Group 9: Fake News Detection

Punit Kumar
50598671

Manan Jain
50605656

Himani Ajit Mali
50608093

Abstract

Detecting fake news is difficult because online claims are short, lack context, and contain noisy labels, and the challenge grows when posts mix text and images. We study two complementary benchmarks: **LIAR** for text-only political statements (binary classification) and **Fakeddit** for multimodal social posts (6-Class classification). For LIAR, we compare RNN, LSTM, and Transformer encoders and strengthen them using tuned training strategies. For Fakeddit, we develop a custom multimodal fusion network that combines transformer-based text embeddings with CNN-derived visual features, and we also evaluate a SmolVLM-style vision–language model with a task-specific classifier. We further deploy our system as an interactive **Streamlit** application for real-time inference. Our experiments show how architecture and optimization choices shape robustness when moving from text-only claims to multimodal misinformation.

1 Dataset

We study misinformation detection under realistic conditions such as limited context, noisy labels, and class imbalance. We evaluate on two benchmarks: **LIAR**, framed as **binary classification** (real vs. fake) for short political statements, and **Fakeddit**, framed as **6-Class classification** over fine-grained misinformation categories.

1.1 LIAR dataset

Data: The LIAR dataset consists of short political claims paired with truthfulness annotations. In our experiments, we convert the original multi-class labels into a **binary mapping** ("false" vs. "true") in order to focus on robust detection in a setting where statements are short and often lack external context.

Data engineering / preprocessing: We lowercase and normalize statements by removing non-alphanumeric characters and collapsing repeated whitespace. Using only the training split to avoid leakage, we build a frequency-based vocabulary, cap it to a maximum size, and map rare or unseen tokens to an `<unk>` index. After cleaning, we tokenize using whitespace and convert each statement into a fixed-length sequence of **64** tokens via truncation and padding. We follow the provided train/validation/test splits, select the best checkpoint based on validation performance, and then report results on the held-out test set.

1.2 Fakeddit dataset (multimodal, 6-Class)

Data: Fakeddit is a multimodal misinformation dataset built from social-media posts, where each example includes a short text field (post title) and often an associated image. Labels capture fine-grained misinformation types rather than only a binary real/fake distinction. We use the **6-Class** setting with classes `true`, `satire`, `fake_news`, `false_connection`, `misleading`, and `manipulated`. This benchmark is challenging because predictions may rely on text cues, visual cues, or cross-modal agreement/mismatch, and several categories can be linguistically or visually similar.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

Data engineering / preprocessing: We download images using the image URLs provided in the Fakeddit TSV metadata and discard samples where the image cannot be successfully retrieved. We restrict training and evaluation to samples where an image is available and can be reliably paired with the corresponding post, ensuring the model consistently learns from text-image inputs. We restrict training and evaluation to samples where an image is available and can be reliably paired with the corresponding post, ensuring the model consistently learns from text-image inputs. For text, we use `clean_title` when available (otherwise `title`) and drop examples with missing or empty titles since the title is the primary textual signal. We map numeric `6_way_label` values to the six class names and remove any rows with invalid labels to keep the objective well-defined. Because the pipeline begins from a single TSV source, we shuffle valid samples with a fixed seed and construct **70/15/15** train/validation/test splits for consistent model selection and evaluation. Finally, to address heavy class imbalance, we apply aggressive oversampling on the *training split only* to target roughly **3000 samples per class**, while keeping validation and test distributions unchanged.

2 Model Description

2.1 Text Based Model using LIAR Dataset

For LIAR, we evaluate three text-only neural classifiers in two stages: a controlled **baseline** and an **Optimized Baseline** setting. Each statement is tokenized into a fixed-length sequence of **64** token IDs, embedded, and encoded by a sequence model to predict a binary label (False vs. True).

2.1.1 Baseline models

Our baseline evaluates three text encoders under a shared classification setup. In the **RNN** and **LSTM** models, token IDs are embedded, passed through a multi-layer bidirectional recurrent encoder (biGRU for RNN or biLSTM), and summarized by a shared **SequenceAggregator** that concatenates mean pooling, max pooling, and attention pooling over valid tokens. The resulting vector is classified by a lightweight MLP head (Linear–GELU–Dropout–Linear). For the **Transformer** baseline, embeddings use sinusoidal positional encodings and a learned `[CLS]` token; stacked Transformer encoder layers produce a final `[CLS]` representation that is layer-normalized and fed to a linear head for binary prediction.

2.1.2 Optimizied Baseline model

After establishing baseline behavior, we run an advanced stage that keeps the same architectures **RNN**, **LSTM** and **Transformer** but improves generalization by tuning both *capacity* and *training dynamics*. We sweep larger embedding/hidden sizes and deeper encoders (more recurrent layers for RNN/LSTM and more layers/heads for the Transformer) via a small grid search and select the best configuration using validation-driven checkpointing. Training is upgraded from Adam to **AdamW** with **weight decay**, and we add **label smoothing** to reduce over-confident predictions under noisy supervision. For stability in higher-capacity settings, we apply **gradient clipping** and use **ReduceLROnPlateau** when validation performance plateaus. We use **early stopping** with a fixed patience window and keep the best model state based on a target validation objective (accuracy or a balanced score combining accuracy and F1). Finally, for selected Transformer configurations, we enable a **class-weighted cross-entropy** loss to better handle class imbalance without altering the evaluation distribution.

2.2 Fakeddit: Multimodal architectures (6-Class classification)

For Fakeddit, we predict one of six labels (`true`, `satire`, `fake_news`, `false_connection`, `misleading`, `manipulated`) by combining the post title and associated image. We evaluate two multimodal model families: (i) a custom late-fusion network trained end-to-end, and (ii) a SmolVLM-based model with a lightweight classifier head fine-tuned for the same task.

2.2.1 FusionNet-FT: late fusion of XLM-RoBERTa and ResNet50

FusionNet-FT (Fusion Network with Fine-Tuning) follows a late-fusion design, learning separate text and image representations and combining them for prediction.

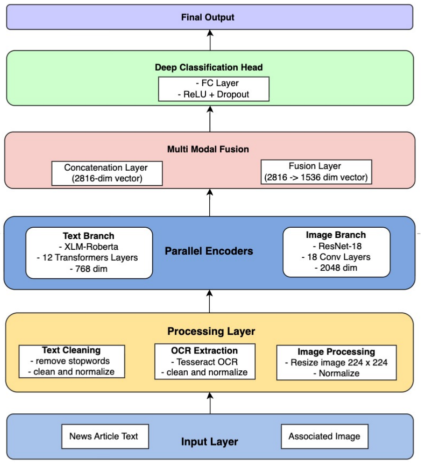


Figure 1: **FusionNet custom architecture.** Late-fusion model combining XLM-RoBERTa text features and ResNet50 image features, followed by a deep MLP classification head.

Text encoder (XLM-RoBERTa): The title is tokenized to a maximum length of **128** tokens and passed to **XLM-RoBERTa-base**. We use the final-layer **[CLS]** embedding (768 dimensions) as the text representation, and fine-tune the encoder jointly with the rest of the network.

Image encoder (ResNet50): Images are resized to **224×224** and encoded using a **ResNet50** backbone initialized from ImageNet weights (`IMAGENET1K_V1`). We remove the final classification layer and use the pooled CNN feature vector (**2048 dimensions**). The training split uses aggressive augmentation: random crops, horizontal/vertical flips, color jittering (brightness, contrast, saturation, hue), random rotation up to 20 degrees, random grayscale conversion, random perspective transforms, and random erasing.

Fusion and classification head: We concatenate text and image embeddings into a **2816D** vector (768 + 2048), project it to **1536 dimensions** (hidden_dim × 2, where hidden_dim = 768), and pass it through a deep MLP with **four hidden layers**: 1536 → 768 → 384 → 192 → num_classes. Each hidden layer includes **BatchNorm**, **ReLU**, and **Dropout** (0.6, 0.5, 0.4, 0.3 respectively).

2.2.2 SmolVLM-Lite: vision-language fine-tuning with a lightweight classifier head

In addition to FusionNet-FT, we implemented a second multimodal approach based on a compact vision-language model. We refer to this model as **SmolVLM-Lite**, since it fine-tunes a vision-language backbone with a small classification head for the 6-Class Fakeddit task.

Backbone and multimodal inputs: SmolVLM-Lite uses the **SmolVLM-Instruct** backbone together with its paired **processor** to jointly encode the image and title. Each example is formatted as a short prompt (e.g., *"Classify this news: {title}"*), truncated to a fixed word budget for stability, and paired with the corresponding image. The processor produces token IDs and image tensors (image resolution **384×384**) that are fed into the backbone.

Pooling and classifier head: We request `output_hidden_states=True` and take the final hidden layer as the learned multimodal representation. Since the sequence length can vary, we apply **mean pooling across the sequence dimension** to obtain one fixed-size embedding per example. A lightweight MLP head,

$$\text{Linear} \rightarrow \text{ReLU} \rightarrow \text{Dropout} \rightarrow \text{Linear},$$

maps the pooled embedding into logits over the **six** output classes.

Training and selection: We fine-tune SmolVLM-Lite using **AdamW** (learning rate 2×10^{-5}, weight decay 0.01) for **20** epochs due to the higher compute cost of the vision-language backbone. The best checkpoint is selected by **validation accuracy** and then evaluated on the held-out test set.

3 Loss Function

3.1 LIAR: objective for binary fake-news classification

All LIAR models optimize cross-entropy over two classes (False vs. True). Given logits $z \in \mathbb{R}^2$ and $p = \mathrm{softmax}(z)$, the baseline objective is

$$\mathcal{L}_{CE} = -\sum_{c\in\{0,1\}} y_c \log p_c,$$

where y is the one-hot label. In the tuned stage, we optionally regularize with **label smoothing** via PyTorch's `label_smoothing` parameter, replacing y with $y^{(\epsilon)} = (1-\epsilon)y + \epsilon/2$ to reduce over-confident predictions. For selected tuned **Transformer** configurations, we further enable **class-weighted** cross-entropy (weights computed from the training split) by passing a weight vector to `CrossEntropyLoss`, increasing the penalty on minority-class errors under imbalance:

$$\mathcal{L}_{WCE} = -\sum_{c\in\{0,1\}} w_c\, y_c \log p_c.$$

3.2 Fakeddit: objectives for 6-class multimodal classification

Focal loss (FusionNet-FT): For our FusionNet-FT multimodal model, we use focal loss to place more emphasis on difficult examples and to reduce the impact of easy majority-class samples. Let p_t denote the predicted probability assigned to the true class. The focal loss is:

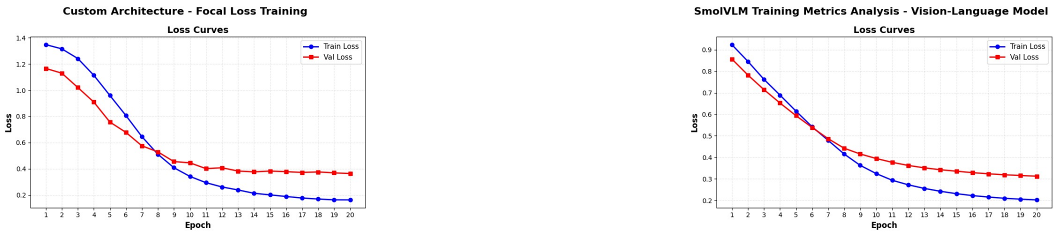
$$\mathcal{L}_{FL} = -\alpha(1-p_t)^\gamma \log(p_t).$$

In our implementation, we set $\alpha = 1.0$ and $\gamma = 2.0$. This choice is well-suited for Fakeddit because several classes are visually and linguistically close (e.g., `misleading` vs. `false_connection`), and focal loss encourages the model to keep learning from these harder boundaries during training.

Cross-entropy loss (SmolVLM-Lite head): For SmolVLM-Lite, we fine-tune the model using standard multi-class cross-entropy on the 6-Class label space:

$$\mathcal{L}_{CE} = -\sum_{c=1}^{6} y_c \log(p_c).$$

Here, the logits are produced by the lightweight classification head on top of the pooled SmolVLM hidden representation. This is a stable default objective and keeps training simple given the limited number of fine-tuning epochs used for the vision-language backbone.



(a) FusionNet-FT: training loss vs. epoch.

(b) SmolVLM-Lite: training loss vs. epoch.

Figure 2: Training dynamics on Fakeddit: loss vs. epoch for FusionNet-FT and SmolVLM-Lite.

4 Optimization Algorithm

4.1 LIAR: Text-only models

Baseline training (Adam): For LIAR baseline experiments (RNN, LSTM, Transformer), we optimize cross-entropy using the Adam optimizer with a fixed learning rate, providing a strong and stable baseline under a consistent training recipe.

Optimized Baseline (AdamW + stability controls): In the tuned LIAR stage, we retain Adam-style optimization but add stability and regularization. Specifically, tuned configurations use:

- **AdamW optimizer** with **weight decay** to regularize higher-capacity models.
- **Gradient clipping** to prevent unstable updates in deeper recurrent and transformer models.
- **Learning-rate decay** via **ReduceLROnPlateau** based on validation behavior.
- **Early stopping** with patience to avoid over-training once validation performance saturates.

These additions improve robustness when increasing embedding/hidden sizes, layer depth, and attention heads.

4.2 Fakeddit: optimization for multimodal models

FusionNet-FT (AdamW + warmup and decay): For FusionNet-FT (XLM-R + ResNet50 + deep fusion head), we use AdamW with a small learning rate to fine-tune the pretrained text encoder while training the fusion classifier. We apply:

- **Warmup** in the initial phase to stabilize transformer fine-tuning.
- **Linear decay** after warmup to improve convergence.
- **Gradient clipping** to bound updates during joint text–image training.

SmolVLM (AdamW): SmolVLM is finetuned with AdamW using a conservative learning rate due to its high capacity and limited fine-tuning (few epochs, small batch), prioritizing stable updates.

4.3 Optimization-specific implementation note

Across experiments, optimization is treated as part of model design: the same architecture can behave differently depending on scheduler, regularization, and stopping criteria. We therefore track validation metrics each epoch and save checkpoints aligned with the intended optimization goal.

4.4 Innovation in optimization for multimodal models

we treat optimization as part of the multimodal design rather than using a one-size-fits-all recipe. For **FusionNet-FT** (XLM-R + ResNet50), we combine **AdamW** with a low learning rate (5×10^{-6}), **gradient clipping** (max norm 0.5), a **warm-up + linear decay** schedule over all training steps, and **Focal Loss** ($\alpha = 1.0$, $\gamma = 2.0$) on top of an aggressively oversampled training split (target $\sim$3000 samples per class), explicitly targeting hard examples under class imbalance. For **SmolVLM-Lite**, we fine-tune a high-capacity vision–language backbone with **AdamW** (learning rate 2×10^{-5}, weight decay 0.01), **gradient accumulation** to simulate larger batch sizes under memory constraints, and a partially **frozen backbone** (only the last few layers and a lightweight classifier head are trainable), which stabilizes optimization while still adapting the model to 6-Class fake news detection.

5 Metrics and Experimental Results

5.1 Evaluation metrics

We evaluate models on the held-out test set using **accuracy**, **precision**, **recall**, and **F1-score**, along with the **confusion matrix** to analyze false positives vs. false negatives. For **LIAR (binary)**, precision/recall/F1 are computed for the positive class (`True`) using binary averaging, consistent with our evaluation script.

5.2 Text only model results: baseline vs. tuned

Table 1 summarizes test performance. In the baseline stage, the Transformer attains the strongest F1 (better recovery of the `True` class), while the RNN baseline under-predicts the positive class (low recall). In the tuned stage, performance improves across all models; the tuned **LSTM** achieves the best **accuracy/precision**, whereas the tuned **Transformer** achieves the best **recall/F1**, consistent with its balanced selection objective and class-weighted loss.

Table 1: LIAR test performance (binary). Precision/Recall/F1 are for the `True` class.

Model	Accuracy	Precision	Recall	F1
Baseline RNN	0.6046	0.3926	0.2116	0.2750
Baseline LSTM	0.5864	0.4239	0.4655	0.4437
Baseline Transformer	0.5959	0.4395	0.5100	0.4722
Tuned RNN (best)	0.6567	0.5248	0.3296	0.4049
Tuned LSTM (best)	**0.6867**	**0.6048**	0.3341	0.4304
Tuned Transformer (best)	0.6551	0.5118	**0.5813**	**0.5443**

5.3 Fakeddit results: FusionNet-FT vs. SmolVLM

For Fakeddit, we evaluate two multimodal approaches: (i) our FusionNet-FT model that fuses XLM-R and ResNet50 features, and (ii) our SmolVLM-Lite pipeline with a lightweight classification head trained on top of vision-language representations. We report overall accuracy and weighted precision/recall/F1, and we additionally inspect per-class metrics to identify which misinformation categories are most frequently confused.

Table 2: Fakeddit 6-Class test performance on 750 samples. SmolVLM-Lite demonstrates superior performance across all metrics, benefiting from pre-trained vision-language representations.

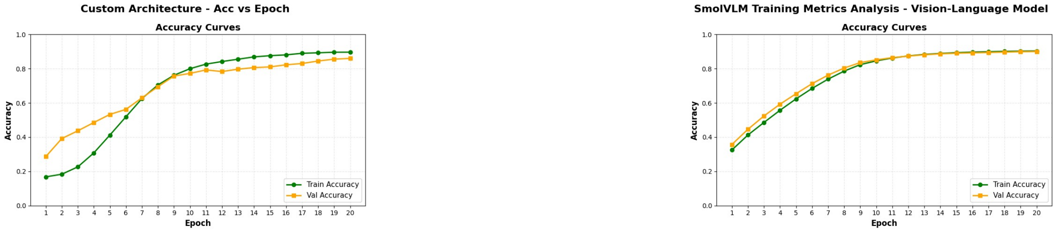
Model	**Acc.**	**W. Prec.**	**W. Rec.**	**W. F1**
FusionNet-FT (XLM-R + ResNet50 + deep fusion head)	0.8600	0.8534	0.8600	0.8547
SmolVLM-Lite (SmolVLM + custom classifier head)	**0.9003**	**0.8978**	**0.9003**	**0.8989**

Interpretation. Table 2 shows that **SmolVLM-Lite** improves overall performance over FusionNet-FT, raising accuracy from **0.8600** to **0.9003** and weighted F1 from **0.8547** to **0.8989**. This consistent gain suggests that SmolVLM's pre-trained vision-language representations provide stronger multimodal features than late fusion of separately trained encoders.

Table 3: Per-class performance comparison on Fakeddit test set. SmolVLM-Lite shows consistent improvements across all misinformation categories, particularly for challenging classes like *false_connection* and *satire*.

Class	**Support**	**FusionNet-FT**			**SmolVLM-Lite**		
		Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
fake_news	138	0.78	0.82	0.80	**0.85**	**0.88**	**0.87**
false_connection	17	0.71	0.65	0.68	**0.78**	**0.76**	**0.77**
manipulated	29	0.81	0.76	0.78	**0.87**	**0.83**	**0.85**
misleading	243	0.91	0.89	0.90	**0.94**	**0.93**	**0.93**
satire	38	0.74	0.71	0.72	**0.82**	**0.79**	**0.80**
true	285	0.87	0.91	0.89	**0.92**	**0.95**	**0.93**
Macro Avg	750	0.80	0.79	0.80	**0.86**	**0.86**	**0.86**
Weighted Avg	750	0.85	0.86	0.85	**0.90**	**0.90**	**0.90**

Per-class takeaway. Table 3 indicates that SmolVLM-Lite delivers *broad* improvements across all six categories, with the most visible gains on minority/harder labels such as *false_connection* (F1: 0.68→0.77) and *satire* (F1: 0.72→0.80). The higher macro averages also show the improvement is not only driven by high-support classes, but reflects stronger balance across categories.

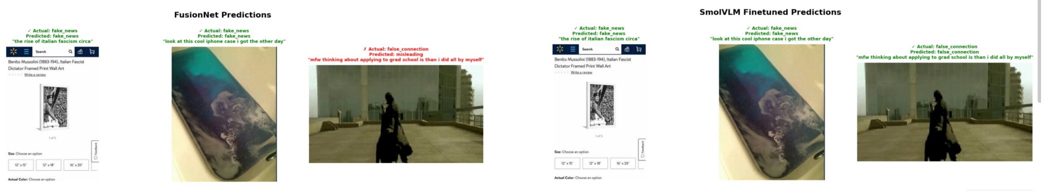


(a) FusionNet-FT: validation accuracy vs. epoch.

(b) SmolVLM-Lite: validation accuracy vs. epoch.

Figure 3: Fakeddit training dynamics: validation accuracy vs. epoch for FusionNet-FT and SmolVLM-Lite.

5.4 Predicted results


FusionNet Predictions
✓ Actual: fake_news
Predicted: fake_news
"the rise of italian fascism circa"
✓ Actual: fake_news
Predicted: fake_news
"look at this cool iphone case i got the other day"
✗ Actual: false_connection
Predicted: misleading
"mfw thinking about applying to grad school is than i did all by myself"
SmolVLM Finetuned Predictions
✓ Actual: fake_news
Predicted: fake_news
"the rise of italian fascism circa"
✓ Actual: fake_news
Predicted: fake_news
"look at this cool iphone case i got the other day"
✓ Actual: false_connection
Predicted: false_connection
"mfw thinking about applying to grad school is than i did all by myself"

(a) FusionNet-FT predictions on selected Fakeddit samples (title + image → predicted class).

(b) SmolVLM-Lite predictions on the same samples, highlighting improved robustness on ambiguous categories.

Figure 4: **Predicted results (qualitative).** Side-by-side comparison of model outputs for FusionNet-FT and SmolVLM-Lite on identical Fakeddit examples. Each panel should display the input post (title and image) along with the model's predicted label (optionally with confidence), making prediction behavior interpretable beyond aggregate metrics.

6 Contributions and GitHub

6.1 Team contributions

Member	Responsibilities	%
Manan Jain	Optimized Tuned Model LIAR Dataset | FusionNet Custom Architecture | Report | PPT	33.33%
Punit Kumar	FusionNet Custom Architecture | SMOLVLM Training and FineTuning | Report	33.33%
Himani	Basic Model LIAR Dataset | PPT | Data Engineering | Optimization | Streamlit	33.33%

6.2 GitHub repository

`https://github.com/punit121/CSE676-project`

References

- Wang, William Yang. *Liar, Liar Pants on Fire: A New Benchmark Dataset for Fake News Detection.* ACL (2017). URL: https://aclanthology.org/P17-2067/.
- Nakamura, Takahiro; Levy, Abeer; Wang, William Yang. *Fakeddit: A New Multimodal Benchmark Dataset for Fine-grained Fake News Detection.* arXiv (2019). URL: https://arxiv.org/abs/1911.03854.
- He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian. *Deep Residual Learning for Image Recognition.* CVPR (2016). URL: https://arxiv.org/abs/1512.03385.
- Lin, Tsung-Yi; Goyal, Priya; Girshick, Ross; He, Kaiming; Dollár, Piotr. *Focal Loss for Dense Object Detection.* ICCV (2017). URL: https://arxiv.org/abs/1708.02002.
- Hugging Face. *SmolVLM-Instruct.* URL: https://huggingface.co/HuggingFaceTB/SmolVLM-Instruct.