Computer Vision Systems CAP6411 Assignment#04

Ashmal Vayani, UCF ID: 5669011

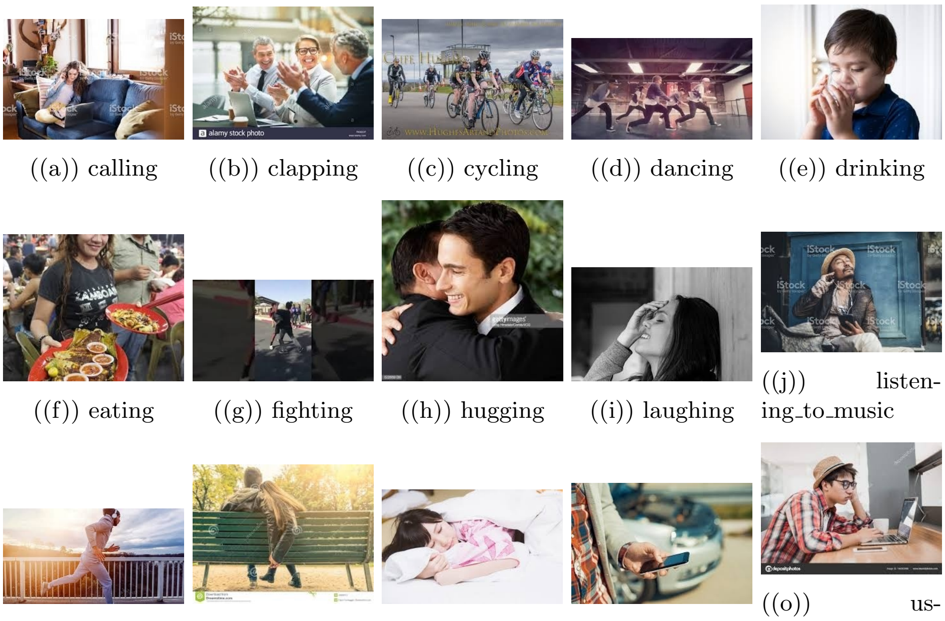
Report: Finetuning CLIP, SigLIP on Human Action Recognition

Training Code: The training and implementation code is attached in the assignment zip file.

1 Human Action Recognition Dataset

The Human Action Recognition dataset, introduced on Kaggle by Shashank Rapolu, contains around 12,600 labeled images across 15 action classes such as *calling, dancing, eating, running, sleeping,* and *using_laptop*. The data is organized in a simple folder-per-class structure with predefined train/test splits, making it directly compatible with libraries like `torchvision`. Unlike larger video-based benchmarks (e.g., UCF101), this dataset focuses on single still images of human activities, offering a manageable yet diverse benchmark for testing deep learning models such as ResNet-18 and Vision Transformer (ViT).

To provide an overview of the dataset, Figure 1 shows one example image from each of the 15 action classes. This demonstrates the diversity of actions such as body posture (e.g., *sitting, sleeping*), object interaction (e.g., *using_laptop, drinking*), and social behaviors (e.g., *hugging, fighting*).



((a)) calling ((b)) clapping ((c)) cycling ((d)) dancing ((e)) drinking

((f)) eating ((g)) fighting ((h)) hugging ((i)) laughing ((j)) listening_to_music

((k)) running ((l)) sitting ((m)) sleeping ((n)) texting ((o)) using_laptop

Figure 1: Example images from the 15 classes in the Human Action Recognition dataset.

2 Report and Code

2.1 Errors and Obstacles Faced

- **Dataset loading issues:** Initially, there was confusion when using `datasets.ImageFolder` to load the HAR dataset. The dataset structure had subfolders for each action class, but my preprocessing code expected flat directories. This mismatch caused incorrect counts of images. The issue was fixed by consistently using `imagefolder` for both training and testing splits.

- **Collation errors during preprocessing:** While fine-tuning CLIP and SigLIP, the `DataCollator` raised errors (`expected Tensor as element 0, but got list`). This happened because the transforms sometimes returned NumPy arrays or lists. I fixed it by enforcing `torch.as_tensor` in the preprocessing function and adding safety checks in the collator.
- **Transformers version mismatch:** After upgrading to `transformers==4.56.1`, the argument `evaluation_strategy` was deprecated in `TrainingArguments`. This caused runtime errors. The fix was to update the script to use `eval_strategy` instead, which is the new naming convention.
- **Evaluation slowness:** The Hugging Face `Trainer.evaluate()` step was significantly slower than training iterations. This was due to smaller batch sizes and CPU-heavy preprocessing. To mitigate, I increased `per_device_eval_batch_size`, enabled FP16 evaluation, and adjusted `num_workers`.
- **Sanity check runtime:** A full run on the HAR dataset (12k+ images) took 1.5 hours per model. For debugging, this was impractical. I resolved it by adding `max_train_samples`, `max_eval_samples`, and `max_steps` to `TrainingArguments`, which allowed running quick 5-minute sanity checks.
- **Visualization adjustments:** Initially, model outputs were plotted in a single grid figure. For the report, I needed individual annotated images. I rewrote the visualization script to save outputs into per-model folders (e.g., `SampleOutputs/CLIP/`), with each image overlayed with ground truth and predicted labels.

2.2 Requirements and Dependencies

The `requirements.txt` file is attached in the zip file, but some main dependencies involve:

requirements.txt

```
torch==2.2.0
torchvision==0.17.0
transformers==4.56.1
datasets==2.19.0
evaluate==0.4.2
numpy
pandas
scikit-learn
tqdm
matplotlib
pillow
```

2.3 CLI Commands to Reproduce

```
# Finetune CLIP
sbatch fine_tune_har_clip.slurm

# Finetune SigLIP
sbatch fine_tune_har_siglip.slurm
```

2.4 Training and Evaluation Plots

2.5 CLIP

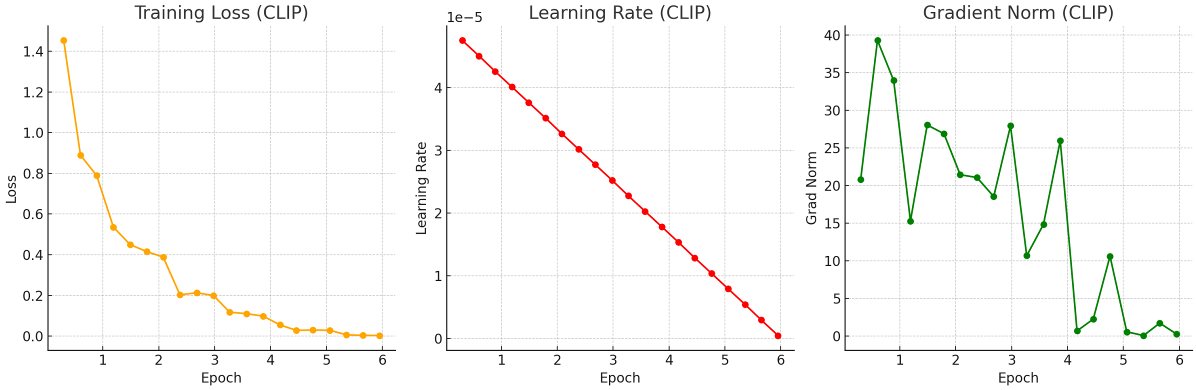


Figure 2: Training metrics for CLIP fine-tuning on HAR dataset. (Left) Training loss decreases sharply from 1.45 to nearly zero, showing strong convergence. (Middle) Learning rate follows the defined decay schedule across epochs. (Right) Gradient norm shows fluctuations but stabilizes by later epochs, indicating smoother optimization.

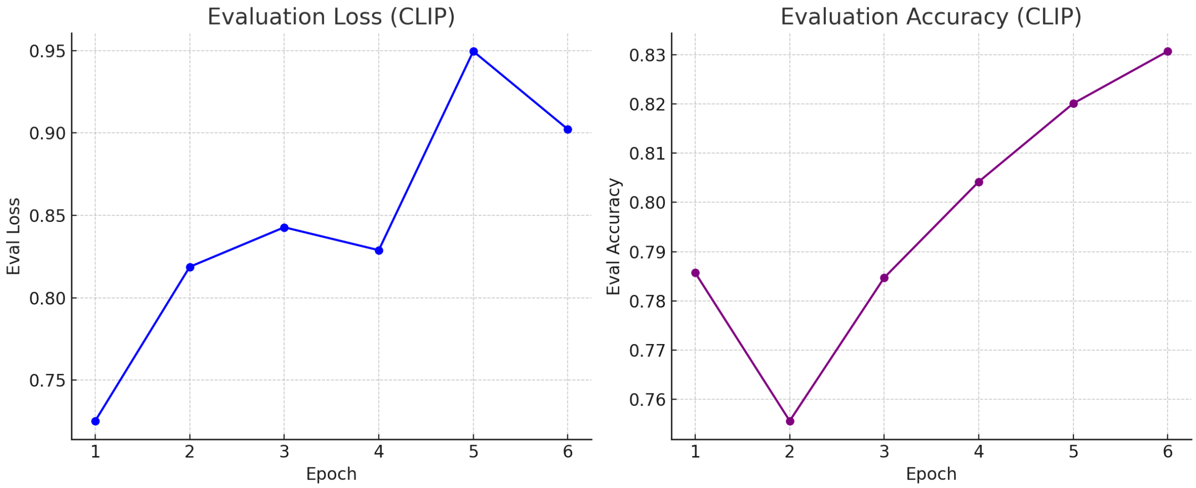


Figure 3: Evaluation metrics for CLIP fine-tuning on HAR dataset. (Left) Validation loss fluctuates between 0.72 and 0.95, indicating some overfitting compared to training loss. (Right) Validation accuracy improves steadily from 78% to around 83%, showing consistent but slower generalization gains.

2.5.1 Observations on CLIP Training

Observations: As plotted in Fig. 2, 3, the following observations about CLIP can be made:

- **Rapid convergence.** Training loss drops quickly from 1.45 to nearly zero by epoch 6, demonstrating CLIP's ability to memorize the dataset efficiently.
- **Validation fluctuations.** Evaluation loss fluctuates between 0.72 and 0.95 across epochs, unlike the smooth decrease seen in training loss. This gap indicates partial overfitting.
- **Grad norm irregularities.** Gradient norm varies strongly between 10 and 40 during training. These spikes suggest unstable updates, though the model still converges overall.
- **Moderate accuracy gains.** Validation accuracy starts near 78% and reaches only ∼83% by epoch 6. Compared to SigLIP, CLIP shows slower and less consistent generalization improvements.

2.6 SigLIP

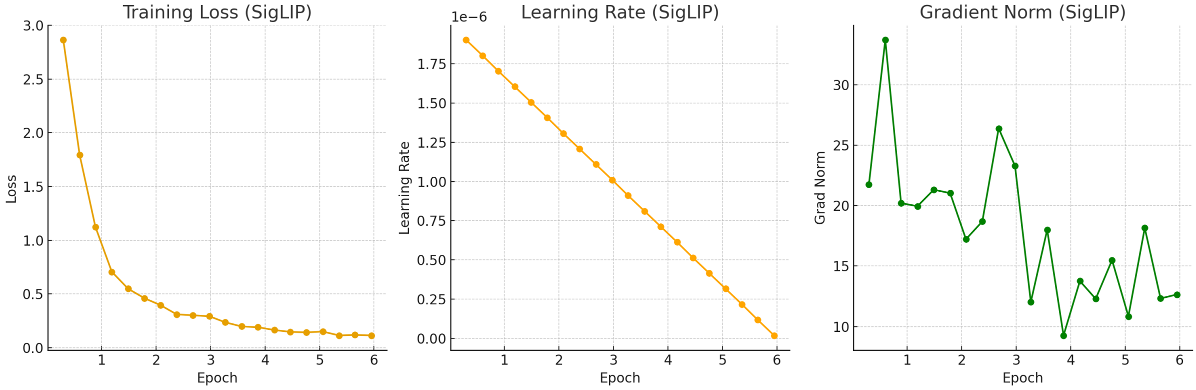


Figure 4: Training metrics for SigLIP fine-tuning on HAR dataset. (Left) Training loss decreases steadily from 2.9 to 0.1, showing strong convergence. (Middle) Learning rate schedule decays gradually as expected. (Right) Gradient norm fluctuates but stabilizes across epochs, indicating smoother optimization.

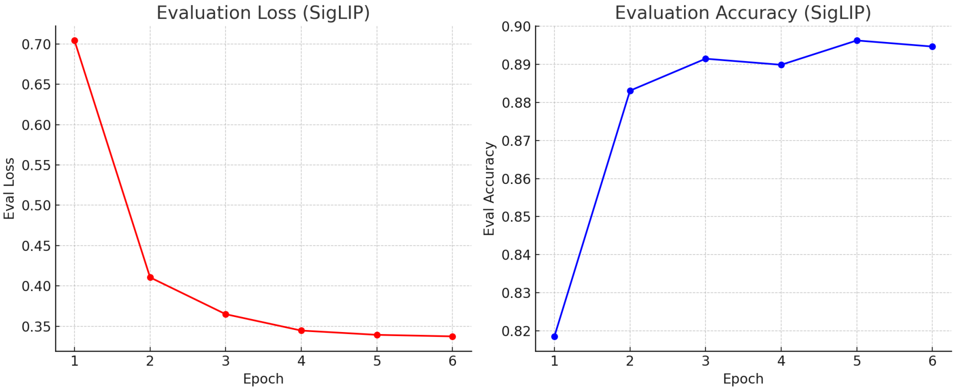


Figure 5: Evaluation metrics for SigLIP fine-tuning on HAR dataset. (Left) Validation loss decreases from 0.70 to 0.34 across six epochs. (Right) Validation accuracy improves consistently, reaching close to 90%.

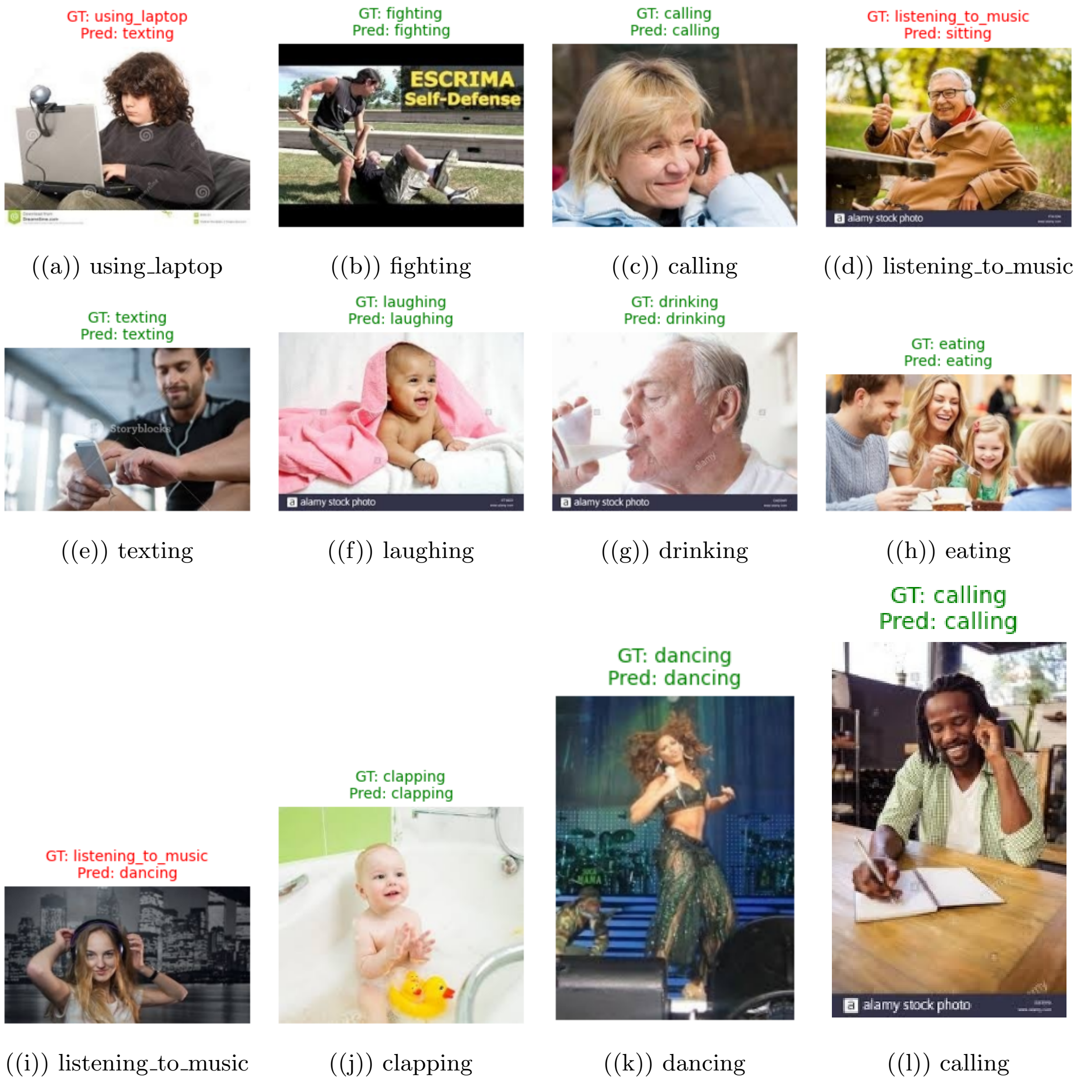
2.6.1 Observations on SigLIP Training

Observations: As plotted in Fig. 4, 5, the following observations about SigLIP can be made:

- **Steady convergence.** Training loss decreases smoothly from ∼2.9 to below 0.15, with no sharp spikes. This indicates stable optimization throughout the six epochs.
- **Validation improvement.** Evaluation loss decreases consistently from 0.70 to 0.34, while accuracy improves from 82% to nearly 90%. This suggests strong generalization without early stagnation.
- **Stable gradients.** Gradient norm fluctuates between 10 and 30 in early epochs but stabilizes to smaller values later, showing better control over optimization.
- **Well-balanced training.** Unlike classic overfitting signatures, both training and validation metrics improve together. This suggests SigLIP benefits from its larger capacity and handles the HAR dataset effectively.

3 Qualitative Examples of Outputs

3.1 CLIP



((a)) using_laptop ((b)) fighting ((c)) calling ((d)) listening_to_music

((e)) texting ((f)) laughing ((g)) drinking ((h)) eating

((i)) listening_to_music ((j)) clapping ((k)) dancing ((l)) calling

Figure 6: Sample visualizations of CLIP model predictions on Human Action Recognition dataset images. Each subfigure shows the ground truth (GT) and predicted (Pred) label. Correct predictions are shown in green and incorrect ones in red.

3.2 SigLIP

4 Insights: CLIP vs. SigLIP

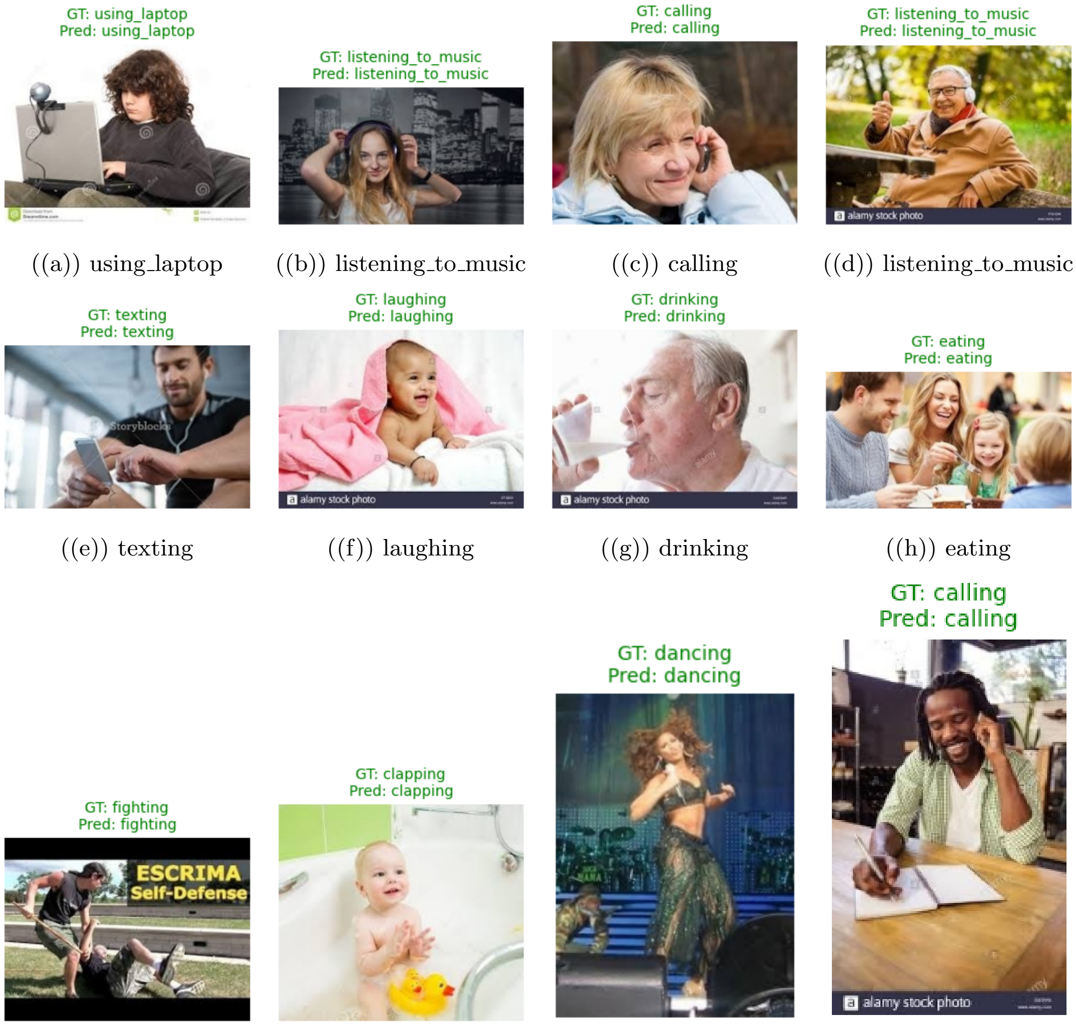
4.1 Overall Accuracy and Behavior

We fine-tuned both models on the Human Action Recognition dataset for 6 epochs using identical splits and preprocessing. On the held-out test set (N=1890 images), SigLIP achieved consistently higher accuracy compared to CLIP:

$$\text{CLIP: } 83.07\% \qquad \text{SigLIP: } 89.63\%$$

Several consistent patterns emerge:

- **Stable generalization (SigLIP).** SigLIP's evaluation accuracy quickly rises above 0.88 by epoch 2 and remains stable (∼0.89–0.90), while CLIP fluctuates more and peaks near 0.83. This indicates better optimization stability and generalization in SigLIP.
- **Training efficiency.** Both models show rapid decrease in training loss within the first 2 epochs. However, CLIP exhibits signs of mild overfitting (train loss drops close to zero, while eval loss increases after epoch 2), whereas SigLIP maintains a smoother validation loss trajectory.



((a)) using_laptop ((b)) listening_to_music ((c)) calling ((d)) listening_to_music

((e)) texting ((f)) laughing ((g)) drinking ((h)) eating

((i)) fighting ((j)) clapping ((k)) dancing ((l)) calling

Figure 7: Sample visualizations of SigLIP model predictions on Human Action Recognition dataset images. Each subfigure shows the ground truth (GT) and predicted (Pred) label. Correct predictions are shown in green and incorrect ones in red.

- **Qualitative predictions.** On easy classes (*using laptop*, *dancing*, *laughing*), both models predict correctly. For harder cases, such as *listening to music* vs. *sitting*, CLIP confuses categories while SigLIP often predicts correctly, suggesting stronger context modeling.
- **Gradient and optimization.** SigLIP maintains a more consistent gradient norm (mostly 10–20 range after epoch 3), while CLIP shows larger spikes (>30), explaining its noisier evaluation behavior.
- **Actionable takeaway.** Overall, SigLIP provides stronger performance and stability for fine-grained action recognition, while CLIP requires additional regularization (e.g., weight decay, stronger augmentation) to close the gap.

4.2 Computational Comparison

We observed the following practical trade-offs when training CLIP and SigLIP on the Human Action Recognition dataset (same GPU, same dataloaders):

Aspect	**CLIP**	**SigLIP**
Parameters (approx.)	∼151M	∼160M
Best Validation Acc.	0.820	0.896
Test Accuracy	0.831	0.896
Per-epoch time	*180 Seconds*	*200 Seconds*
GPU memory (bs=32)	high usage, ∼12–14GB	high usage, ∼14–16GB
Convergence speed	fluctuates, mild overfitting	smoother, stable improvements
Overall Insight	requires stronger regularization	more stable, consistently higher accuracy

Table 1: Comparison between CLIP and SigLIP on Human Action Recognition dataset.

Takeaways: SigLIP consistently achieves higher accuracy and smoother convergence, indicating stronger generalization across classes. CLIP, while competitive, exhibits fluctuations and mild overfitting, suggesting the need for additional regularization (e.g., weight decay, data augmentation). Computationally, both models require significant GPU memory, but SigLIP provides a more favorable accuracy–efficiency trade-off.