DeepDiet: Multimodal Deep Learning for Nutritional Content Estimation

Mini Rawat minir07@stanford.edu

Mark Gernitis gernitis@stanford.edu

Neetish Sharma neetishs@stanford.edu

CS230

1 Introduction

In this report, we present an implementation of Nutrition5k-based nutrition estimation using 3 modern computer vision backbones: CNN-LSTM, Swin CLS and ConvNeXt. The Multi-modal CNN-LSTM is a multi-branch fusion architecture that jointly learns portion estimation and nutrient density predictions in a single end-to-end trainable model, in contrast to the original Nutrition5k pipeline, which trains separate models for each task and combines predictions at inference. ConvNeXt[4] revisits classic convolutional network design, modernizing it to close the performance gap with vision transformers while retaining the simplicity and efficiency of ConvNets. Meanwhile, Swin-based models leverage hierarchical attention and shifted-window mechanisms[3] to capture both local and long-range dependencies in images, making them a strong backbone for dense visual tasks such as object detection, segmentation, and, in this context, nutritional estimation. By applying these backbones to Nutrition5k, this project explores how different architectural choices—classic convolution vs. LSTM vs hierarchical-transformer-inspired models—impact the task of food nutrition estimation. The goal is to assess their respective strengths and limitations, and to provide insights into which kinds of vision models are most suitable for nutrition-related tasks.

2 Related Work

Early work in automatic food analysis mainly addressed food recognition using datasets such as Food-101 [1] and FooDD [6], which enabled deep classifiers to identify dish categories but lacked nutritional annotations required for calorie or macronutrient estimation. As a result, these datasets supported recognition but could not be used for quantitative dietary assessment.

Nutrition5K [8] introduced the first large-scale dataset containing both visual data and high-quality nutritional measurements. It includes multi-view RGB videos, overhead RGB-D captures, detailed ingredient lists, per-ingredient masses, and dish-level calories and macronutrients. Their baseline Inception-V2 model demonstrated that regression from images can approach or exceed human nutritionists in calorie estimation, and further showed the value of depth information for robust portion prediction.

Recent works[5] extend this direction through improved sensing or modeling. Depth-prediction and fusion approaches replace explicit depth sensors by estimating geometry from RGB. 3D reconstruction methods infer food volume from single images, while energy-density maps estimate caloric contribution at the pixel level.

More recent studies[2] leverage Vision Transformers and multimodal models. Pre-trained ViTs have been evaluated on Nutrition5K and found competitive with the original JFT-trained CNN baseline. Emerging multimodal, instruction-tuned models integrate image understanding with reasoning about portion sizes and ingredients, suggesting a path toward more generalizable nutrition estimation systems.

3 Dataset and Features

3.1 Dataset Description

The experiments were conducted on the Nutrition5k dataset [8], which contains imaging from overhead RGB-D sensors and four rotating side-angle cameras of 5,006 real-world cafeteria meals, along with ground-truth nutritional annotations for five nutritional quantities: calories (kcal), mass (g), fat (g), carbohydrates (g), and protein (g). However, each dish does not contain all input types (see Table 6 and Table 10). We follow the official Nutrition5k train/test split based on dish IDs to avoid leakage across sets.

3.2 Input Modalities and Preprocessing

Overhead RGB: Single top-down 640×480 image capturing the dish from directly above, providing a canonical view for portion estimation.

Depth Map: Corresponding 640×480 depth image from an Intel RealSense sensor, encoding distance to the food surface. Depth provides volumetric cues that correlate with food mass.

Side Frames: Up to 120 frames per dish from 4 rotating cameras (A, B, C, D) positioned at table level, capturing the dish from 360° as cameras rotate. Original resolution is 1920×1080 (16:9 aspect ratio). We sample 16 frames total (4 per camera) to balance coverage with memory constraints.

Image Resizing: All images are resized to 288 pixels on the shorter edge, then center-cropped to 256×256. This crop size was chosen to balance detail preservation with GPU memory constraints when processing multiple frames.

4 Methods

4.1 Multi-modal CNN-LSTM architecture

This model uses a multi-branch encoder-fusion architecture for multi-task nutritional prediction. Each input is processed by a separate encoder, features are concatenated, and shared fusion layers produce predictions for five nutritional targets. An architecture diagram is shown in Figure 3.

Visual Encoders. We use EfficientNet-B0 [7] pretrained on ImageNet as the backbone for all visual encoders. The final classification layer is removed, exposing the 1280-dimensional feature vector from the global average pooling layer. For overhead RGB images $x_{\text{rgb}} \in \mathbb{R}^{3\times256\times256}$, the encoder produces features $f_{\text{rgb}} = \phi_{\text{rgb}}(x_{\text{rgb}}) \in \mathbb{R}^{1280}$. The depth encoder is identical but with a modified first convolutional layer to accept single-channel input $x_{\text{depth}} \in \mathbb{R}^{1\times256\times256}$.

Temporal Side-Frame Processing. For side-angle frames, we first extract per-frame features using the EfficientNet-B0 encoder. These features are processed by a bidirectional LSTM to capture cross-frame relationships. The final side representation is the mean of concatenated hidden states across all timesteps.

Feature Fusion. Each branch's features are concatenated and passed through fully-connected layers with ReLU activations and dropout for regularization.

Task-Specific Heads. Three separate heads predict: calories $\hat{y}_{cal}$, mass $\hat{y}_{mass}$, and macronutrients $(\hat{y}_{fat}, \hat{y}_{carb}, \hat{y}_{protein})$.

Loss Function. We minimize MAE with equal weights across all five targets: $\mathcal{L} = \sum_{i\in\mathcal{T}} \frac{1}{N}\sum_j |y_i^{(j)} - \hat{y}_i^{(j)}|$ [8].

4.2 The Cross-Swin CLS

This model employs a Feature Pyramid Network (FPN) with Swin Transformer backbone and cross-attention decoder. The architecture consists of: a Swin Transformer Tiny backbone (microsoft/swin-tiny-patch4-window7-224)[3] pretrained on ImageNet; an FPN for multi-scale feature extraction across 4 levels; a 4-layer cross-attention decoder with 8 heads and CLS token for global context; and an Multi-layer perceptron (MLP) regressor with 3072-dimensional hidden layers. RGB and depth inputs are concatenated into a 4-channel tensor, processed through the backbone and FPN, then decoded via cross-attention with positional embeddings to predict all five nutritional targets. Training configuration can be seen in Table 1.

Table 1: Cross-Swin-CLS Training Configuration

Parameter	Value
Learning rate	1e-4
Batch size	3
Weight decay	1e-4
Loss function	Multi-task L1 loss with Lp norm (p=0.8)
Optimizer	Adam with cosine annealing
Training epochs	40–41
Warmup epochs	10–15

4.3 ConvNeXt Nutrition Model

ConvNeXt is a 4-stage hierarchical ConvNet; we use a mostly-frozen ConvNeXt-Base backbone with a lightweight MLP regression head. A 1×1 fusion layer maps 4-channel RGB+depth input to 3 channels, followed by ConvNeXt feature extraction and adaptive pooling. Training uses differential learning rates: Run 1 unfreezes the last two stages, while Run 2 unfreezes three stages with slow backbone tuning (1e-5) and faster head updates (1e-4) to reduce overfitting. The head is a two-layer MLP with LN, ReLU, dropout, and category-specific regressors. Parameter grouping separates backbone (LR=1e-5) and head components (LR=1e-4) for efficient adaptation.

5 Experiments/Results/Discussion

5.1 Multi-Modal CNN–LSTM

We establish a baseline using a two-stream fusion architecture with separate EfficientNet-B0 encoders for overhead RGB and depth (hyperparameters in Table 7). Treating RGB and depth as separate modalities with dedicated encoders allows the network to learn complementary geometric and appearance cues more effectively than the naive channel concatenation used in the Nutrition5k model. Our baseline achieves competitive calorie prediction (49.9 vs 47.6 MAE) and substantially outperforms the "Depth as 4th Channel" approach from Nutrition5k on mass prediction (33.0 vs 40.7 MAE), approaching the more complex Volume Scalar pipeline (29.4 MAE). Macronutrient predictions lag behind the paper's results by $\sim 2\times$, likely because our end-to-end approach must jointly learn portion estimation and density prediction, whereas Nutrition5k used separate portion-independent density predictors. Per-target baseline metrics are provided in Appendix Table 8.

We then conducted three additional experiments (Exp 2 - Exp 4) to improve performance by incorporating side-angle video frames from the dataset's 4 rotating cameras.

Exp	Configuration	Best Val MAE	Δ
1	Overhead + Depth (baseline)	102.7	—
2	+ Side frames (BiLSTM)	134.5	+31%
3	+ Encoder freezing	131.9	+28%
4	+ Missing modality handling	218.3	+113%

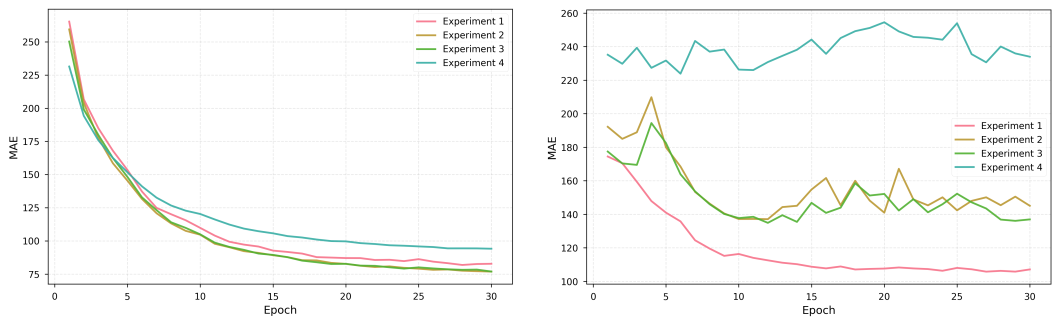
Table 2: Experiment summary. All side-frame variants degraded performance relative to the overhead+depth baseline.

Experiment 2 extended the baseline by adding 16 side-angle frames (4 per rotating camera, uniformly sampled across each camera's timeline for 360° coverage), aggregated via a bidirectional LSTM before fusion. Despite providing additional visual information from multiple viewpoints, side frames increased validation loss by 31%. The model exhibited severe overfitting, validation loss plateaued at epoch 17 while training loss continued to decrease (train-val gap of +65). The BiLSTM's 1.6M parameters combined with only 2,638 training dishes allowed the model to memorize training-specific frame patterns rather than learn generalizable food features.

Experiment 3 attempted to reduce overfitting by freezing pretrained EfficientNet encoders for the first 10 epochs, then fine-tuning with discriminative learning rates (encoders at 10$\times$ lower LR than other layers). These changes yielded only marginal improvement (131.9 vs 134.5 MAE). Training remained unstable with high variance, and the model showed similar overfitting patterns. Reducing capacity did not address the fundamental issue of side frames providing memorization shortcuts.

Experiment 4 increased training data by 53% (detailed in Appendix Table 9) (2,638 $\rightarrow$ 4,047 dishes) by allowing partial modality availability, using learned embeddings to substitute for missing inputs. However, performance degraded substantially (218.3 vs 131.9 MAE). When $\sim$32% of training samples lack overhead/depth, the strongest predictors from Experiment 1, the model must learn from weaker side-frame signals alone. The learned embeddings appeared to introduce noise rather than meaningful substitutes, suggesting data quantity cannot compensate for data quality.

Key Finding: Side-angle frames consistently degraded performance regardless of regularization, capacity reduction, or increased training data. The overhead camera provides a top-down view that proved well-aligned for portion estimation, while side angles introduce data that the model exploits for memorization rather than generalization.



(a) Training MAE (b) Validation MAE

Figure 1: Training curves across all experiments. Experiment 1 (baseline) shows stable convergence while side-frame variants exhibit overfitting with validation loss plateauing or increasing.

5.2 Cross-Swin-CLS and ConvNeXt Experimental Summary

5.2.1 Cross-Swin-CLS

We trained a Cross-Swin-CLS model as an alternative transformer-based baseline. We observe rapid learning in the Initial Phase (Epochs 0-5). Both losses drop sharply: 3.6-3.7 $\rightarrow$ 1.6-1.8 About 50-55% reduction in the first 5 epochs. In this phase, model learns basic patterns quickly and train and test losses are close, indicating good early generalization. During mid phase (Epochs 5-20), we continue to see steady improvement in train loss: smooth decline from 1.6 $\rightarrow$ 1.3 but start observing test loss being more volatile, oscillating between 1.4-1.6. We also see a gap emerging with test loss consistently higher than train loss. Model continues learning but with more variability on test data. In late epochs 20-40, there is a gradual decline from 1.3 $\rightarrow$ 0.85 in train loss. Test loss fluctuates around 1.1-1.2 with peaks and valleys. We can see the train-test gap widening with final gap 0.25 and test loss plateaus. The plot shows Cross-Swin-CLS demonstrating effective learning with 76% training loss reduction and good generalization with a 29% train-test gap. Appendix A.4.

5.2.2 ConvNeXt

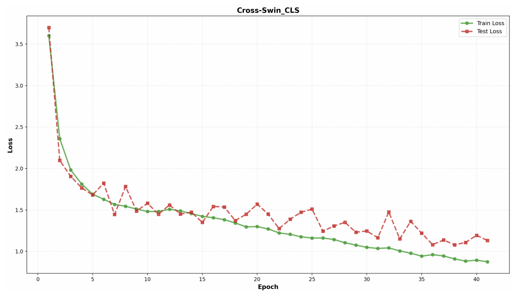
We evaluated ConvNeXt under two different training configurations (Run 1 and Run 2), using distinct learning-rate schedules. The plot shows four curves comparing two ConvNeXt training runs: Run 1 Train (blue solid line): 40 epochs, uniform LR=1e-4, unfreeze 2 layers, Run 1 Test (blue dotted line): corresponding test loss. Run 2 Train (orange dashed line): 20 epochs, differential LR (backbone 1e-5, head 1e-4), unfreeze 3 layers, Run 2 Test (orange dotted line): corresponding test loss. We see

Table 3: Cross-Swin-CLS: SMAPE and PMAE Results on Test Set

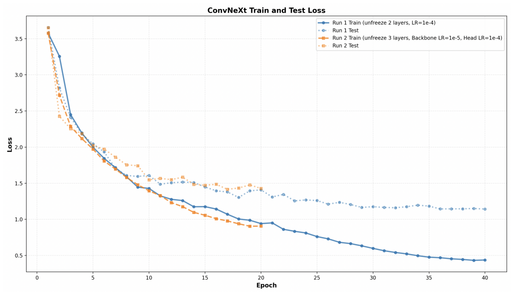
Category	**SMAPE (%)**	**PMAE (%)**
Calories	22.50	16.61
Mass	15.93	10.87
Fat	50.54	27.06
Carbohydrates	37.44	23.81
Protein	37.37	23.95
Average	**32.76**	**20.44**

Table 4: ConvNeXt : SMAPE Results on Test Set

Category	**SMAPE (%)**
Calories	32.13
Mass	22.77
Fat	60.53
Carbohydrates	44.88
Protein	54.19
Average	**42.90**



(a) Cross-Swin-CLS Train/Test Loss



(b) ConvNeXt Train/Test Loss

Figure 2: Comparison of training and test loss curves for Cross-Swin-CLS and ConvNeXt.

overfitting in Run 1. The problem is manifested as a large generalization gap (161%) indicating severe overfitting Evidence: Training loss drops to 0.44 Test loss plateaus at 1.14 Gap widens significantly after epoch 20. Cause: Uniform LR (1e-4) is too aggressive for backbone fine-tuning.

To reduce overfitting and achieve better Generalization in Run 2, we implemented differential LR as differential LR prevents aggressive backbone fine-tuning. This resulted in an improvement with the generalization gap reducing from 161% to 57%. Evidence: More stable test loss (less oscillations) was observed. Smaller relative gap with better balance between train and test errors was also observed.

Metric	**Cross-Swin-CLS**	**ConvNeXt - Run 1**	**ConvNeXt - Run 2**
Train Loss	0.8752	0.4381	0.9066
Test Loss	1.1314	1.1429	1.4276
Train-Test Gap	0.2562	0.7048	0.5210
Generalization Gap	29%	161% [Poor]	57.48% [Moderate]
Winner	Better generalization	Better Training fit	Faster convergence (1/2 the epochs)

Table 5: Comparison of Cross-Swin-CLS and ConvNeXt performance.

6 Conclusion

ConvNeXt achieves the lowest training loss of all models but also exhibits the strongest overfitting. Test losses remain comparable to Cross-Swin-CLS, indicating similar generalization potential. Differential learning rates in Optimized ConvNeXt significantly improve convergence and reduce the train–test gap from severe to moderate levels. Overall, ConvNeXt trains faster and generalizes better once hyperparameters are tuned.

Adding side-angle frames was a novel idea. The overhead camera provides a overhead view optimized for portion estimation, while side angles introduce inputs that the model exploits for memorization rather than generalization. We would explore ConvNeXt and transformer based models for side angle images for better portion estimation as future work.

Contributions

Mini Rawat: I contributed to the development and improvement of deep learning models for nutritional value estimation from food images using the Nutrition5k dataset. My primary contributions include: (1) Literature survey: Read various journal publications in the area of food calorie estimation from images and discussed approaches, challenges and implementation details with TA Bassem Akoush and my project partner Mark. (2) Model Architecture Development: Implemented and evaluated multiple transformer-based architectures including the Cross-Swin-CLS baseline model with Feature Pyramid Network (FPN), Swin Transformer backbone, and cross-attention decoder architecture, as well as a novel ConvNeXt-based nutrition model with efficient transfer learning strategies using frozen backbone layers and differential learning rates for fine-tuning; (3) Enhanced Regressor Design: Developed an advanced 'EnhancedRegressor' module incorporating batch normalization, residual connections, and category-specific regression heads optimized for multi-task learning, significantly improving prediction accuracy for challenging nutrients such as fat, carbohydrates, and protein; (4) Loss Function Innovation: Designed and implemented two novel loss functions—a weighted multi-task loss function with category-specific weights calibrated to address imbalanced prediction difficulty across nutrients, and a SMAPE-aware loss function that directly optimizes for the symmetric mean absolute percentage error metric used for evaluation; (5) Training Infrastructure: Created comprehensive training pipelines in AWS by using course credits to deploy a ML server with A100 GPU and ran multiple training runs each for many hours. (6) Experimental Analysis: Conducted extensive experiments comparing baseline and improved models, analyzing training dynamics, overfitting patterns, and generalization performance. I also documented all my work thoroughly for future development. Core implementation files at: https://github.com/minirawat/Nutrition5k. The full working copy is in AWS instance i-0806a4ae409f450de (AIMLServer) running Deep Learning OSS Nvidia Driver AMI GPU PyTorch 2.8 (Ubuntu 24.04) 20251101 AMI. Cursor was used as a collaborator for research and development for this project.

Mark Gernitis: My code can be viewed at: https://github.com/gernim/deepdiet. I contributed to the development of multi-modal deep learning models for nutritional estimation using the Nutrition5k dataset. My primary contributions include: (1) Multi-Modal Architecture Development: Implemented a two-stream fusion architecture with separate EfficientNet-B0 encoders for overhead RGB and depth modalities, achieving competitive baseline performance with the Nutrition5k reference models. (2) Side Frame Integration: Extended the baseline to incorporate side-angle video frames from 4 rotating cameras, implementing a BiLSTM temporal aggregator and camera-diverse frame sampling strategy to ensure 360° coverage. (3) Transfer Learning Optimization: Developed encoder freezing with discriminative learning rates, freezing pretrained encoders during early epochs then fine-tuning with reduced learning rates to mitigate overfitting. (4) Missing Modality Handling: Implemented learned embedding substitution for missing modalities, enabling training on dishes with partial sensor availability and increasing usable training data by 53%. (5) Training Infrastructure: Deployed training pipelines on Google Cloud Platform, running experiments on V100 GPUs with TensorBoard logging for experiment tracking. (6) Experimental Analysis: Conducted systematic ablation studies analyzing the impact of side frames, regularization strategies, and data augmentation on model generalization. PyCharm AI Assistant was used as a collaborator for research, development, and report writing for this project.

Neetish Sharma: did not contribute to this report.

References

[1] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision (ECCV)*, pages 446–461. Springer, 2014.

[2] Manal Chokr and Shady Elbassuoni. Calories prediction from food images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, pages 4664–4669, 2017.

[3] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, 2021.

[4] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11976–11986, 2022.

[5] Weiqing Min, Shuqiang Jiang, Linhu Liu, Yong Rui, and Ramesh Jain. A survey on food computing. *ACM Computing Surveys*, 52(5):1–36, 2019.

[6] Parisa Pouladzadeh, Abdulsalam Yassine, and Shervin Shirmohammadi. Foodd: Food detection dataset for calorie measurement using food images. In *New Trends in Image Analysis and Processing – ICIAP 2015 Workshops*, volume 9281 of *Lecture Notes in Computer Science*, pages 441–448. Springer, 2015.

[7] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 6105–6114. PMLR, 2019.

[8] Quin Thames, Arjun Karpur, Wade Norris, Fangting Xia, Liviu Panait, Tobias Weyand, and Jack Sim. Nutrition5k: Towards automatic nutritional understanding of generic food. *arXiv preprint arXiv:2103.03375*, 2021. Dataset: `https://github.com/google-research-datasets/Nutrition5k`.

A Appendix

A.1 Data

Split	Dishes	Modalities Available	Source
Train	4,059	RGB: 4,059 / Depth: 2,758 / Side: 3,930	Cafe 1
Test	709	RGB: 709 / Depth: 507 / Side: 676	Cafe 1

Table 6: Dataset splits following the official Nutrition5k paper partitions. Not all dishes have all modalities available.

A.2 Preprocessing for ConvNext and Swin

As input features, we fuse RGB and depth by concatenating them into a 4-channel tensor, then projecting to 3 channels via a 1×1 convolution for ConvNeXt-based models, or processing RGB and depth streams separately for Swin-based models before multi-scale fusion.

A.3 Multi-modal CNN-LSTM

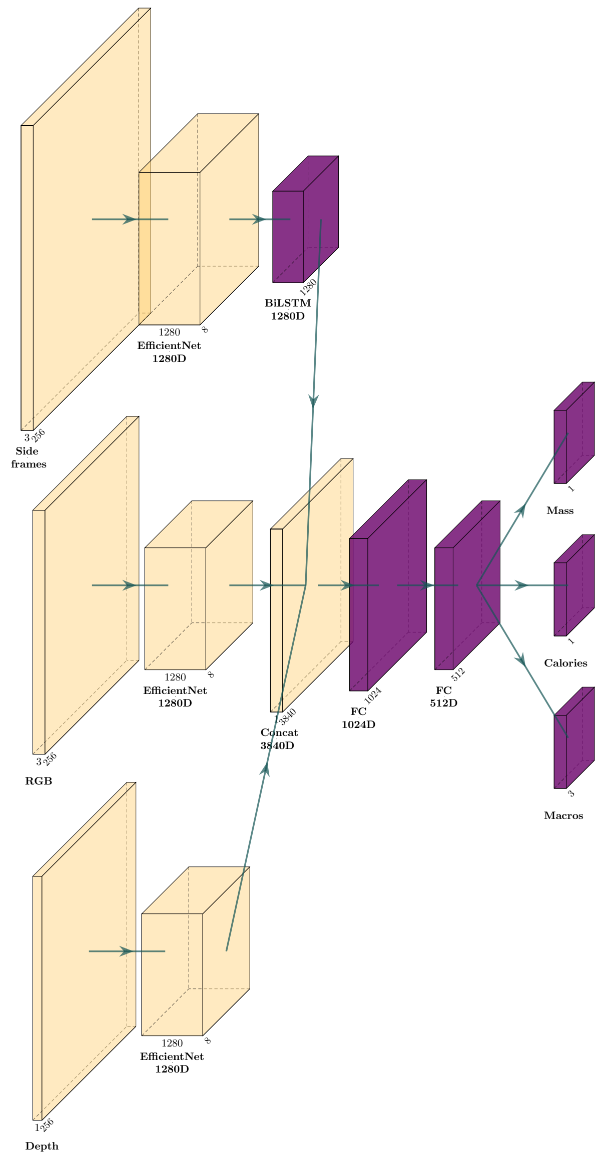


Figure 3: Overview of Multi-modal CNN-LSTM network architecture for our learning experiments.

Table 7: Baseline Hyperparameters

Hyperparameter	Value	Rationale
Learning rate	1e-4	Standard for fine-tuning pretrained CNNs
Batch size	8	Memory constraint (8 frames $\times$ EfficientNet)
Optimizer	Adam	Adaptive learning rates, standard for vision
Weight decay	1e-4	L2 regularization to reduce overfitting
LR schedule	0.5$\times$ every 5 epochs	Gradual decay for convergence
Gradient clipping	max_norm=5.0	Prevent exploding gradients
Image size	256$\times$256	EfficientNet-B0 standard input
Max frames	16	Balance between coverage and memory

Metric	**N5K: Depth 4ch**	**N5K: Volume**	**Ours: Fusion**
Calorie MAE	47.6	41.3	49.9
Mass MAE	40.7	29.4	33.0
Fat MAE	2.27	3.0	4.6
Carb MAE	4.6	4.5	10.1
Protein MAE	3.7	5.2	7.2

Table 8: Baseline: Nutrition5k benchmark models vs. our two-stream RGB+Depth fusion model.

Subset	Train	Test
All three modalities	2,638	474
Side frames only	1,292	202
Overhead only	117	33
Total	**4,047**	**709**

Table 9: Dataset composition with missing modalities

Metric	Exp 3	Exp 4
Training Dishes	2,638	4,047 (+53%)
Best Val Loss	131.9	218.3
Final Val Loss	154.6	255.0
Final Train Loss	75.3	92.4

Table 10: More data did not improve performance

A.4 Cross-Swin-CLS Results

Metric	**Value**	**Notes**
Final Training Loss	0.8752	After 41 epochs
Final Test Loss	1.1314	—
Train–Test Gap	0.2562	Good (small gap)
Loss Reduction (%)	75.71	From initial loss

Table 11: Cross-Swin-CLS performance summary.

A.5 ConvNeXt Results

Metric	**Run 1**	**Run 2**	**Notes**
Final Training Loss	0.4381	0.9066	Run 2 stopped at epoch 20
Final Test Loss	1.1429	1.4276	Similar generalization range
Train–Test Gap	0.7048	0.5210	Run 1 overfits heavily
Gap Increase (%)	160.88	57.48	Relative to train loss
Loss Reduction (%)	87.75	74.65	Faster early convergence in Run 2

Table 12: ConvNeXt performance comparison across two learning-rate schedules.