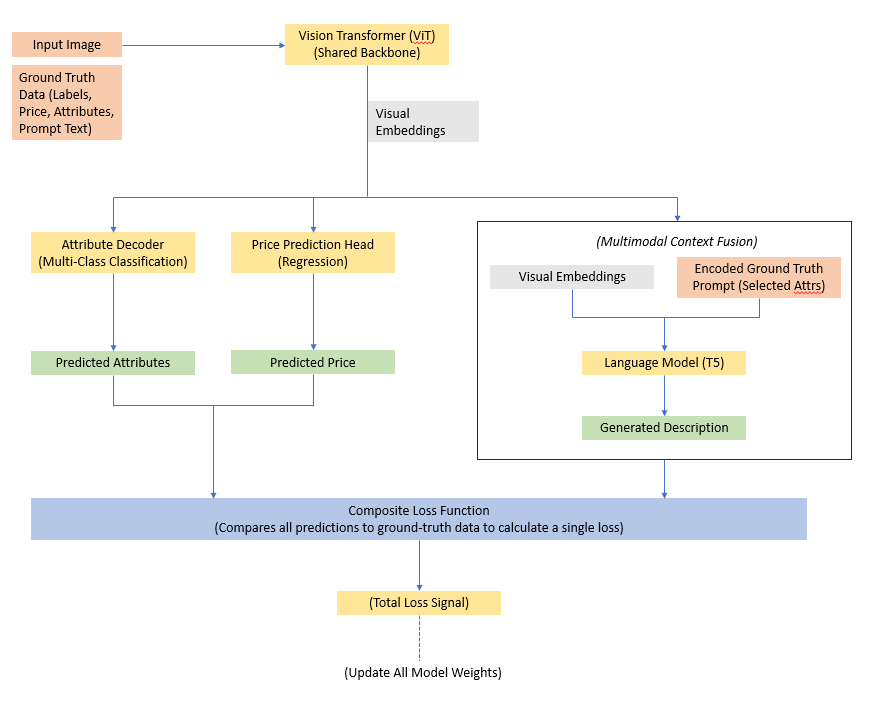
Our research builds upon established work in multimodal learning, controlled text generation, and multi-task learning.

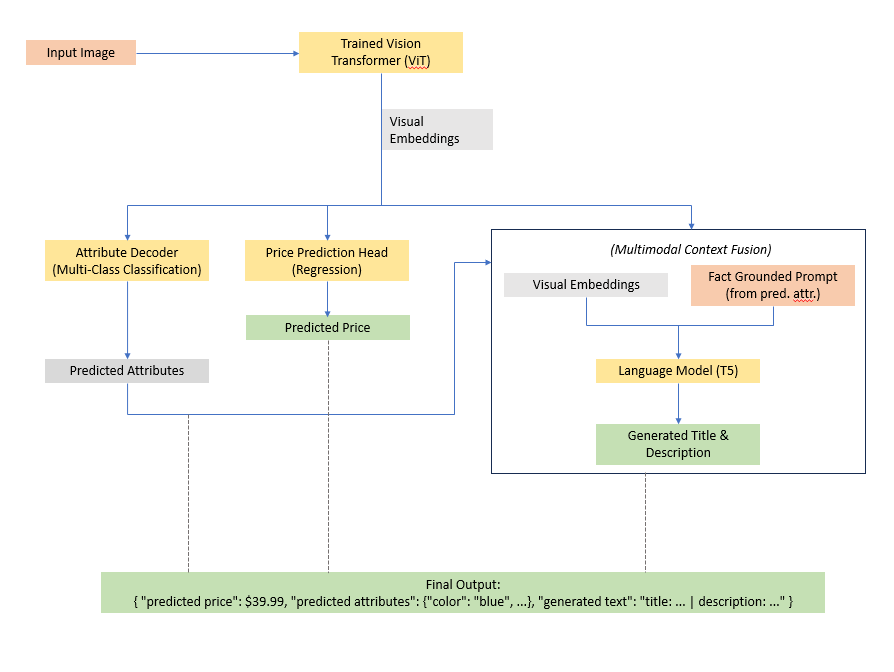
**2.1 Multimodal Learning for E-commerce**  
The application of machine learning in e-commerce has evolved from unimodal to multimodal approaches. Early work often focused on single-modality tasks, such as product classification from text or visual-based recommendation systems (He et al., 2016). Recognizing these limitations, more recent work has focused on the synergy between modalities, as multimodal frameworks demonstrate higher predictive accuracy (Gao et al., 2023). This has led to three key research thrusts relevant to our work: **visual attribute extraction**, which models the problem as a multi-label classification task (Chen et al., 2022); **image-to-text generation**, which leverages VLMs to produce descriptive captions (Vinyals et al., 2015); and **price prediction** from visual cues, where models learn to estimate market value by capturing subtle indicators of quality and style (Bell & Pádraic, 2016). To our knowledge, Opti-List is the first system to unify all three tasks into a single, end-to-end architecture designed for factual consistency and prescriptive analytics.

**2.2 Hierarchical and Controlled Text Generation**  
A primary challenge for LLMs is maintaining factual grounding. To mitigate hallucinations, several control strategies have been developed. Retrieval-Augmented Generation (RAG) grounds model outputs by first retrieving relevant documents from an external knowledge base to provide context (Lewis et al., 2020). Chain-of-thought prompting improves logical consistency by guiding the model to generate intermediate reasoning steps (Wei et al., 2022). Our Hierarchical Generation (HG) method aligns with this paradigm of controlled generation. By first predicting structured attributes and then using them to condition the language model, we use the image's content as a self-contained, verifiable knowledge source to guide the final text output, directly addressing the hallucination problem in a novel way.

**2.3 Multi-Task Learning in Computer Vision**  
Multi-Task Learning (MTL) is a training paradigm where a model learns multiple objectives from a shared representation, often leading to improved data efficiency and generalization by encouraging the model to learn features that are broadly useful (Ruder, 2017). The assumption is that an inductive bias introduced by the auxiliary tasks can help the main task. MTL has been successfully applied across numerous domains, including autonomous driving for joint object detection and segmentation (Teichmann et al., 2018). We apply this principle to learn a Holistic Visual Representation (HVR), positing that the tasks of predicting visual attributes and estimating price are sufficiently related—both relying on latent features of quality, style, and material—to produce a more powerful shared representation of a product's visual identity and market value.

**3. Methodology**





3.1 The Encoder

The proposed model architecture, as shown in Figure **x**, relies on a vision encoder and a text decoder structure. The encoder chosen for our study is Vision transformer, hereafter ViT (Dosovitskiy et al., 2020). ViT relies on self-attention mechanism that is in theory better than traditional CNNs to capture fine grained local features such as fabric texture or holistic context defining the product style such as overall silhouette.

Among the various attributes, a good mix of visually detectable attributes such as colour, sleeve length etc. and derived attributes such as style, occasion etc. was chosen. This deliberate selection of attributes is intended to enable the model to learn not only “what’s visible” but also “what it implies”. For example, if model frequently encounters black one-piece (visually identifiable cues) it can attribute it to evening dress suitable for parties.



For eg.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Dress 1** | **Dress 2** | **Dress 3** |
| **Neck** | **One Shoulder** | **Deep V-Neck** | **Sweetheart Neck** |
| **Sleeve Length** | Sleeveless | Sleeveless (Spaghetti Straps) | Strapless |
| **Print or Pattern Type** | Solid | Solid | Solid |
| **Type** | Bodycon Gown | A-Line Gown | Mermaid Gown |
| **Hemline** | Flared | Flared | Flared |
| **Pattern** | Plain | Plain | Plain |
| **Length** | **Floor Length** | **Floor Length** | **Floor Length** |
| **Sleeve Styling** | No Sleeves | Strappy Sleeves | Strapless |
| **Ornamentation** | None | None | None |
| **Occasion** | **Evening / Formal** | **Evening / Formal** | **Evening / Formal** |
| **Fabric** | Stretch Crepe | Satin or Crepe | Stretch Crepe |
| **Fit** | Slim Fit | Regular Fit | Body Fit |
| **Colour** | Black | Black | Black |

The ViT model processes each image as a hidden state for each image patch, and also provides a pooled representation for the entire image. We take this pooled representation as the input for our task specific heads. As shown in the Figure **X,**  two different MLP heads is attached onto the visual embeddings to process the attributes and price respectively.

3.2. The Decoder

Inputs -

1. The final hidden state of ViT is extracted. This vector is projected to suitable dimension for decoder input, 768 -> 512. We have used T5 small as text decoder.

2. A generic text prompt created from a few ground truth attributes is also passed to the decoder. These are created as text tokens e.g., *"generate a product listing for an item of type 'garment', suitable for 'everyday wear'"*.

occasion = row.get('Occasion', 'everyday wear')

item\_type = row.get('Type', 'garment')

prompt = f"generate a product listing for an item of type '{item\_type}', suitable for '{occasion}'"

Both the visual token and the encoded prompt tokens are concatenated along the sequence dimension. This creates a single, fused context vector (e.g., [IMAGE\_TOKEN, PROMPT\_TOKEN\_1, ...]) that the T5 decoder can attend to. The idea to let the model learn integration of high-level textual instruction with low level visual details. We intentionally didn’t provide detailed visual attributes otherwise model will become lazy and only rephrase the attributes and ignore visual embedding.

A different strategy is employed during inference. The classification head is asked for product attributes. This predicted attribute is transferred to the text generator using a detailed prompt. The T5 decoder uses the visual embedding and the new factual grounded prompt to return the final description and name of the listing. The proposed grounding should help with lowering the hallucination.

Joint Loss Function

The model is jointly optimized for the three tasks together. During training the losses from each task is combined together to form the total loss. The loss from classification (Cross-Entropy) and from regression (MSE) can vary hugely in magnitudes. To prevent one head dominance over the other, they are weighted before summation.

Mathematically,

L\_total = α \* L\_attr + β \* L\_price + γ \* L\_text

Individual Losses:

1. **Price Loss:** Mean Squared Error (MSE) loss calculated between the Predicted Price and the ground-truth price.
2. **Attribute Loss:** Cross Entropy Loss calculated for each of the 52 attributes between the predicted logits and the ground-truth labels. The final attribute loss is the average of these individual losses.
3. **Text Loss:** The standard cross-entropy loss for language modeling is computed internally by the T5 model when labels are provided.

The gradient from total loss backpropagates through all model components, forcing the ViT encoder to learn a feature representation that is beneficial for all three tasks simultaneously. This multitasks learning approach, in theory, incentivizes the model to learn the relationship between attributes and market value. This is demonstrated in Figure 3. The plain white shirt with exact same attributes may be much cheaper than a shirt printed with some characters. Our training wants to simulate learning this relationship.



Figure 3: Which one looks expensive?

**4. Experiments**

**4.1 Dataset**

The dataset consists of 14.2k women clothing listings collected from several public e commerce website. The data is then tabulated using their features. This includes a set of 52 different columns including rating, price, name and description. The images were resized to 224 x 224 pixels. These 52 inputs were segregated in three different types of features – easy visually detectable like neck shape, could be detected like fabric texture and inferred features like occasion. This mix was considered for training the model. Other features that were tooniche for model to learn like brand or unique identifiers were removed. Final attributes selected for training is listed below:

SELECTED\_ATTRIBUTES = [

*# --- Tier 1: Core Visuals ---*

    'Neck',

    'Sleeve Length',

    'Print or Pattern Type',

    'Type',

    'Hemline',

    'Pattern',

*# --- Tier 2: Detailed Visuals ---*

    'Length',

    'Sleeve Styling',

    'Ornamentation',

*# --- Tier 3: Contextual for Text ---*

    'Occasion',

    'Fabric',

    'Fit'

]

**4.2 Evaluation Metrics**

The following metrics were employed to determine the model’s performance.

**Attribute Prediction:** The multi-classification task is evaluated using Macro F1 Score. This score was used because the it provides a balanced assessment across all attribute classes, regardless of their frequency.

**Price Prediction:** For this regression task, we report **Mean Absolute Error (MAE)** for its intuitive interpretation in currency units, **Root Mean Squared Error (RMSE)** to penalize large prediction errors, and **R-squared (R²)** to measure the proportion of the variance in the price that is predictable from the model's inputs.

**Text Generation:** We use a suite of metrics to assess text quality:

* **BLEU-4 and ROUGE (1, 2, L):** We report these standard n-gram overlap metrics for comparability with prior work in text generation.
* **Hallucination Rate:** This is our primary metric for factual consistency, directly evaluating our HG claim. It is defined as the percentage of generated descriptions where the model mentions an attribute value that **contradicts the structured attributes predicted by its own vision module**. This measures the model's internal consistency. A lower rate is better.
* **Model Efficiency:** To evaluate the computational cost of our models, we report:
  + **Parameters (M):** The total number of trainable parameters in millions, indicating the model's size.
  + **Latency (ms):** The average time in milliseconds required to perform a full end-to-end inference for a single sample on our target hardware.

**4.3 Baselines and Ablations**  
To prove that proposed multi task learning heirchial architecture is better suited for this task, the model is compared against several well-defined baselines.

* **Baseline-Siloed:** These category of model consists of two models independently trained for price prediction and attribute prediction task respectively. These models are required for testing the impact of multi- task learning approach. If the proposed solution provides better performance then these independent frameworks then the proposed model was able to learn the relationship between the attributes and price.

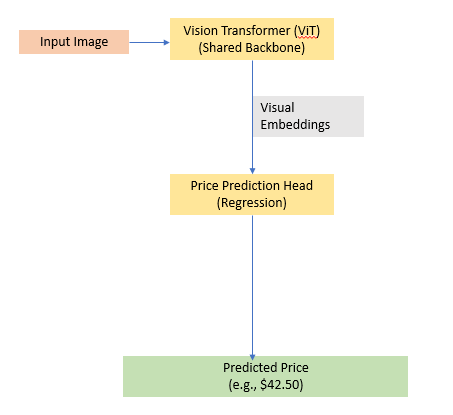


Figure: Model Architecture of Siloed Price Model

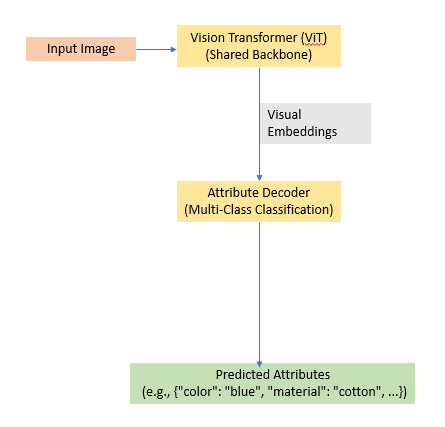


Figure: Model Architecture of Siloed Attributes Model

* **Baseline-Direct VLM:** The direct Vision-to-Language model takes in input image, generate the entire product description directly from the image, without any intermediate structured attribute prediction. This is the primary baseline for evaluating our hierarchical generation claim. Two different VLM model types were chosen as baseline. A single unified transformer structure the GIT-base model fine tuned on the training dataset. The decoder attends to image patches and the previous text. A classic example of Self-Attention mechanism. The other type with ViT encoder and T5 small decoder. Here, the decoder attends to only the output of encoder, an example of cross-attention mechanism. The pre-trained model taken as reference needs to be of similar parameter size as the baseline proposed model. This is necessary requirement as we don’t the reference model to simply brute force the architectural limitation with model size or pre-training data. Consequently, the GIT-base model with ~138M parameters was selected.

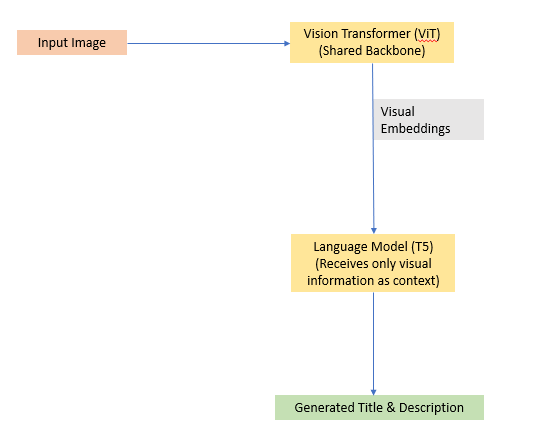


Figure: Model Architecture of direct Vision to Language Model (VLM)

* **Baseline-Hybrid:** This model has two different models trained indepently. One model is XG Boost trained on ground truth attributes tasked to generate the price prediction. The other one the siloed attribute training model that generates attributes from image. During inference the predicted attributes by this model is fed to the XG boost model to predict the price. This model is taken as baseline to prove that information embedded in the visual embedding is necessary. The long data flow chain can omit some relevant information.

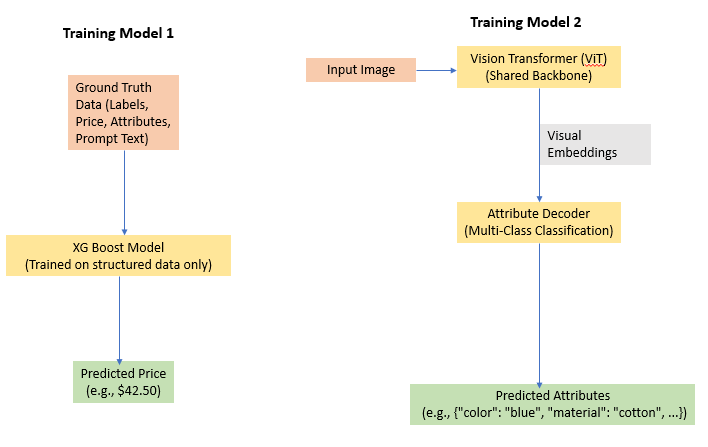


Figure: Training Architecture for Hybrid Model, both models are trained independently

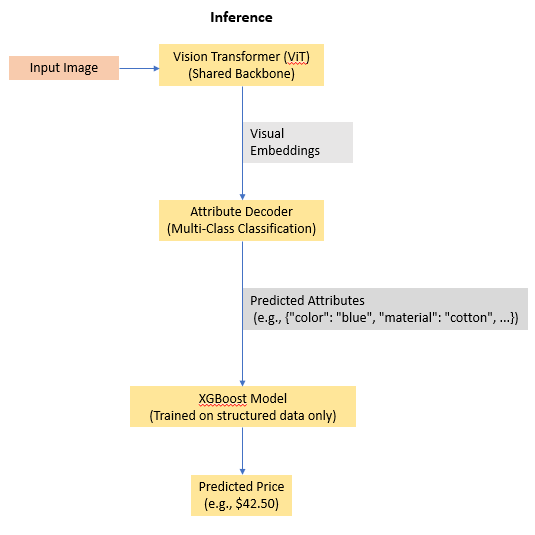


Figure: Inference Architecture for Hybrid Model, independently trained model is joined together

* **Ablate-No MTL:** The same as proposed model architecture but without the price prediction head. Since, the price head is removed, the related loss is also removed from the combined loss. This model tests if learning price and attributes together aids in model performance.
* **Ablate-No-** **Hierarchy:** The same as proposed model but during inference, the text decoder is only passed the visual embedding. The predicted attributes are not passed to the model. This model is required to test whether grounding the text decoder with predicted attributes helps in reducing hallucination or not.

**4 Implementation Details**

The proposed model consists of a ViT-Base/16 vision backbone and a T5-Small text generator. The total parameters is 147.7M. The model was trained end-to-end for 25 epochs using the Adam optimizer with a learning rate of 1e-4 and a batch size of 128. The composite loss function was weighted with α=0.4 (attributes), β=0.1 (price), and γ=0.5 (text), values determined via an ablation study. All experiments were conducted on a single NVIDIA GeForce RTX 4050 with 6 GB of VRAM. Further details on the learning rate schedule, optimizer parameters, and beam search configuration are provided in Appendix A.

**5. Results and Discussion**

**Multi Task Learning vs Independent Task Learning**

While the results shown in Tabe 1 look abysmal for any model to be deployment ready. The task of predicting price is inherently difficult with only image as the input. The brand, often not visually detectable, plays an important role factor here. A plain white T shirt from Gucci may be 100x times more costly than H&M t shirt. The price is also influenced by factors like market trend, seller reputation etc. The model has no access to this information. Also, due to huge variation between prices – 1500INR to 25000INR, the RSME of 1935 INR, though high, is not unexpected. Even with all these influencers our model is able to explain about 50% of variation in price with only images as input.

I do concede that these results are not deployment ready and we need to add more features in the model to correctly simulate the phenomena. However, the results perfectly demonstrate that multitasking is able to explain 3.64% better variance in price than the independent price model.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Price MAE (in INR)** | **Price RMSE (in INR)** | **Price R2** |
| Multi Task Learning | **1067.67** | **1935.81** | **0.476** |
| Siloed - Price | 1074.31 | 1966.55 | 0.460 |

Similarly, the average macro F1 score of 0.3 is modest. It is a massive multi-label, multi-class classification problem with several inherent challenges. We are predicting the attributes for 12 categories. But each of these attributes have many different classes. Thus, each image has 399 (sum of each class under each attribute) possible labels. Also, the dataset contains a long tail of rare attribute values. Some of these labels is difficult for even humans to detect with just eyes for eg. difference between “Fabric” features labels “Cotton”, “Cotton Silk”, "Polycotton", "Organic Cotton", "Pure Cotton", "Cotton Blend" is hard to judge.

Though the absolute value of F1 score is not suitable for deployment, the trend in values lead to an import conclusion. The multi task learning model demonstrated a 6.6% better performance than the Siloed model.

|  |  |
| --- | --- |
| **Model** | **Attribute F1** |
| Multi Task Learning | **0.338** |
| Siloed - Attributes | 0.317 |

**Hybrid Model vs Multi-Task Learning**

As shown in Table **XX,** the proposed model achieved a **5.8% lower Mean Absolute** and a **10.1% lower Root Mean Squared Error** compared to the Hybrid Model. Also, the proposed solution explains the variance in price better by 34.9% than the hybrid model. This result can be because of data loss during the pipeline. The predictor model is not able to “see” the image. It only sees the discrete, simplified representation of that image as described by the 12 attributes. Thus, any data that is not represented under these categories is essentially lost such as craftmanship, subtle brand cues stylist nuances. For e.g. whether the garment's silhouette is elegant and well-constructed or ill-fitting influences the listing price of product.

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Price MAE (in INR)** | **Price RMSE (in INR)** | **Price R2** |
| Multi Task Learning | **1067.67** | **1935.81** | **0.476** |
| Hybrid Model | 1133.66 | 2153.15 | 0.352 |

**Model Efficiency**

The proposed model has 15% fewer parameters than the both the siloed models combined. However, this can provide relevant basis for comparison because the multitask model contains parameters for text generation while the siloed models only generate the attributes or the price. Thus, the time taken is 19ms or 13ms by the siloed models or hybrid model when compared to 251ms taken by MTL model.

As seen in the parameter count the additional MLP heads for price and attribute prediction increases the number of parameters marginally (~0.4M). This may add to a negligible computation cost. However, the information provided by them greatly accelerates the autoregressive text generation process. They provide the language model a concise and fact grounded prompt, thereby, significantly reducing the search space for the decoder. Thus, the beam search algorithm converges quickly, reducing the end-to-end latency by a factor of 3.5x compared to an equivalent VLM model. The GIT-base model despite its higher parameter count takes lesser time to generate the output. The unified transformer mechanism or the length of text generated may be the factor at play here. However, we will not investigate this further in this paper, as our focus remains at Multi Task Learning-Hierarchical model. The proposed model clearly surpasses its counterparts.

|  |  |  |
| --- | --- | --- |
| **Model** | **Parameters (M)** | **Latency (ms)** |
| Multi Task Learning - Hierarchical | **147.7** | **251** |
| ViT-T5small-Direct Vision to Language | **147.3** | **883** |
| GIT-base-Direct Vision to Language | **176.6** | **586** |
| Baseline-Siloed | 173.1 | 19 |
| Baseline-Hybrid | 86.7 | 13 |

**Hierarchical vs Non-Hierarchical**

The data presented in Table **X**, is evaluated on 1422 test samples. The clear distinction between the performance of hierarchical model and non-hierarchical model lies in the hallucination rate metric. The results show that the proposed approach demonstrates a 44.5% relative reduction in factual contradictions.  Thus, a containing prompt with attributes generated from visual analysis is a highly effective strategy to ensure that the final text remains grounded in visual facts. However, this improvement in factual consistency comes at the cost of n-gram based similarity metrics. Since, the non-hierarchical approach is less constrained, it performs 6.4% better on the ROUGE-L metric.

But, this trade off of similarity score for factual consistency is desired. Due to lower constraints the non-hierarchical model may be more creative. Thereby, the model can “invent” some facts to match the reference text n-grams.

The average generated length for hierarchical and non- hierarchical model is 34 words and 32 words respectively. Thus, the additional attribute prompting encourages thoroughness because we are supplying information to include in the final output.

We are not testing the hallucinate rate for direct vision to language models because these models are not tasked to predict the attributes. This is intentional decision made to ensure that VLM models are not overwhelmed with the tasks. This decision remains consistent with the objective of model to generate the name and description of the product.

The hierarchical model performs poorer than the direct VLM model in fluency metrics, e.g. 3.7% lower ROUGE-L score than the ViT-t5small VLM baseline. However, as discussed earlier this is deliberate trade off better factual consistency. If we remove the hierarchy (attribute grounding) from the model during inference the model (Multi Task Learning – Non-Hierarchical) performs 2.3% better than the ViT-t5small VLM baseline on ROUGE L score. This proves that hierarchical training is still the better approach the direct VLM architectures.

The GIT-base VLM model performs poorly on all text-based metrics. This may be because the model lacks pre-trained decoder that was specifically trained on large corpus of text like T5small. This suggests that mixing pre-trained capabilities ensures better results than relying on “purity” of training done on a smaller dataset.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **BLEU** | **ROUGE-1** | **ROUGE-2** | **ROUGE-L** | **Hallucination Rate** |
| Multi Task Learning - Hierarchical | 0.155 | 0.420 | 0.240 | 0.403 | **7.0%** |
| Multi Task Learning – Non-Hierarchical | 0.152 | 0.444 | 0.254 | **0.429** | 12.7% |
| GIT-base-Direct Vision to Language | 0.006 | 0.174 | 0.002 | 0.153 | NA |
| ViT-T5small-Direct Vision to Language | 0.220 | 0.429 | 0.289 | **0.419** | NA |

**6. Limitations and Future Work**

**(Limitations and Future Work placeholder, often included in the Conclusion)**

**7. Conclusion**

**(Conclusion placeholder)**

**References**

Bell, R. M., & Pádraic, M. (2016). Hedonic Price Models and the Evaluation of Product Attributes. *Consumer Demographics and Behaviour*, 1-21.  
Chen, L., Wu, J., Zhang, Y., & Zhang, Y. (2022). A Multi-Task Multi-Label Classification for Attribute Extraction in E-commerce. *Proceedings of the 2022 International Conference on E-business and Mobile Commerce*.  
Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.  
Gao, C., Wang, S., Zhang, J., & He, X. (2023). A Survey of Multimodal Deep Learning in E-commerce. *ACM Transactions on Management Information Systems*.  
He, R., Xu, J., & McAuley, J. (2016). VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. *Proceedings of the AAAI Conference on Artificial Intelligence*.  
Jiang, N., Wang, H., Tang, S., Lin, Z., Wu, L., & Zhuang, F. (2024). A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. *arXiv preprint arXiv:2311.05232*.  
Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459-9474.  
Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners. *OpenAI blog*, 1(8), 9.  
Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.  
Teichmann, M., Weber, M., Zöllner, M., Cipolla, R., & Urtasun, R. (2018). MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving. *2018 IEEE Intelligent Vehicles Symposium (IV)*.  
Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. *Proceedings of the IEEE conference on computer vision and pattern recognition*.  
Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35, 24824-24837.

|  |
| --- |
| mtl\_hierarchical |
| mtl\_non\_hierarchical |
| direct\_vlm – ViT-T5  direct\_vlm – GIT base  etc. |

I have fine-tuned vision to language models for generating the names and descriptions of clothes from images. I am evaluating the models now. I have trained multiple models mtl\_hierarchical

mtl\_non\_hierarchical

direct\_vlm – ViT-T5

direct\_vlm – GIT base

etc.

**Evaluation & Experimentation Action Plan**

**Phase 1: Core Performance & Efficiency Benchmarking**

This phase gathers the primary quantitative results for your paper's main comparison tables.

* **Action 1: Run Comprehensive Model Evaluation**
  + **Purpose:** To get the core performance metrics (MAE, RMSE, R², F1-Score) for the MTL and Siloed models. This directly addresses **Experiment 1.1**.
  + **Script:** evaluate\_all\_models.py
  + **Command:**

codeBash

python -m src.evaluation.evaluate\_all\_models --output\_dir results/final\_evaluation

* + **Key Outputs:**
    - results/final\_evaluation/model\_comparison\_summary.csv: The main table comparing Ours-MTL and Baseline-Siloed on price and attribute metrics.
    - Individual JSON files for detailed results.
* **Action 2: Evaluate the Hybrid Price Baseline**
  + **Purpose:** To get the price prediction performance for the powerful non-visual baseline. This addresses **Experiment 1.2**.
  + **Script:** evaluate\_hybrid\_test.py
  + **Command:**

codeBash

python -m src.evaluation.evaluate\_hybrid\_test

* + **Key Outputs:** Console output with the MAE, RMSE, and R² for the Baseline-Hybrid model. You will manually compare these numbers to the price metrics from Action 1.
* **Action 3: Run Efficiency Benchmarks**
  + **Purpose:** To measure the parameter counts and inference latency for all models, providing the data for your efficiency claims. This addresses **Experiment 4.1**.
  + **Script:** benchmark.py
  + **Command:**

codeBash

python benchmark.py

* + **Key Outputs:**
    - benchmark\_results.csv: A summary table comparing all models on parameter count and end-to-end latency.
    - Detailed console output.

**Phase 2: Text Generation Quality & Hallucination Analysis**

This phase focuses on proving the superiority of the hierarchical generation approach.

* **Action 4: Evaluate Hierarchical Generation**
  + **Purpose:** To compare Ours-MTL, Ours-Ablate-NoHier, and Baseline-DirectVLM on text quality (BLEU, ROUGE) and, most importantly, factual consistency (**Attribute Hallucination Rate**). This is the core of **Experiment 2.1**.
  + **Script:** evaluate\_hierarchical\_generation.py
  + **Command:**

codeBash

python -m src.evaluation.evaluate\_hierarchical\_generation --output\_dir results/hierarchical\_evaluation

* + **Key Outputs:**
    - results/hierarchical\_evaluation/hierarchical\_generation\_summary.csv: The primary results table for your paper's text quality section.
    - results/hierarchical\_evaluation/prediction\_examples.json: Raw text outputs for qualitative review.
* **Action 5: Generate Qualitative Visuals**
  + **Purpose:** To create the side-by-side image and text comparisons for your paper's figures, providing visual proof for **Experiment 2.2**.
  + **Script:** generate\_qualitative\_examples.py
  + **Command:**

codeBash

python -m src.generation.generate\_qualitative\_examples --output\_dir results/qualitative\_examples

* + **Key Outputs:** comparison\_example\_\*.png files in the output directory, ready to be inserted into your paper.

**Phase 3: Business Value & User Study Preparation**

This phase creates the materials needed for human evaluation.

* **Action 6: Generate User Study Assets**
  + **Purpose:** To produce all the necessary data and files to conduct the A/B user study on the value of strategic suggestions. This addresses **Experiment 3.1**.
  + **Script:** generate\_user\_study\_data.py
  + **Command:**

codeBash

python -m src.generation.generate\_user\_study\_data --output\_dir results/user\_study --num\_samples 50

* + **Key Outputs:** A folder containing images, user\_study\_data.json (for you), and evaluation\_template.json (to send to evaluators).

**Phase 4: Supporting Ablation & Statistical Analysis**

These final steps add depth and rigor to your claims.

* **Action 7: Run Ablation Studies**
  + **Purpose:** To analyze the impact of your model's design choices (loss weights, hierarchical prompting) and justify your final architecture.
  + **Script:** run\_ablation\_study.py
  + **Command:**

codeBash

python -m src.analysis.run\_ablation\_study --output\_dir results/ablation\_studies

* + **Key Outputs:** CSV files (loss\_weight\_ablation.csv, etc.) showing how performance varies with different configurations.
* **Action 8: Perform Statistical Significance Testing**
  + **Purpose:** To formally prove that the observed differences in performance (e.g., the lower hallucination rate of your model) are statistically significant.
  + **Script:** statistical\_tests.py
  + **Execution:**
    1. Create a Jupyter Notebook for analysis.
    2. Load the detailed results from the JSON files generated in Phase 1 and 2.
    3. Import the StatisticalTester class: from src.analysis.statistical\_tests import StatisticalTester.
    4. Use the class methods to perform paired t-tests on the metrics and generate a significance report.