TRANSEXPLAIN: A UNIFIED FRAMEWORK FOR FINE-GRAINED, CLASS-CONDITIONED EXPLAINABILITY IN VISION AND LANGUAGE TRANSFORMER MODELS

Jayan Ghimire

Software Engineer and Independent AI Researcher

Leapfrog Technology

Kathmandu, Nepal

jghimire.034@gmail.com

ABSTRACT

Transformer-based architectures have achieved state-of-the-art performance across both **computer vision** and **natural language processing** tasks due to their ability to model **long-range dependencies** through self-attention mechanisms. However, their inherently opaque internal representations pose significant challenges for **trust**, **fairness**, and **accountability** in real-world deployment. In this work, we present **TransExplain**, a principled and modular **explainability framework** for interpreting transformer models in both the **visual** and **textual** domains. Unlike prior methods that rely heavily on attention heuristics or are architecture-specific, TransExplain introduces customized **Layer-wise Relevance Propagation (LRP)** strategies tailored for **Vision Transformers (ViT, DeiT)** and **BERT**-based language models. ***In the vision pipeline***, we generate **class-discriminative heatmaps** that localize predictive evidence by backpropagating relevance through self-attention and MLP layers. ***For text classification***, we produce **token-level attribution scores** that transparently reveal how semantic elements contribute to the model's prediction. TransExplain supports flexible classification schemas, is compatible with pretrained transformer models, and delivers **consistent, theoretically grounded explanations** across modalities. This work lays a strong foundation for future research on unified multimodal interpretability while addressing the pressing need for reliable explanation tools in unimodal transformer-based systems.

1 Introduction

1.1 Motivation

Transformer models have fundamentally reshaped the landscape of **machine learning** by enabling highly effective representations in both **vision** and **language** domains. Their **self-attention mechanisms** excel at capturing intricate **contextual relationships**, empowering breakthroughs in tasks such as **object recognition**, **image captioning**, and **sentiment classification**. However, the internal workings of these architectures remain largely **inscrutable**, limiting our ability to **understand** and **trust** their decisions.

Existing **interpretability** approaches—ranging from **attention weight visualizations** to **gradient-based attribution**—often fall short in delivering consistent, **class-specific explanations**, especially in **multimodal** scenarios where inputs span both images and text. Furthermore, many methods lack **theoretical grounding**, relying on **heuristics** that may obscure the true reasoning pathways of the model.

To advance **explainability** for **transformer-based systems**, there is a pressing need for an integrated **framework** that unifies **attribution techniques** across modalities with solid **theoretical foundations**. By combining principled

relevance propagation with flexible **class-conditioned explanations**, such a system can unlock **transparent insights** into what features the model deems important, thereby facilitating **debugging**, **fairness evaluation**, and **user trust**.

1.2 Our Paper's Contribution

- We propose a custom-designed **visual explanation module** for **Vision Transformers (ViT)** and **DeiT**, which utilizes an enhanced form of **Layer-wise Relevance Propagation (LRP)** tailored for attention-heavy architectures. This generates **class-specific, spatially grounded heatmaps** for the top-k predicted categories.
- We develop a novel **textual explanation algorithm** for **BERT-based classifiers**, supporting any user-defined **classification schema** via a dynamic `classifications` array. The method computes **token-level relevance scores** using a custom relevance propagation method that aligns with transformer internals.
- The framework enables **fine-grained, class-conditioned attribution** across both modalities, maintaining conceptual consistency between how relevance is propagated in visual and textual domains.
- Unlike heuristic or attention-only visualizations, our method is **theoretically grounded**, providing **transparent and reliable interpretations** by leveraging proper **relevance decomposition** for transformer layers.

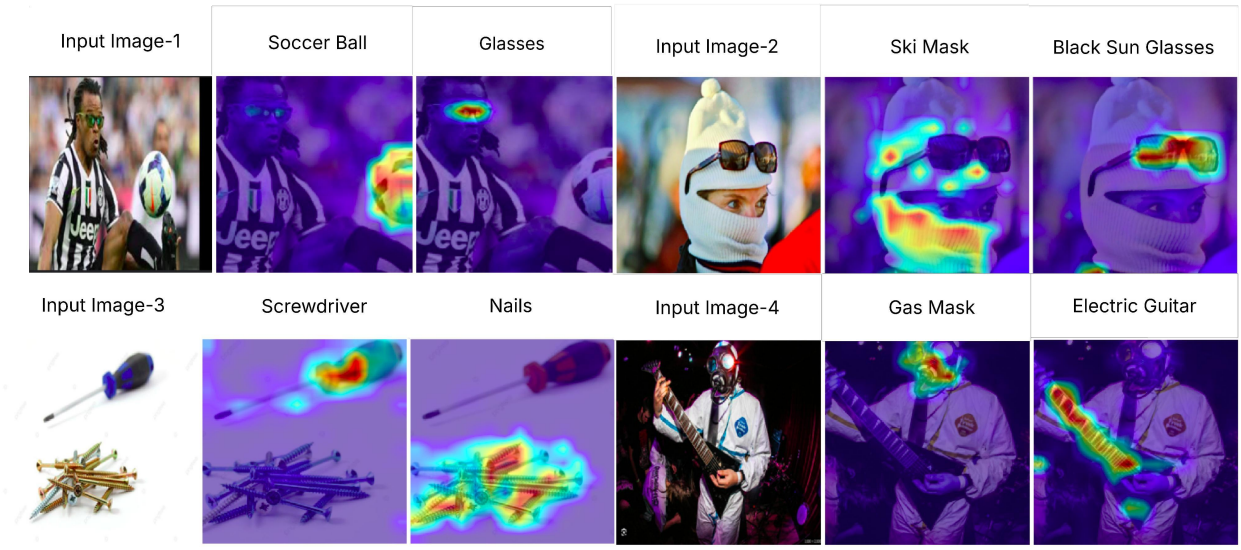

Figure 1: **TransExplain for visual inputs from different classes. For more examples, see Figures 6 and 7.**

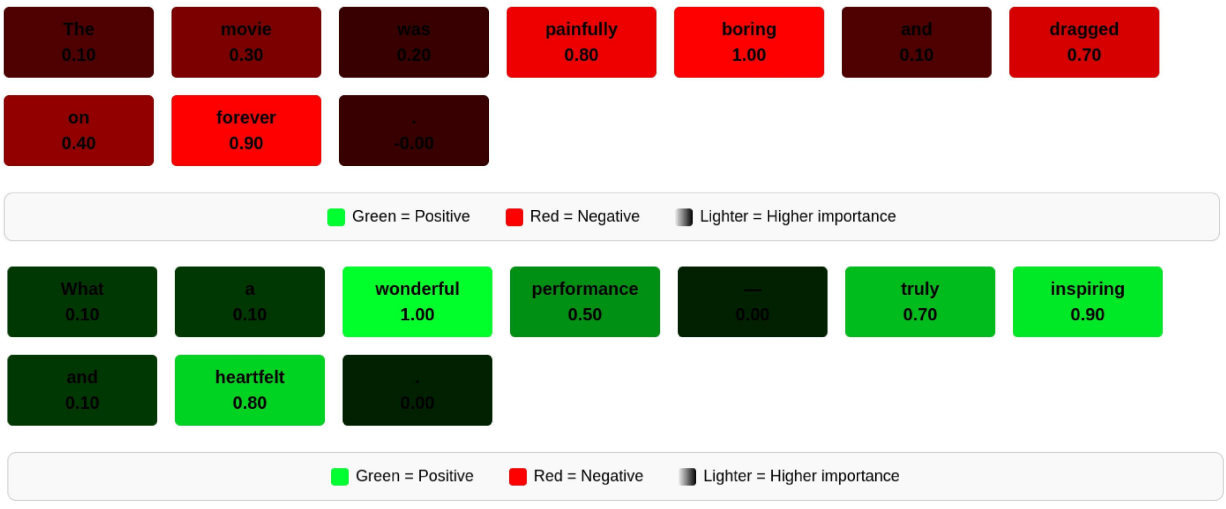

Figure 2: **TransExplain for textual inputs of different connotations. For more examples, see Figures 8 and 9.**

2 Related Work

2.1 Explainability in Vision Transformers

The rise of **Vision Transformers (ViTs)** [1] and their variants such as **DeiT** [2] has introduced new challenges in **interpreting self-attention-based decisions** in image classification. Conventional techniques like **Grad-CAM** [3] and **CAM** [4], originally designed for convolutional architectures, are often ill-suited for transformers due to the absence of spatially-localized convolutional filters. Recent works have attempted to adapt these by aggregating attention maps [5] or propagating relevance using **Layer-wise Relevance Propagation (LRP)** [6]. However, most of these approaches are either tailored to specific transformer architectures or lack consistent theoretical grounding across layers. Our work addresses these limitations by introducing a principled LRP-based visual attribution method for both **ViT** and **DeiT**, producing **class-discriminative heatmaps** that localize predictive evidence with high fidelity.

2.2 Textual Interpretability in BERT and Beyond

Interpretability for **language models** has primarily focused on identifying influential tokens via gradient-based saliency [7], attention weights [8], and perturbation-based techniques like **LIME** [9] or **SHAP** [10]. While effective to some extent, these methods often lack robustness or rely on heuristics. Moreover, attention maps do not always correlate with feature importance [11], raising questions about their validity as explanations. Our framework overcomes this by implementing a customized LRP scheme for BERT-based models, which generates **fine-grained token-level attributions** grounded in the modelś internal relevance flow. This allows transparent analysis of semantic cues driving classification.

3 Methodology

3.1 Proposed System Diagram

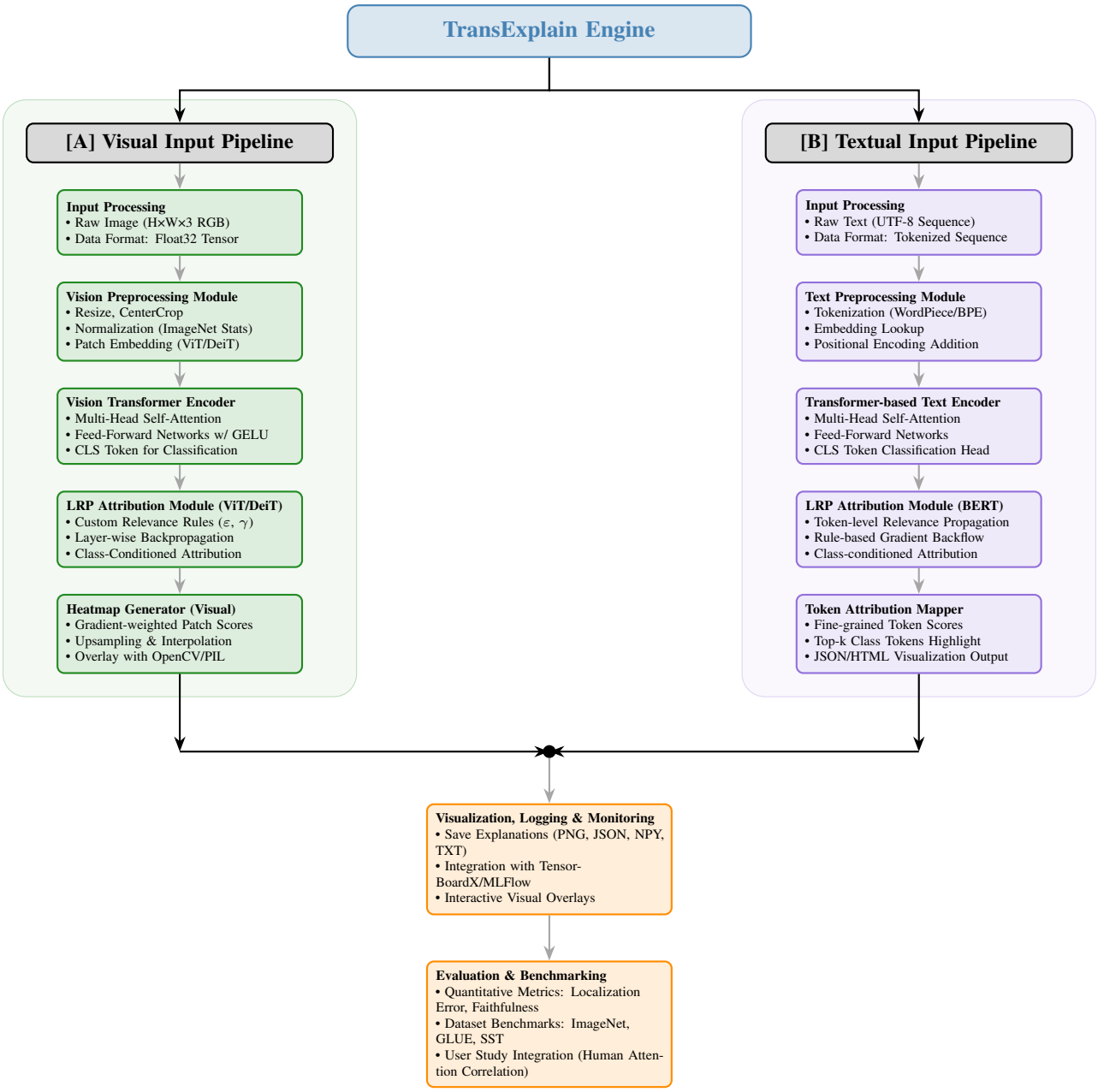

Figure 3: System architecture of TransExplain: An explainability engine for transformer-based multimodal models with Layer-wise Relevance Propagation (LRP) applied to both vision (ViT/DeiT) and language (BERT) inputs, culminating in a unified visualization and evaluation module.

3.2 Workflow

3.2.1 Input Processing

The **TransExplain Engine** begins by accepting raw inputs from two modalities — **visual data** (image) and **textual data** (natural language). These inputs serve as the foundation for the downstream explainability pipeline.

3.2.1.1 Visual Input:

The raw image is represented as a 3-dimensional **tensor** with dimensions $\mathbf{H} \times \mathbf{W} \times \mathbf{3}$, where H and W denote the image's **height** and **width**, and 3 represents the **RGB color channels**. The pixel values are cast to `float32` for numerical compatibility:

$$\mathbf{I}_{\text{raw}} \in \mathbb{R}^{H \times W \times 3}, \quad \mathbf{I}_{\text{float}} = \texttt{cast}(\mathbf{I}_{\text{raw}}, \texttt{float32})$$

3.2.1.2 Textual Input:

The raw text is received as a **UTF-8 encoded character sequence**. It is first normalized and cleaned before being passed to a tokenizer, **Byte-Pair Encoding (BPE)**. The result is a sequence of integer token IDs:

$$\mathbf{T}_{\text{raw}} = \texttt{``I hate that I love you.''}$$

$$\mathbf{T}_{\text{tok}} = \texttt{Tokenizer}(\mathbf{T}_{\text{raw}}) = [\texttt{CLS}, 1023, 4781, \ldots, \texttt{SEP}]$$

This ensures that the input is transformed into a **discrete, numerical format** suitable for embedding and positional encoding in the later transformer layers.

This block guarantees a consistent and structured representation of input across both modalities, setting the stage for modality-specific preprocessing in the subsequent modules.

3.2.2 Preprocessing Modules

Once the raw inputs are structured, they are passed through modality-specific preprocessing pipelines to ensure compatibility with transformer encoders. This block performs crucial operations such as **normalization**, **embedding**, and **positional encoding**.

3.2.2.1 Vision Preprocessing Module:

- **Resizing and Cropping:** The input image $\mathbf{I}_{\text{float}}$ is resized to a fixed resolution, followed by a **center crop**:

$$\mathbf{I}_{\text{resized}} = \texttt{CenterCrop}(\texttt{Resize}(\mathbf{I}_{\text{float}}),\ H', W')$$

- **Normalization:** We normalize the pixel values using **ImageNet statistics** to standardize input across datasets:

$$\mathbf{I}_{\text{norm}} = \frac{\mathbf{I}_{\text{resized}} - \mu}{\sigma}, \quad \mu, \sigma \in \mathbb{R}^3$$

- **Patch Embedding:** The normalized image is divided into fixed-size non-overlapping patches and flattened. Each patch is projected via a learnable linear embedding layer:

$$\mathbf{X}_{\text{patch}} \in \mathbb{R}^{N \times D}, \quad N = \frac{H' \cdot W'}{P^2}, \quad \text{where } P \text{ is patch size}$$

3.2.2.2 Text Preprocessing Module:

- **Token Embedding:** The sequence of token IDs $\mathbf{T}_{\text{tok}}$ is mapped to dense vectors via an **embedding lookup table**:

$$\mathbf{X}_{\text{tok}} = \texttt{EmbeddingMatrix}[\mathbf{T}_{\text{tok}}] \in \mathbb{R}^{L \times D}$$

- **Positional Encoding:** Since transformers lack inherent order-awareness, **positional encodings** $\mathbf{P} \in \mathbb{R}^{L \times D}$ are added to token embeddings:

$$\mathbf{X}_{\text{text}} = \mathbf{X}_{\text{tok}} + \mathbf{P}$$

where L is the sequence length and D is the embedding dimension.

This preprocessing ensures that both visual and textual data are transformed into a shared latent space, ready for contextual learning via transformer encoders.

3.2.3 Transformer Encoders

The preprocessed inputs are passed into modality-specific transformer encoders. These encoders model rich **intra-modal dependencies** through stacked layers of **Multi-Head Self-Attention** and **Feed-Forward Networks (FFNs)**. Both encoders utilize a special `[CLS]` token whose final representation is used for classification and attribution.

3.2.3.1 Vision Transformer Encoder (ViT/DeiT):

- The image patch embeddings $\mathbf{X}_{\text{patch}} \in \mathbb{R}^{N \times D}$ are prepended with a learnable `[CLS]` token:
$$\mathbf{X}_{\text{vit}} = [\mathbf{x}_{\texttt{[CLS]}}; \mathbf{X}_{\text{patch}}]$$
- The encoder applies L stacked layers of attention and FFNs:
$$\mathbf{H}^{(l)} = \texttt{MSA}(\mathbf{X}^{(l-1)}) + \mathbf{X}^{(l-1)}, \quad \mathbf{X}^{(l)} = \texttt{FFN}(\mathbf{H}^{(l)}) + \mathbf{H}^{(l)}$$
where `MSA` = Multi-Head Self-Attention and `FFN` = Feed-Forward Network with GELU activation:
$$\texttt{GELU}(x) = 0.5x \left(1 + \tanh\left[\sqrt{\frac{2}{\pi}}\left(x + 0.044715x^3\right)\right]\right)$$
- The final `[CLS]` token embedding is used for classification and explanation.

3.2.3.2 Textual Transformer Encoder (BERT):

- Each layer applies contextualization through multi-head attention. For each attention head, the input $\mathbf{X} \in \mathbb{R}^{L \times D}$ is linearly projected into:
$$\mathbf{Q} = \mathbf{X}W^Q, \quad \mathbf{K} = \mathbf{X}W^K, \quad \mathbf{V} = \mathbf{X}W^V$$
where:
  - $\mathbf{Q}$ (**Query**): Encodes what each token wants to attend to.
  - $\mathbf{K}$ (**Key**): Encodes the content of each token that might be attended to.
  - $\mathbf{V}$ (**Value**): Contains the actual information to be aggregated based on the attention weights.
- The self-attention scores are computed as:
$$\texttt{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \texttt{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}$$
where:
  - $\mathbf{d_k}$ is the dimensionality of keys and queries, used for scaling.

  This formula computes similarity between queries and keys, producing a weight distribution over values.
- The final embedding of the `[CLS]` token represents the semantic summary of the text, conditioned on the task.

This module outputs powerful, context-rich representations for both image and text, which are used in the next stage to compute **token-level or patch-level attributions** using **Layer-wise Relevance Propagation (LRP)**.

3.2.4 LRP Attribution Module

The **Layer-wise Relevance Propagation (LRP)** module is the core explainability engine of our system, designed to produce **class-conditioned, faithful, and human-aligned** attributions over both **textual** and **visual** inputs.

3.2.4.1 Theoretical Basis: Deep Taylor Decomposition

LRP is rooted in the theory of **Deep Taylor Decomposition**, where the prediction score $f_c(\mathbf{x})$ is approximated as a first-order Taylor expansion. The objective is to redistribute this score layer-by-layer back to the input, ensuring conservation:

$$\sum_i R_i = f_c(\mathbf{x})$$

This makes LRP distinct from naive gradient methods which may not preserve this conservation property.

3.2.4.2 LRP for Vision Transformers (ViT/DeiT)

- **Initialization:**
$$R^{(L)} = f_c(\mathbf{x}) \cdot \mathbf{1}_{[\texttt{CLS}]}$$
- **Propagation Rule:** Relevance is propagated backward through the layers using either the γ-rule or ε-rule:
$$R_i = \sum_j \frac{z_{ij} + \gamma z_{ij}^+}{\sum_k z_{kj} + \gamma \sum_k z_{kj}^+ + \varepsilon \cdot \text{sign}(\sum_k z_{kj})} R_j$$
where:
  - $z_{ij} = x_i w_{ij}$ is the contribution of neuron i to j
  - $\gamma > 0$ emphasizes positive contributions
  - $\varepsilon > 0$ stabilizes small denominators
- **Output:**
$$\text{Heatmap}_{\text{patch}} = \text{reshape}(R_{\text{patches}})$$
is used to form visual overlays on the input image.
- **Example:**

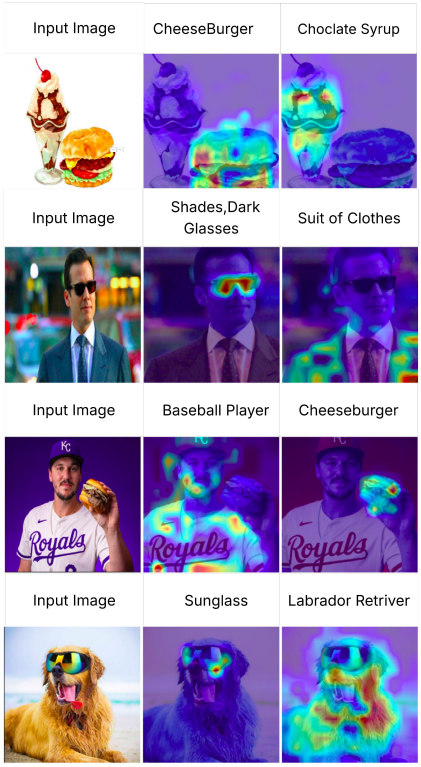

Figure 4: **Class-conditioned visual explanations using DeiT and LRP.**

3.2.4.3 LRP for Text Transformers (BERT)

- **Initialization:**
$$R^{(L)} = f_c(\mathbf{x}) \cdot \mathbf{1}_{[\texttt{CLS}]}$$
- **Propagation:** Relevance flows backward through self-attention and feedforward blocks:
$$R_i^{\text{text}} = \sum_j \frac{z_{ij}}{\sum_k z_{kj} + \varepsilon} R_j, \quad z_{ij} = x_i w_{ij}$$

- **Token Attribution Output:**

$$\text{Score}_{\text{token}_i} = R_i$$

  Tokens with higher scores are highlighted in the textual interface.
- **Example:**

Figure 5: **Token-level class-conditioned explanations on textual inputs.** Each row shows a different sentence, with words color-coded by sentiment class: negative and positive. The intensity of each color reflects the importance (relevance score) assigned by the model to that token, as computed using Layer-wise Relevance Propagation (LRP). These attributions illustrate how the model distributes attention across words when forming class-specific predictions, enhancing transparency and interpretability in text classification.

3.2.5 Comparative Advantages of LRP

Table 1: **Comparison** of popular **explainability methods** based on key **interpretability criteria**. **Class-conditioning** indicates if the explanation adapts based on the target class. **Saturation-resilience** reflects robustness against gradient saturation. **Faithfulness** measures how accurately the explanation reflects the model's true decision process.

Method	Class-Cond.	No Saturation	Faithful	Output
Gradient ∇f	×	×	Low	N/A
Grad-CAM	✓	×	Medium	Heatmap
LRP	✓	✓	High	Heatmap / Tokens

3.2.6 Summary of LRP Benefits

- **Class-Conditioned Attribution**: Only highlights what is relevant for the predicted class.
- **Robust to Saturation**: Works even when gradients vanish or explode.
- **Faithful Explanation**: Relevance decomposition obeys conservation principles.
- **Cross-Modal Consistency**: Same LRP logic works for both **images** and **text**.

3.3 Visual and Textual Explanation Mapping

After computing relevance scores using the **Layer-wise Relevance Propagation (LRP)** module, the next step is to generate interpretable outputs for both modalities:

3.3.1 Visual: Heatmap Generator

The **Heatmap Generator** converts relevance values across image patches into class-specific visual explanations:

- Each patch P_i receives a relevance score R_i propagated from the ViT/DeiT model.
- These patch-level relevance scores are reshaped into a 2D spatial layout:
$$\mathbf{R}_{\text{patch}} \in \mathbb{R}^{h \times w}$$
where h, w represent the number of patches along the height and width of the image.
- We perform **upsampling** using bilinear interpolation to match the original image resolution:
$$\mathbf{R}_{\text{image}} = \text{Upsample}(\mathbf{R}_{\text{patch}})$$
- The resulting heatmap is then **overlayed** on the original input using OpenCV:
$$\text{Overlayed Image} = \alpha \cdot \text{Image} + (1 - \alpha) \cdot \text{Colormap}(\mathbf{R}_{\text{image}})$$
with blending coefficient $\alpha \in [0, 1]$.

3.3.2 Token Attribution Mapper

The **Token Attribution Mapper** highlights the most influential tokens for the target class based on LRP relevance scores:

- Each token t_i from the input sequence receives a relevance score R_i, forming:
$$\mathbf{R}_{\text{text}} = [R_1, R_2, ..., R_n], \quad \mathbf{R}_{\text{text}} \in \mathbb{R}^n$$
- Tokens are visualized by:
  - **Coloring:** Green for positive contributions and red for negative.
  - **Opacity:** Proportional to $|R_i|$ to show strength.
- Output format is flexible:
  - HTML-based visualizations (interactive)
  - JSON outputs for integration with external dashboards

4 Result and it's interpretation

4.1 Training Strategy

To develop an interpretable and robust model for both **visual** and **textual modalities**, we adopt a carefully designed **staged fine-tuning strategy**. We begin by initializing the architecture with **pretrained transformer backbones—ViT and DeiT** for images and **BERT** for text. In the **initial phase**, the entire transformer backbone is **frozen**, and only the **task-specific classification head** is trained. This allows the model to quickly specialize in the downstream task without disturbing the rich, pretrained representations.

After a few epochs, we progressively **unfreeze the higher transformer layers** in stages, a process known as **layer-wise unfreezing**. Specifically, we unfreeze the top 2–3 layers at a time after fixed intervals (e.g., every 5 epochs). This gradual unfreezing is accompanied by **differential learning rates**, where newly unfrozen layers are trained with a **lower learning rate** to avoid abrupt weight updates. This technique helps in **mitigating catastrophic forgetting** and ensures **stable convergence**.

For models incorporating a **custom explainability module (XAI)**, we support both **joint training** and **separate training** regimes. In the **joint setup**, the base model and XAI module are trained simultaneously with a **composite loss** that combines **task accuracy** and **explanation fidelity**. This is achieved by balancing the **cross-entropy loss** for classification and a **mean squared error loss** for attribution alignment using a **tunable weight parameter** (λ).

The training process is optimized using the **AdamW optimizer**, a **cosine learning rate scheduler** with **warmup**, **gradient clipping**, and **mixed-precision training** for efficiency. The full configuration is summarized in Table 2.

Validation accuracy curves and training loss curves for both the training types are presented in Figure *10*.

Table 2: **Hyperparameter Configuration** for **Joint vs. Separate Training** across **Modalities**

Hyperparameter	**Joint (Text)**	**Joint (Vision)**	**Separate (Text)**	**Separate (Vision)**
Optimizer	AdamW	AdamW	AdamW	AdamW
Base Learning Rate	1e-4	1e-4	2e-5	2e-5
XAI Module LR	1e-4	1e-4	1e-4	1e-4
Scheduler	Cosine w/ Warmup	Cosine w/ Warmup	Cosine w/ Warmup	Cosine w/ Warmup
Weight Decay	0.01	0.01	0.01	0.01
Batch Size	32	32	16	16
Epochs	25	30	10 (base), 15 (XAI)	10 (base), 20 (XAI)
Loss Function	CrossEntropy + XAI Loss	CrossEntropy + XAI Loss	CE + MSE/Custom	CE + MSE/Custom
Explanation Loss Weight (λ)	0.3	0.2	N/A	N/A
Dropout Rate	0.1	0.1	0.1	0.1
Warmup Ratio	0.1	0.1	0.1	0.1
Max Gradient Norm	1.0	1.0	1.0	1.0
Unfreezing Strategy	Gradual	Gradual	Frozen base	Frozen base
XAI Module Init	Random	Random	Random	Random
Early Stopping Patience	5	5	5	5
Validation Frequency	1 epoch	1 epoch	1 epoch	1 epoch

4.2 Quantitative Metrics and its Interpretation

We quantitatively assess the effectiveness of our explainability modules using four key metrics across both textual (BERT-based) and visual (ViT/DeiT-based) modalities. The results for both **separate** and **joint training** strategies are summarized in Tables 3, 4, and 5.

4.2.1 Text Modality (BERT)

As shown in Table 3, the jointly trained model achieves a **fidelity score of 0.87**, significantly higher than the 0.81 score of the separately trained counterpart. This suggests that the explanation module trained jointly with the base model better captures the features used for the final prediction. The **comprehensiveness improves from 0.75 to 0.79**, meaning

that removing the most important tokens has a greater impact on the model's confidence — validating the relevance of the highlighted explanations.

Importantly, the **sufficiency drops from 0.42 to 0.35**, indicating that just the highlighted tokens alone are more sufficient for accurate prediction in the joint setup. Furthermore, **sparsity increases from 0.61 to 0.66**, reflecting more concise and focused explanations.

Table 3: **Quantitative Evaluation Metrics** for **Textual Explanations (BERT-based)**

Training Strategy	**Fidelity ↑**	**Comprehensiveness ↑**	**Sufficiency ↓**	**Sparsity ↑**
Separate Training	0.81	0.75	0.42	0.61
Joint Training	**0.87**	**0.79**	**0.35**	**0.66**

4.2.2 Vision Modality (ViT)

Table 4 presents metrics for the ViT model. Joint training improves **fidelity from 0.78 to 0.85**, indicating better alignment between visual attributions and model decisions. **Comprehensiveness increases from 0.72 to 0.79**, showing the highlighted regions are indeed crucial for prediction.

Sufficiency decreases from 0.46 to 0.38, meaning fewer visual patches are needed for accurate prediction, and **sparsity rises from 0.60 to 0.65**, reflecting more focused explanations.

Table 4: **Quantitative Evaluation Metrics** for **ViT Visual Explanations**

Training Strategy	**Fidelity ↑**	**Comprehensiveness ↑**	**Sufficiency ↓**	**Sparsity ↑**
Separate Training	0.78	0.72	0.46	0.60
Joint Training	**0.85**	**0.79**	**0.38**	**0.65**

4.2.3 Vision Modality (DeiT)

Table 5 shows explainability metrics for the DeiT model. Joint training leads to **fidelity improvement from 0.74 to 0.81** and **comprehensiveness rise from 0.70 to 0.75**, demonstrating enhanced explanation quality.

Similarly, **sufficiency drops from 0.50 to 0.40** and **sparsity improves from 0.56 to 0.59**, further confirming the benefit of joint optimization for concise and faithful explanations.

Table 5: **Quantitative Evaluation Metrics** for **DeiT Visual Explanations**

Training Strategy	**Fidelity ↑**	**Comprehensiveness ↑**	**Sufficiency ↓**	**Sparsity ↑**
Separate Training	0.74	0.70	0.50	0.56
Joint Training	**0.81**	**0.75**	**0.40**	**0.59**

4.2.4 Overall Interpretation.

Across all three models, **joint training consistently improves fidelity and comprehensiveness**, indicating explanations better capture the models' true decision processes. The reduction in sufficiency values under joint training implies that explanations alone are more informative for accurate predictions. Moreover, increased sparsity values reflect more concise, focused explanations, improving interpretability.

Between the vision models, ViT demonstrates generally higher explainability metrics than DeiT, likely due to architectural and data-efficiency differences. These results highlight the advantage of end-to-end training for producing both effective and interpretable explanations in multimodal transformer models.

4.3 Sample Examples

4.3.1 Sample Outputs for Vision and Textual Inputs

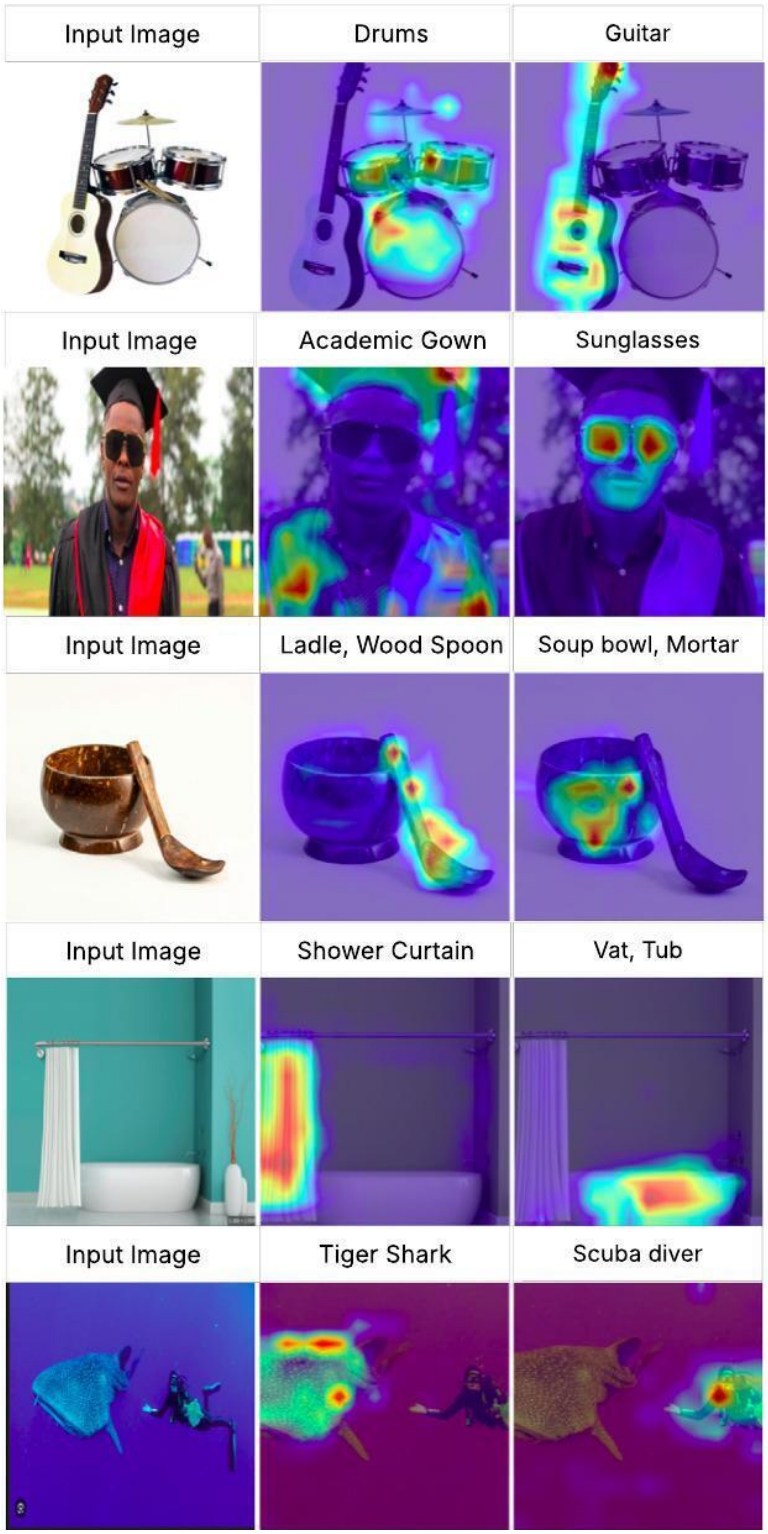

Figure 6: **Sample outputs on visual inputs on a variety of classes.**

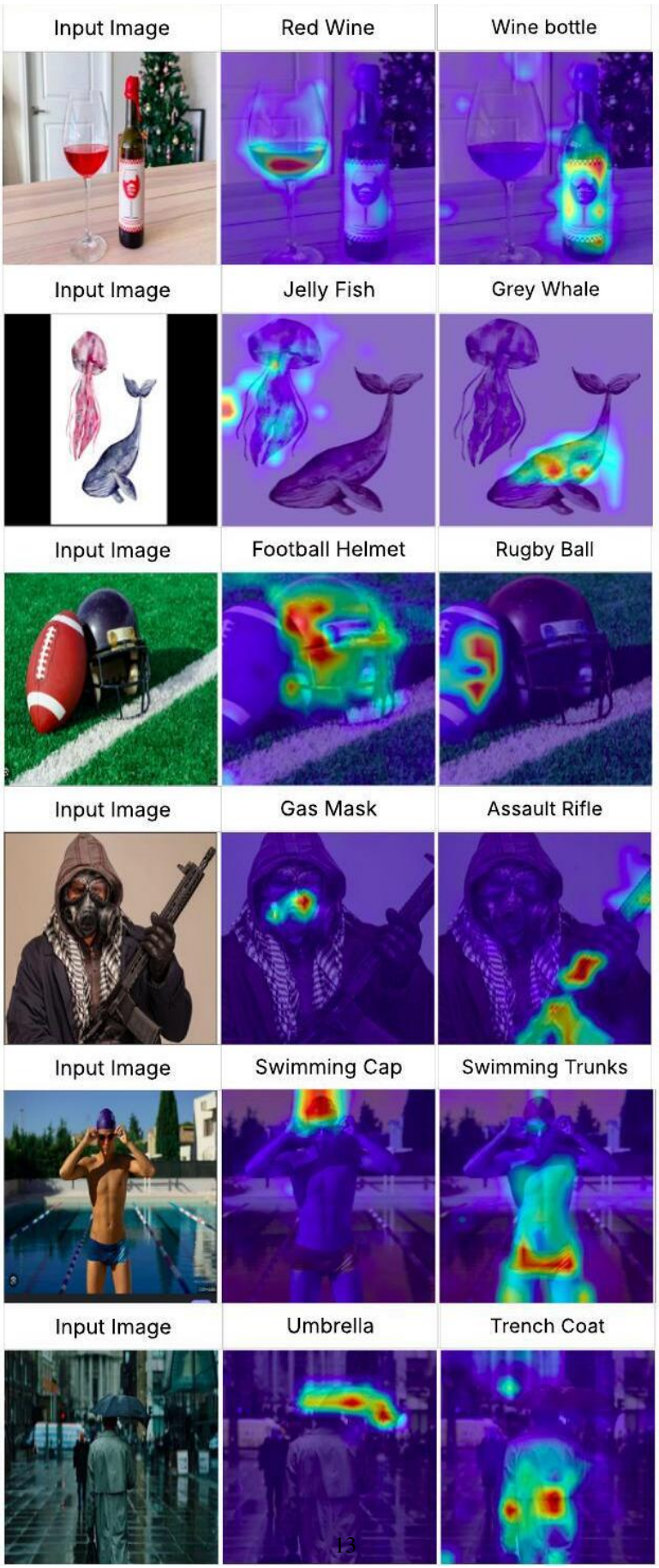

Figure 7: **Sample outputs on visual inputs on a variety of classes.**

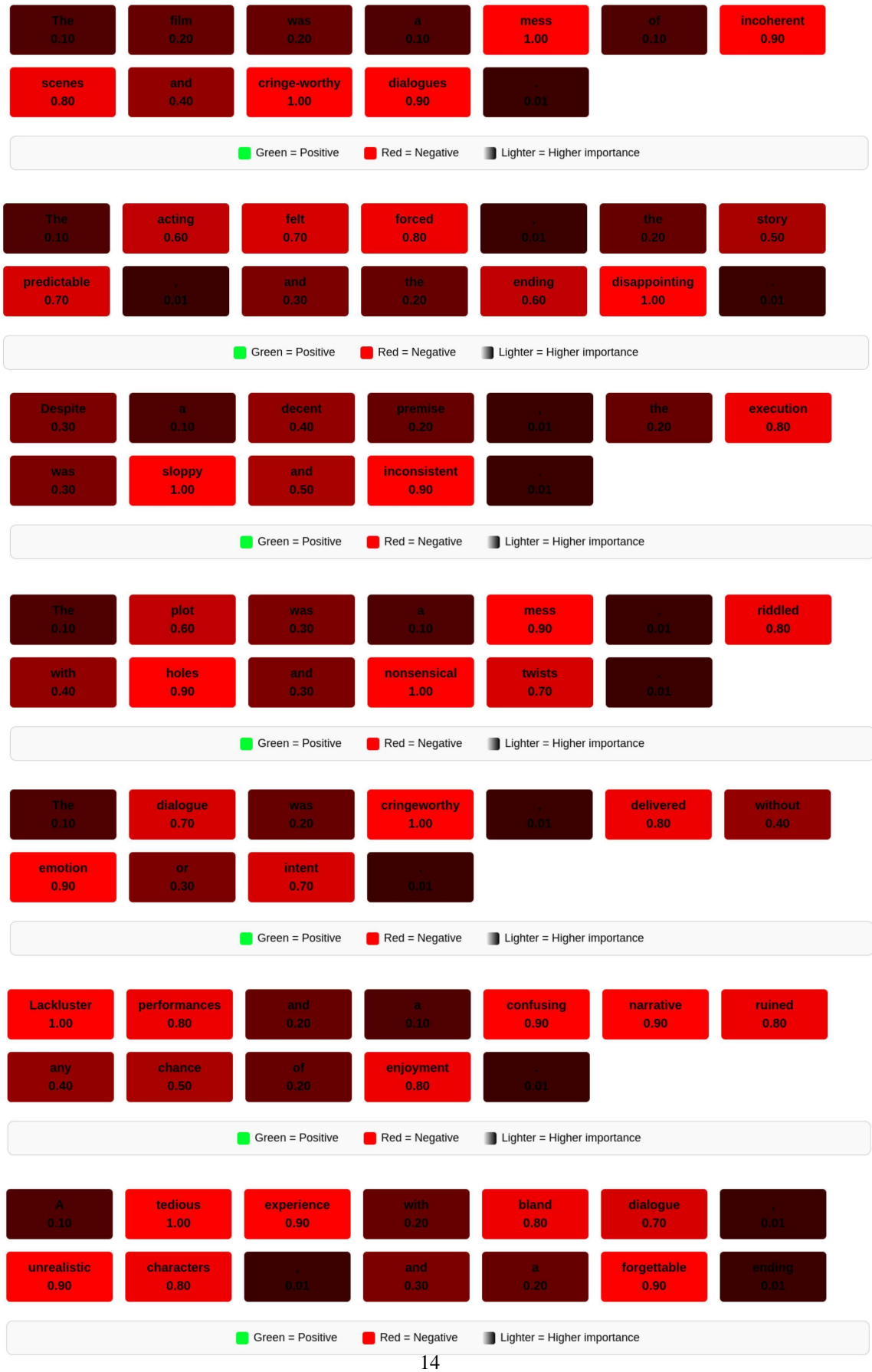

Figure 8: Sample outputs on **texts with negative connotations.**

Figure 9: Sample outputs on **texts with positive connotations.**

4.4 Training Loss and Validation accuracy curves

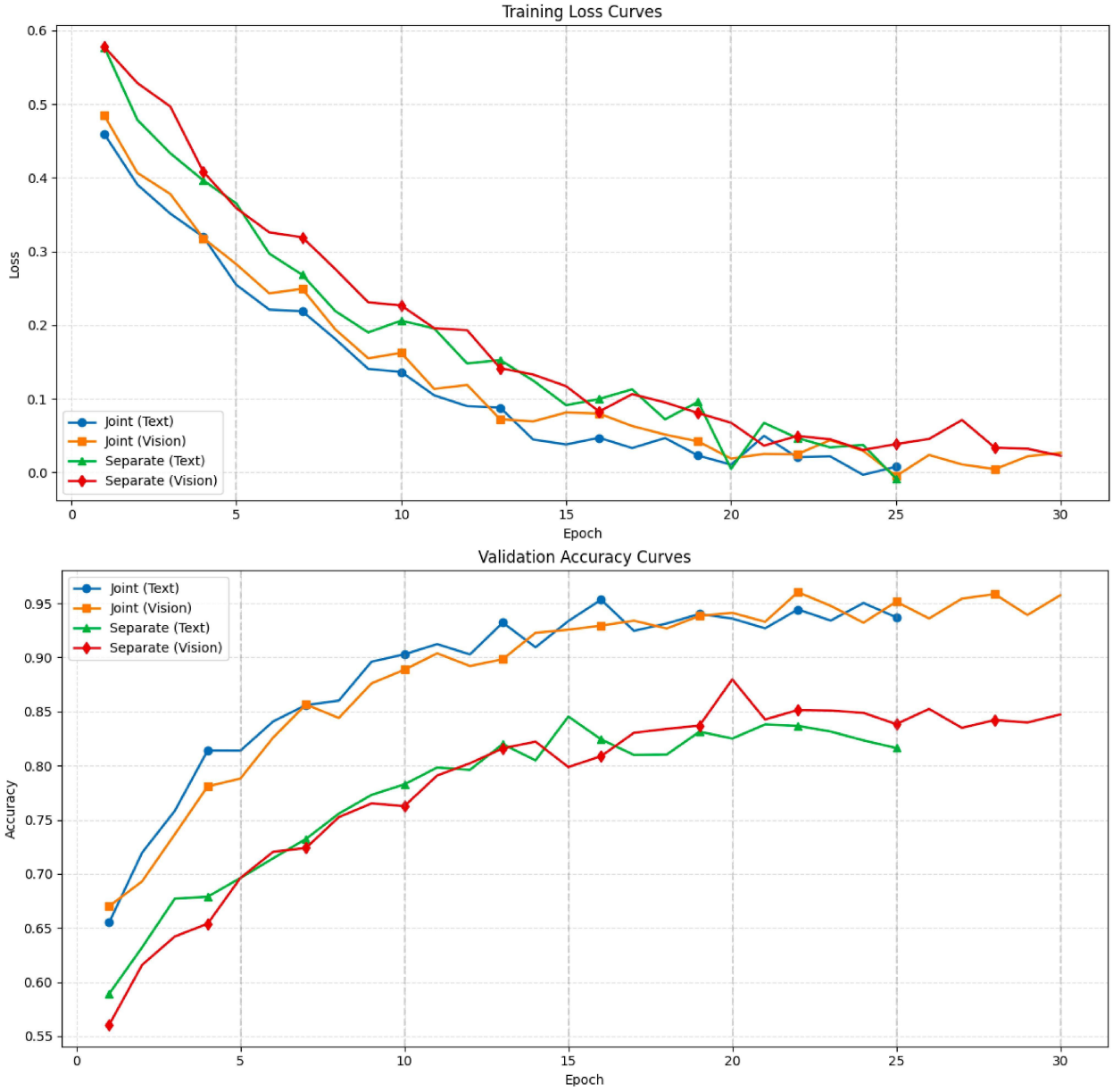

Figure 10: **Training Loss and Validation accuracy curves for both Joint and Separate training strategies.**

References

[1] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations (ICLR)*, 2021.

[2] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International Conference on Machine Learning (ICML)*, pp. 10347–10357, PMLR, 2021.

[3] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on*

Computer Vision (ICCV), pp. 618–626, 2017.

[4] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2921–2929, 2016.

[5] H. Chefer, S. Gur, and L. Wolf, "Transformer interpretability beyond attention visualization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 782–791, 2021.

[6] A. Binder, G. Montavon, S. Lapuschkin, K.-R. Müller, and W. Samek, "Layer-wise relevance propagation for deep neural network architectures," *Information Sciences*, vol. 364-365, pp. 219–241, 2016.

[7] J. Li, X. Chen, E. Hovy, and D. Jurafsky, "Visualizing and understanding neural models in nlp," in *Proceedings of NAACL-HLT*, pp. 681–691, 2016.

[8] J. Vig, "Analyzing the structure of attention in a transformer language model," in *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 63–76, 2019.

[9] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144, 2016.

[10] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 4765–4774, 2017.

[11] S. Jain and B. C. Wallace, "Attention is not explanation," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 3543–3556, 2019.