NutriFusionNet: A Graph-Attention Multimodal Approach for Nutritional Analysis

Yancheng Liu
Department of Computer Science,
Brown University
Providence, RI, USA
yancheng_liu@brown.edu

David Ning
Department of Computer Science,
Brown University
Providence, RI, USA
zhenglin_ning@brown.edu

Yunqi Li
Department of Computer Science,
Brown University
Providence, RI, USA
yunqi_li@brown.edu

1 INTRODUCTION

Nutritional analysis plays a crucial role in personal health and fitness management. Individuals ranging from fitness enthusiasts to health-conscious individuals often struggle to obtain accurate nutritional information, especially when dining out or preparing homemade meals. Traditional methods of nutritional tracking rely on manual input, pre-printed labels, or approximate estimates, which can be time-consuming and imprecise.

The advent of computer vision and deep learning technologies presents a promising solution to this challenge. By leveraging advanced machine learning techniques, it becomes possible to estimate nutritional facts from simple food images. This project introduces NutriFusionNet, an innovative approach to nutritional analysis that goes beyond traditional image recognition methods.

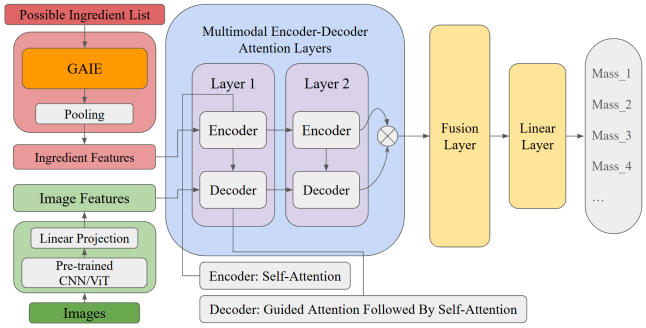

Figure 1: Structure Of NutriFusionNet

The primary motivation for this research stems from the practical needs of fitness communities and health-conscious individuals who require precise nutritional information. Existing solutions often fall short in providing accurate and detailed nutritional insights, particularly for dishes without readily available nutritional labels. **NutriFusionNet** addresses these limitations by developing a sophisticated deep learning model capable of estimating ingredient masses and nutritional content from overhead dish images.

Key contributions of this research include:

(1) A novel **visual question answering** framework adapted for nutritional analysis.
(2) Development of a Graph-Attentive Ingredient Embedding (**GAIE**) to capture meaningful ingredient relationships.
(3) A multimodal approach that combines image feature extraction with ingredient-level analysis.
(4) Significant improvements in nutritional fact prediction compared to existing methods.

By rethinking the nutritional estimation problem as an ingredient mass prediction task, NutriFusionNet provides more ingredient-level insights into food composition. The model's ability to identify and quantify ingredients offers a more nuanced approach to nutritional understanding.

1.1 Code Implementations

The complete code implementations are available at NutriFusionNet GitHub Repository.

2 RELATED WORK

This project draws inspiration from Google's Nutrition5k dataset [15] and its associated model for estimating nutritional facts, such as mass, protein, and calories of food items, using images. The original model leverages a convolutional neural network (CNN) architecture, specifically the Inception V2 Body, to extract visual features from food images. These features are flattened and passed into fully connected layers, with multiple output heads designed for specific tasks. Many related works adopt similar structures, combining a pre-trained vision backbone with multi-layer perceptrons to directly predict nutritional facts or classify food categories. [12][6][10]

Our approach introduces a novel angle by predicting ingredient-level masses instead of directly estimating nutritional values. This reformulation aligns with visual question answering frameworks and allows the model to perform a hybrid task that fuses regression and classification. This design not only improves performance but also provides granular insights into the composition of food items beyond their overall nutritional metrics. Our method emphasizes the relationships between ingredients, bridging gaps in existing models and enhancing interpretability.

3 DATASET

The dataset used in this study is Nutrition5k, provided by Google [15]. It contains approximately 5,000 images and videos of dishes, including RGB-D images captured using a RealSense D435 camera. Each dish is annotated with nutritional facts and ingredient-level details, including ingredient masses. The dataset also includes USDA composition data for all ingredients involved, specifying nutritional values such as calories, fat, carbohydrates, and protein per gram.

The dataset is organized incrementally, where different portions of the same dish are treated as unique entries. Predefined train-test splits ensure that similar dishes do not overlap between training and testing data. The dataset includes two CSV files listing dish IDs for training and testing.

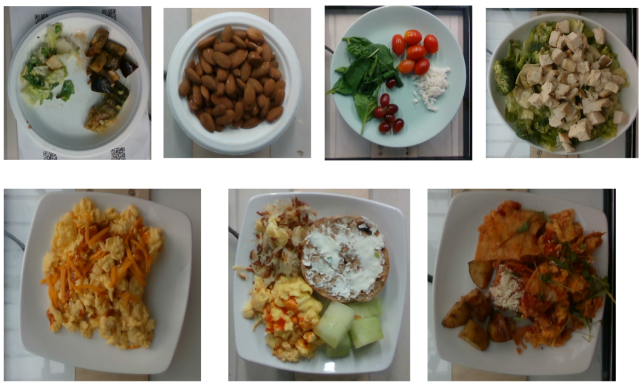

Figure 2: Examples Of Overhead Dish Images

ID	Calories	Mass	Fat	Carb	Protein	Egg	Quinoa	...
1	98.391266	74	2.671786	15.661942	2.793359	22	50.485435	...
2	37.720001	92	0.184	9.2	0.828	0	0	...
3	100.300003	243	0.58	24.836	1.655	0	0	...
...	...	...	...	...	...	...	...	...

Figure 3: Sample Nutrition And Ingredient Label Entries from Nutrition5k

3.1 Issues with the Dataset

Upon inspection, we identified several issues in the dataset. First, only about **3,200** dishes have valid overhead images, which is critical for our project as we aim to enable users to predict nutritional information directly from overhead images. Consequently, we excluded dishes without overhead images or those relying on RGB-D data.

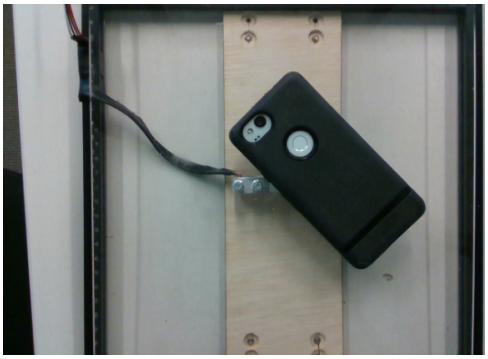

Figure 4: Overhead Image of Dish ID 1559172356

Second, some overhead images were invalid, such as Dish ID 1559172356, which features a cellphone rather than food. Additionally, there are repeated images (e.g., 1562611580 and 1562611522 both have identical pizzas), and certain labels contain errors, such as total mass being zero while ingredient masses are non-zero.

Finally, the dataset's incremental nature may limit generalization as shown in Figure 5. However, this structure is advantageous for identifying individual ingredients rather than direct prediction of dish-level nutritional information.

Figure 5: The Incremental Nature Of The Nutrition5k Dataset

3.2 Data Reconstruction and Preprocessing

To prepare the dataset, we first filtered the data to retain only valid entries with both overhead images and accurate labels. Invalid entries, such as the image featuring a cellphone, were excluded. The reconstructed dataset consists of all valid overhead images. For each dish, we retained only the nutritional information and ingredient masses. The model was trained using the ingredient masses, while nutritional information was reserved for evaluating model performance.

The reconstructed dataset includes **199** possible ingredients, with masses ranging from 0 to over 1,000 grams. However, the distribution is heavily right-skewed, with most ingredients averaging around 3 grams. To address this skewness and stabilize the loss scale, we applied a **log transformation** to the ingredient masses. The dataset was then split into training, validation, and testing sets, containing **2,755**, **203**, and **304** samples, respectively.

For preprocessing, all images were resized to 224×224 RGB format to align with the pre-trained image extractors described in the next section. We augmented the training set with random rotations, resizing, color jittering, flipping, and affine transformations to enhance generalization. [13] Validation and test sets underwent normalization only. Each transformed image is denoted as $\tilde{I}$. Labels were transformed into arrays of size $(B, 199)$, where B is the batch size, and 199 corresponds to the possible ingredients. The i-th element in the label vector represents the mass of the i-th ingredient. As a result, each dataloader outputs inputs of size $(B, 3, 224, 224)$ and corresponding labels of size $(B, 199)$.

4 METHODOLOGY

Visual Question Answering (VQA) [1] is a framework that combines visual and textual inputs to answer questions about images, enabling models to perform complex reasoning across modalities. It integrates image features with question semantics to generate precise, context-aware responses. We use pre-trained models like ResNet [5]/Vision Transformer [4] to extract detailed image features, and BERT [3] to capture semantic nuances of ingredient names. This combination ensures robust image-text representation for accurate reasoning.

We adopted the multimodal encoder-decoder attention network proposed by Chen et al. [2] within a VQA framework to focus the model on relevant features. The encoder captures fine-grained

attention in the text, while the decoder applies question-guided attention to the image, enabling dense interactions between textual components and their corresponding visual representations. This approach ensures precise alignment between visual regions and the queried components, facilitating accurate predictions.

In our case, it is intuitive to frame the problem as a question like, "How much of [ingredient] is in this dish?" within the VQA framework, where the answer is the mass of the specified ingredient. Thus, we treat the ingredient list as the "question" and predict the mass of each ingredient as the "answer." This approach guides the model to focus on specific regions in the image according to ingredients which can improve model's prediction accuracy. Additionally, processing each ingredient-question independently enables the framework to handle diverse and complex dishes effectively.

4.1 Naive NutriFusionNet

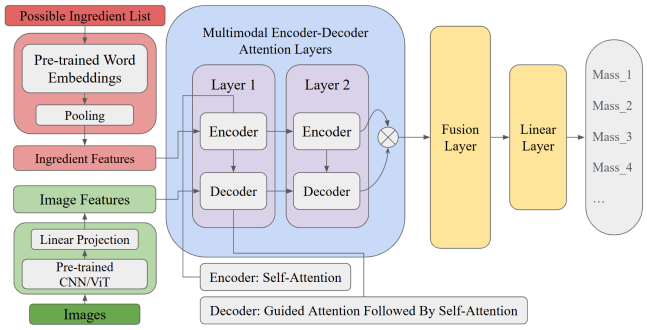

Figure 6: Structurer Of Naive NutriFusionNet (NNFN)

The Naive NutriFusionNet (**NNFN**) integrates pre-trained image and word embeddings with multimodal encoder-decoder attention layers to align visual and textual features effectively. The overall structure of NNFN is depicted in Figure 6.

4.1.1 Structural Change From The Original Paper. In Chen's paper [2], LSTM layers process input question embeddings by using sequential and contextual relationships. However, in our case, ingredient embeddings lack such correlations, making LSTMs unnecessary. Their use introduces additional computational overhead and risks losing global relationships without significant performance gains.

To address this, we replace LSTMs with global pooling methods, such as mean [8], max, or attention pooling [17], to aggregate ingredient embeddings into single features for subsequent encoders. This approach enables the model to capture global relationships through linear layers while simplifying computations.

Testing shows that pooling methods achieve comparable performance to LSTMs but significantly improve training and inference speeds. For example, on 3,000 inputs, the average inference time decreased by 5%, making pooling a more efficient and effective choice for handling ingredient embeddings.

4.1.2 Data Flow in Naive NutriFusionNet. The Naive NutriFusionNet processes both ingredient and image data through parallel embedding extraction pipelines, followed by multimodal attention layers for feature alignment and regression. Let L denote the **list of possible ingredients** (199 in our case), and I represent the **original dish image**. The ingredient embeddings are computed using a pre-trained Bert model:

$$\hat{L} = \text{Bert}(L),$$

where $\text{Bert}(\cdot)$ maps each ingredient to a fixed-length embedding. To aggregate these embeddings into a single feature vector E, a pooling function $P(\cdot)$ is applied. For instance, using average pooling:

$$E = P(\text{Bert}(L)) = \frac{1}{N}\sum_{i=1}^{N}\text{Bert}(\text{Ingr}_i), \tag{1}$$

where N is the number of ingredients in L. This pooled vector provides a global representation of all ingredients.

The image I is preprocessed (Section 3.2) and passed through a pretrained ResNet for feature extraction:

$$V = FF(\text{ResNet}(\tilde{I})),$$

where $\tilde{I}$ is the preprocessed image, $\text{ResNet}(\cdot)$ extracts visual features, and $FF(\cdot)$ projects the output to a lower-dimensional feature space via a fully connected layer.

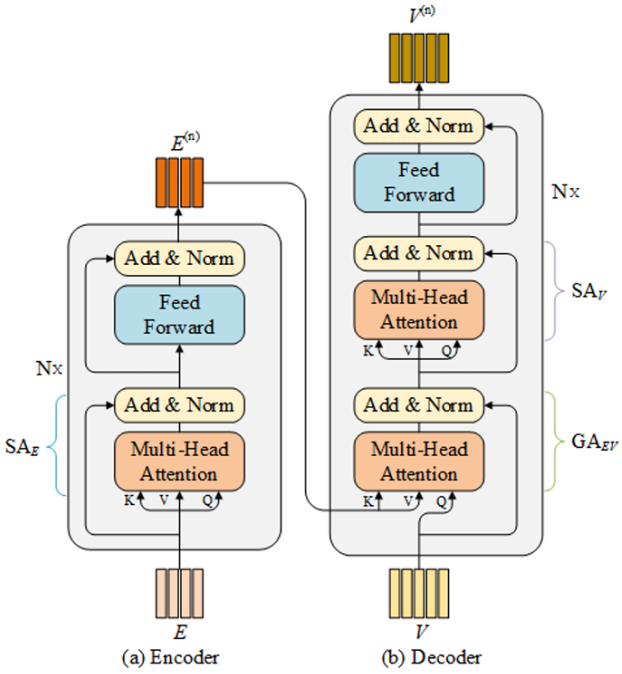

Figure 7: Attention Mechanism For Multimodal Encoder-Decoder Attention Layer [2]

For the attention layer, we followed the design of Chen's with some small tuning based on our dataset. The original design can be seen in Figure 7. The multimodal encoder-decoder processes ingredient features E and image features V using attention mechanisms. For the encoder, ingredient embeddings E are passed as the query,

key, and value to a self-attention layer:

$$F_E = \text{Attn}(Q, K, V) = \text{Softmax}\left(\frac{Q \cdot K^\top}{\sqrt{d_k}}\right) \cdot V, \tag{2}$$

where $Q = K = V = E$. A residual connection and layer normalization stabilize the output:

$$\hat{F}_E = \text{LN}(E + F_E).$$

The output is passed through a feed-forward layer with ReLU activation, followed by another residual connection and layer normalization:

$$\begin{aligned}\overline{E} &= \text{LN}(\text{ReLU}(FF(\hat{F}_E)) + \hat{F}_E) \\ &= \text{LN}(\max(0, \hat{F}_E \cdot W_1 + b_1) + \hat{F}_E),\end{aligned}$$

where W_1 and b_1 are the weights and bias of the feed-forward layer.

In the decoder, the guided attention mechanism aligns image features V with ingredient features $\overline{E}$. The guided attention is computed as:

$$\overline{V} = \text{LN}\left(V + \text{Attn}(Q_V, K_{\overline{E}}, V_{\overline{E}})\right),$$

where $Q_V = V$, $K_{\overline{E}} = V_{\overline{E}} = \overline{E}$. Self-attention is applied to $\overline{V}$ to capture correlations between image regions:

$$\begin{aligned}\overline{E} &= \text{LN}(\text{ReLU}(FF(\overline{V})) + \overline{V}) \\ &= \text{LN}(\max(0, \overline{V} \cdot W_2 + b_2) + \overline{V}),\end{aligned}$$

where W_2 and b_2 are the parameters of the decoder's feed-forward layer. Multiple layers of attention can be stacked by using the output of the current layer as the input for the next layer. This hierarchical structure allows the model to iteratively refine the alignment between ingredient and image features. The detailed procedure follows the approach outlined in Chen's paper. [2]

The final ingredient and image features are concatenated:

$$T = \overline{E} \otimes \tilde{V}.$$

This fusion avoids overfitting by combining multimodal features in a compact representation. The fused features T are passed through a fully connected layer for regression and final prediction:

$$M = FF(T),$$

where M represents the predicted ingredient masses.

The data flow of **NNFN** combines **global pooling** for efficient ingredient feature extraction and attention mechanisms for multimodal alignment, avoiding the limitations of LSTMs.

4.2 Graph-Attentive Ingredient Embedding

4.2.1 Problems With Pre-Trained Embedding. During our experiments, we observed significant limitations in using pre-trained Bert Word Embeddings for ingredient similarity. Specifically, in the top 100 pairs of ingredients with the highest similarity scores, more than half of them were found to never co-occur in dishes.

For example, the similarity score between the embeddings of "bacon" and "cereal" is 0.96, while the similarity score between the ones of "bacon" and "Oliver Oil" is only 0.62. However, in our dataset, bacon and cereal never co-occurred in dishes, whereas bacon and olive oil co-occurred 48 times. This discrepancy suggests that Bert embeddings are not well-suited for capturing co-occurrence relationships specific to ingredients.

The root cause of this issue likely stems from the broad and general-purpose training data used by Bert, which categorizes all food items under a shared semantic "food" class without accounting for specific co-occurrence patterns. Consequently, Bert embeddings tend to reflect general semantic relationships rather than the domain-specific patterns necessary for modeling ingredients effectively.

This misalignment directly impacts the attention mechanism in both the encoder and decoder of our model. Assume the embeddings for bacon, cereal, and olive oil are represented as $\mathbf{h}_{\text{bacon}}$, $\mathbf{h}_{\text{cereal}}$, and $\mathbf{h}_{\text{olive oil}}$, respectively. Pooling functions, such as average pooling or attention pooling, do not alter the underlying relationships between embeddings; they merely reduce dimensionality and smooth variations. As a result, the high similarity score between Bert(Bacon) and Bert(Cereal) will continue to dominate, even though these ingredients never co-occur.

For example, in the encoder, the attention weight assigned to "cereal" relative to "bacon" is:

$$\alpha_{\text{bacon, cereal}} = \frac{\exp\left(\frac{\mathbf{h}_{\text{bacon}} \cdot \mathbf{h}_{\text{cereal}}}{\sqrt{d_k}}\right)}{\sum_j \exp\left(\frac{\mathbf{h}_{\text{bacon}} \cdot \mathbf{j}}{\sqrt{d_k}}\right)},$$

based on Equation 2. While the attention weight for "olive oil" relative to "bacon" is:

$$\alpha_{\text{bacon, olive oil}} = \frac{\exp\left(\frac{\mathbf{h}_{\text{bacon}} \cdot \mathbf{h}_{\text{oliver oil}}}{\sqrt{d_k}}\right)}{\sum_j \exp\left(\frac{\mathbf{h}_{\text{bacon}} \cdot \mathbf{j}}{\sqrt{d_k}}\right)}.$$

Given Bert embeddings, where $\mathbf{h}_{\text{bacon}} \cdot \mathbf{h}_{\text{cereal}} = 0.96$ and $\mathbf{h}_{\text{bacon}} \cdot \mathbf{h}_{\text{oliver oil}} = 0.62$, the attention weight $\alpha_{\text{bacon, cereal}}$ becomes disproportionately higher than $\alpha_{\text{bacon, olive oil}}$. This misalignment skews the encoder's representation, emphasizing irrelevant relationships (e.g., "bacon" and "cereal") over meaningful ones (e.g., "bacon" and "olive oil"). Consequently, the decoder inherits this flawed representation, trying to align the irrelevant image features with the misleading ingredients. This will further decrease the relevance, which results in inaccurate predictions.

4.2.2 Proposed Solution: Custom Ingredient Embeddings. To address this issue, we propose replacing Bert embeddings with custom embeddings trained specifically to capture co-occurrence patterns within our dataset. We construct a co-occurrence graph where ingredients are represented as nodes, and edges are weighted by the frequency of co-occurrence between ingredients. For example, an edge weight w_{ij} represents the co-occurrence count of ingredient i with ingredient j.

We followed the approach of Rong et al.'s [11] and Mikolov et al.'s [9] to construct **CoOccurrenceGNN**, a graph neural network designed to learn embeddings from co-occurrence graphs. [18] The model uses two stacked Graph Attention (GAT) [16] layers to dynamically weigh the importance of neighboring nodes and incorporate edge attributes during message passing. Each layer is followed by batch normalization for stability, dropout for regularization, and optional residual connections when input and output dimensions match. The overall structure can be seen in Figure 8.

The co-occurrence graph $G = (V, E)$ consists of nodes V, edges E, node features $X \in \mathbb{R}^{N \times d}$, and edge attributes $A \in \mathbb{R}^{|E| \times d_e}$. Each

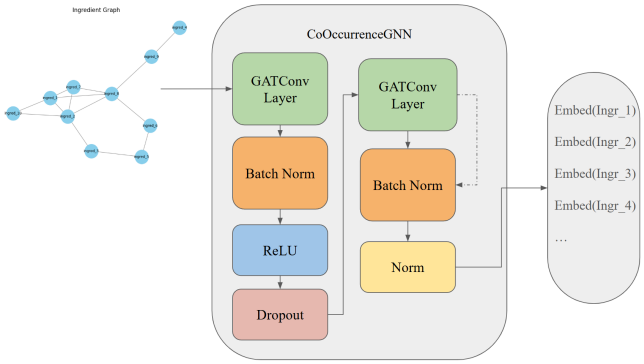

Figure 8: Structure Of CoOccurrenceGNN

GAT layer updates node embeddings by attending over neighbors, incorporating edge attributes. The attention weight [16] is defined as:

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(a^\top [W_h h_i \| W_h h_j \| W_a a_{ij}]\right)\right)}{\sum_{k \in \mathcal{N}(i)} \exp\left(\text{LeakyReLU}\left(a^\top [W_h h_i \| W_h h_k \| W_a a_{ik}]\right)\right)},$$

where h_i is the input feature of node i, a_{ij} is the edge attribute between nodes i and j, and W_h, W_a are learnable weight matrices. The GAT attention illustration can be seen in Figure 9. The output for each node is computed as:

$$h_i' = \text{ReLU}\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij} (W_h h_j + W_a a_{ij})\right).$$

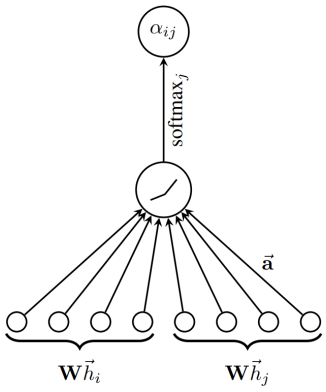

Figure 9: Attention Mechanism For GAT [16]

The model processes the graph in two GAT layers. In the first layer, the node features are updated as:

$$H_1 = \text{Dropout}\left(\text{ReLU}\left(\text{BatchNorm}\left(\text{GATConv}(X, \text{edge\_index}, A)\right)\right)\right).$$

In the second layer, the output embeddings are refined:

$$H_2 = \text{BatchNorm}\left(\text{GATConv}(H_1, \text{edge\_index}, A)\right).$$

If the input and output dimensions match, a residual connection is added:

$$H_2 = H_2 + X.$$

The final embeddings are L_2-normalized to ensure scale invariance:

$$H_{\text{final}} = \frac{H_2}{\|H_2\|}.$$

The model optimizes a loss function that ensures frequently co-occurring ingredients, such as "bacon" and "olive oil," have high similarity scores, while rarely or never co-occurring pairs, such as "bacon" and "cereal," have low similarity scores. The loss function is defined as:

$$\mathcal{L} = \frac{1}{|E|} \sum_{(i,j) \in E} \left(w_{ij} - \mathbf{h}_i^\top \mathbf{h}_j\right)^2,$$

where w_{ij} is the normalized co-occurrence weight, and $\mathbf{h}_i$, $\mathbf{h}_j$ are the learned embeddings. By training on the co-occurrence graph, the embeddings directly encode the relationships specific to the dataset.

The model is trained using a supervised approach. During each epoch, the graph is split into training and validation sets. Node embeddings are computed via the forward pass, and the dot product of connected node embeddings predicts edge attributes. Loss between predicted and true edge attributes is minimized using the Adam optimizer [7]. Validation loss is computed at the end of each epoch without parameter updates, allowing monitoring of overfitting and generalization.

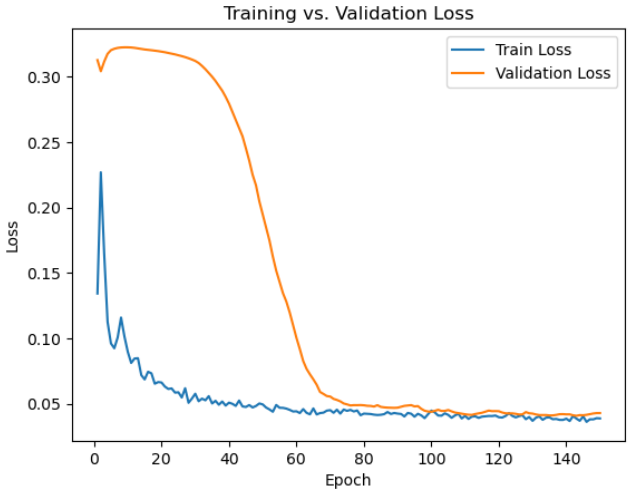

Figure 10: The Losses Of CoOccurrenceGNN

Training is conducted over 150 epochs with periodic logging of training and validation losses. The convergence of these losses is visualized in Figure 10. The final validate loss is 0.0428 and the test loss is 0.0546. We denote the finalized ingredient embeddings through CoOccurrenceGNN as **GAIE** (Graph-Attentive Ingredient Embedding).

4.2.3 Evaluation For GAIE. The evaluation assesses the performance of CoOccurrenceGNN and GAIE for link prediction by predicting the existence of edges in the graph. The trained model generates node embeddings based on features and edge attributes, and similarity scores for edges are computed using the dot product

of connected node embeddings. Positive edges (present in the test graph) are compared against dynamically sampled negative edges (non-connected node pairs) to evaluate the model.

Similarity scores and labels (1 for positive, 0 for negative edges) are used to compute two metrics: **ROC AUC** (Receiver Operating Characteristic Area Under the Curve) and **PR AUC** (Precision-Recall Area Under the Curve). ROC AUC measures the model's ability to distinguish between connected and non-connected edges, while PR AUC evaluates its performance in imbalanced scenarios. A precision-recall curve visualizes the model's precision across different recall values as shown in Figure 11.

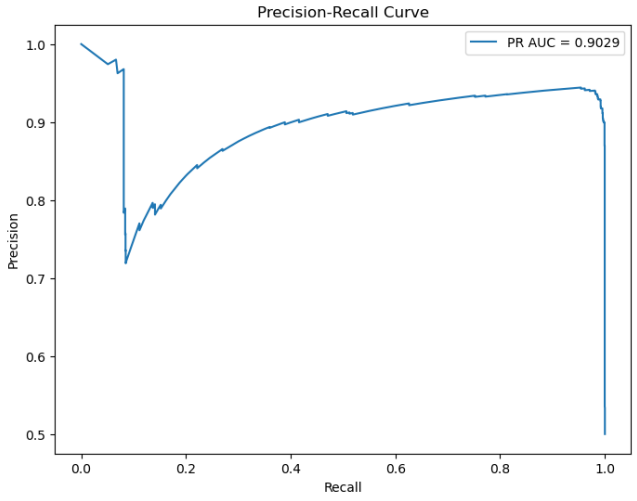

Figure 11: Precision-Recall Curve For GAIE

Results demonstrate the model's effectiveness, with a ROC AUC of 0.95 indicating excellent discrimination between edges and a PR AUC of 0.90 highlighting robustness in imbalanced edge distributions. The precision-recall curve confirms high precision over a wide range of recalls, ensuring reliable edge predictions. These results validate CoOccurrenceGNN's suitability for link prediction tasks using node and edge information.

To qualitatively observe the difference, the similarity between GAIE for "bacon" and "olive oil" is 0.60, while the similarity between "bacon" and "cereal" is reduced to −0.1339. This demonstrates that GAIE effectively captures the co-occurrence information specific to the dataset, distinguishing frequently co-occurring pairs from unrelated ones. Furthermore, among the top 100 pairs with the highest similarity scores in GAIE embeddings, 99 of them co-occur in at least 10 dishes, further validating the effectiveness of our approach in aligning the embeddings with the actual co-occurrence patterns in the dataset.

Figure 12 demonstrates the feasibility of GAIE by visualizing ingredient embeddings in a 2D space. The clear separation and clustering of points suggest that GAIE effectively captures meaningful co-occurrence patterns from the dataset. Ingredients that frequently co-occur are positioned closer together, while those with weak or no relationships are farther apart. This indicates that GAIE embeddings accurately reflect co-occurrence-driven relationships, making them suitable for downstream tasks of ours.

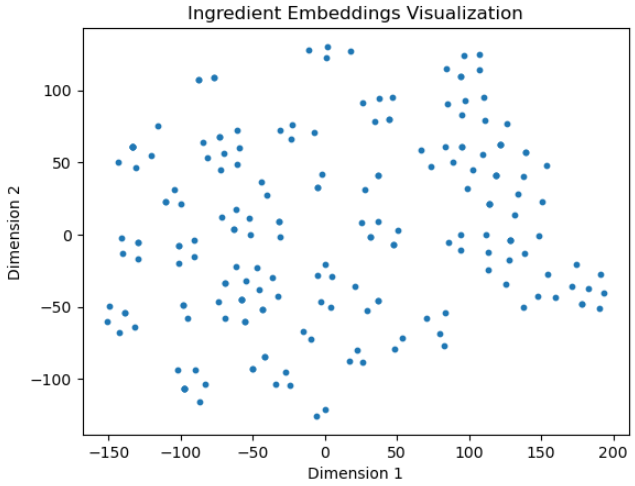

Figure 12: GAIE Visualization in a 2D Space

4.2.4 Impact on Attention Mechanism. **GAIE** address the limitations of Bert embeddings by aligning similarity scores with true co-occurrence patterns. For instance, the similarity $\mathbf{h}_{\text{bacon}} \cdot \mathbf{h}_{\text{olive oil}}$ is now significantly higher than $\mathbf{h}_{\text{bacon}} \cdot \mathbf{h}_{\text{cereal}}$. As a result, the attention weights $\alpha_{\text{bacon, olive oil}}$ and $\alpha_{\text{bacon, cereal}}$ will better reflect the true relationships between ingredients, leading to more accurate representations in both the encoder and decoder.

4.3 Refined NutriFusionNet

In the refined NutriFusionNet (**NFN**), we replace pre-trained embeddings like Bert with GAIE in Equation 1. Hence, we have that

$$E = P(\text{GAIE}(L))$$

as the input ingredient features to the encoder in the attention layer. The overall structure of NFN is illustrated in Figure 13.

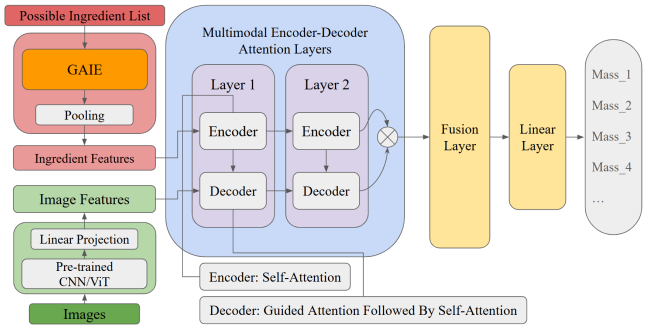

Figure 13: Structure of NFN

4.3.1 Loss Function Design. To address the imbalance in ingredient mass data (where only a few ingredients per dish have non-zero masses), we employ a custom loss function that combines a weighted Mean Squared Error (MSE) with L_2-regularization. This approach ensures that the model focuses on accurately predicting

significant ingredient masses while maintaining robustness and generalizability.

Let N represent the number of possible ingredients and B the batch size. The predicted masses are denoted as $\hat{\mathbf{y}} \in \mathbb{R}^{B\times N}$, the ground truth masses as $\mathbf{y} \in \mathbb{R}^{B\times N}$, and a binary mask $\mathbf{m} \in \{0,1\}^{B\times N}$, where $m_{ij} = 1$ indicates a non-zero ground truth mass, and $m_{ij} = 0$ otherwise.

The Mean Squared Error (MSE) for each ingredient is calculated as:

$$\text{MSE}_{ij} = (\hat{y}_{ij} - y_{ij})^2,$$

where $\hat{y}_{ij}$ and y_{ij} are the predicted and ground truth masses for the j-th ingredient in the i-th sample. A weighting factor w_{ij} is introduced to assign higher importance to non-zero masses:

$$w_{ij} = \alpha \cdot m_{ij} + (1-\alpha)\cdot(1-m_{ij}),$$

where α (e.g., $\alpha = 0.9$) controls the relative importance of non-zero masses versus zero masses. The weighted MSE is then defined as:

$$\mathcal{L}_{\text{Weighted MSE}} = \frac{1}{B\cdot N}\sum_{i=1}^{B}\sum_{j=1}^{N} w_{ij}\cdot \text{MSE}_{ij}.$$

To prevent overfitting and further regularize the model, we add an L_2-regularization term on the model parameters θ. The final loss function is formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{Weighted MSE}} + \lambda\|\theta\|_2^2,$$

where λ is a hyperparameter controlling the regularization strength.

4.3.2 Parameter Setting and Training. We conducted extensive parameter tuning for the model. For pooling layers, we compared LSTM, average pooling, max pooling, and attention pooling. Additionally, we tested 1-2 multimodal attention layers, finding that performance degraded with 3 layers, likely due to overfitting.

The final model uses a batch size of 16, 75 epochs, and a learning rate of 0.0001. Training utilizes the Adam optimizer with weighted MSE and $L2$-regularization ($\alpha = 0.9$ and $\lambda = 1e^{-3}$). Pretrained image feature extractors include ResNet and ViT, chosen for their strong performance in visual representation tasks. ResNet exhibited smoother convergence, while ViT showed fluctuations during training.

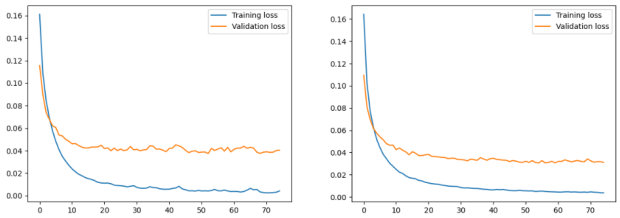

Figure 14: Left: The Losses Of NFN with ViT and Global Average Pooling; Right: The Losses Of NFN with Resnet and Global Average Pooling

We saved the model with the best validation loss and used early stopping with a patience of 25 epochs. A loss plot for two of the training sessions are shown in Figure 14, with additional plots available in the project repository.

5 RESULTS

For each input dish, the model predicts the mass of all possible ingredients. A threshold is applied to filter out ingredients with negligible predicted masses, enhancing prediction accuracy. The filtered masses are then used to compute the overall nutritional values of the dish by multiplying each ingredient's mass with its corresponding USDA composition data and summing the results. In this way, we can get the nutritional information of the dish.

5.1 Comparison To Existing Method

To evaluate our model, we followed the evaluation framework outlined in the Nutrition5k paper [15], which serves as a widely accepted standard. Performance is measured using Mean Square Error (MSE) for each nutritional category and the overall MSE.

Our best-performing model, NutriFusionNet (**NFN**), incorporates a **ViT** backbone, **global average pooling**, and 512-dimensional **GAIE**. It demonstrates significant improvements over existing methods.

Model	Overall MAE	Calories	Mass	Fat	Carbs	Protein	Classification Acc	Overall Improvement
Baseline	65.13	168.59	118.27	10.35	13.91	14.54	-	-
CNN+MLP	53.31	140.41	96.05	7.78	11.41	10.92	0.37	20.22%
ResNet + LSTM	43.90	112.30	82.11	6.19	9.74	9.14	0.49	33.97%
Google Direct Prediction	25.52	70.60	**40.40**	5.00	6.10	**5.50**	-	-
NutriFusionNet	**24.66**	**66.35**	41.59	**4.29**	**5.27**	5.81	**0.86**	**61.38%**

Figure 15: Comparison of Model Performance Metrics

From Figure 15, we can see that NFN achieves the lowest overall MAE of 24.66, representing a substantial 61.38% improvement over the baseline, which predicts average values for each task across the dataset.

Notably, our approach outperforms existing methods despite relying solely on overhead images, whereas the Nutrition5k paper [15] uses multiple image angles and RGB-D data. This highlights the effectiveness of our VQA framework with Multimodal Attention and Graph-Attentive Ingredient Embedding GAIE.

Additionally, the model demonstrates high classification accuracy of 0.86, reflecting its ability to not only produce precise numerical predictions but also accurately classify and understand dish compositions.

Other NutriFusionNet configurations with varying parameters and backbones also achieve comparable performance to the Nutrition5k model, further emphasizing the robustness and adaptability of the VQA framework in addressing this task.

5.2 Results With GAIE

Based on Section 4.2, **GAIE** should have impacts on the performance of the model. To assess this impact, we compare the performance of **NNFN** (using BERT embeddings) with **NFN** (using GAIE).

Table in Figure 16 shows the the average performance of the NutriFusionNet model using two embedding methods: BERT and Graph-Attentive Ingredient Embedding (GAIE, 512-dimensional embeddings). GAIE slightly outperforms BERT in classification

accuracy, suggesting better alignment of features with ground truth labels for ingredient classification.

Metric	NutriFusionNet (BERT)	NutriFusionNet (GAIE)	Difference
Regression Loss	0.0333	**0.0340**	+0.0007
Classification Accuracy	0.7607	**0.7618**	+0.0011
Precision	**0.1597**	0.1588	-0.0009
Recall	**0.9167**	0.9154	-0.0013
F1 Score	**0.2630**	0.2624	-0.0006
Average MAE	27.3169	**26.9485**	-0.3684
Calories MAE	74.6093	**73.2640**	-1.3453
Mass MAE	45.3893	**44.9532**	-0.4361
Fat MAE	4.1753	**4.1448**	-0.0305
Carbs MAE	**6.0773**	6.1538	+0.0765
Protein MAE	6.3330	**6.2268**	-0.1062
Overall Improvement	57.9710%	**58.3384%**	+0.3674%

Figure 16: Average Performance Comparison Between NutriFusionNet (BERT) and NutriFusionNet (GAIE)

Both models exhibit high recall values, indicating effective detection of relevant ingredients. However, precision is slightly lower, reflecting the potential inclusion of irrelevant ingredients in predictions.

NFN shows consistent improvements in Calories MAE and Mass MAE, signifying better predictions in these categories. NNFN has slightly better MAE for Carbs, suggesting nuanced differences in how the embeddings represent certain ingredients. NFN achieves a marginally higher overall improvement score, reflecting its better generalization across all metrics.

The comparison highlights that GAIE embeddings generally offer improved performance across key metrics, particularly in calories and mass predictions. These results underscore the importance of task-specific embeddings like GAIE for capturing ingredient relationships, which BERT may overlook due to its broad pretraining. The high recall for both methods suggests robustness in detecting ingredients, while further refinement may enhance precision.

To interpret the model's behavior, we use integrated gradients [14] for visualizing feature importance. Integrated gradients attribute the contribution of each input feature to the model's prediction by computing the path integral of gradients along a scaled input. This method provides a clearer understanding of which parts of the image influence the model's decisions.

The visualization in Figure 17 highlights that the NFN primarily focuses on the food items within the image, while the NNFN includes noise near the edges of the plate. Compared to NNFN, the attention layers in NFN allocate less focus on aligning image features with irrelevant ingredients. Instead, the model directs attention toward ingredients more likely to co-occur, improving the alignment between image and ingredient features. This reduces unnecessary connections between ingredients and their surroundings, enhancing the model's ability to identify relevant features effectively.

5.3 Results Discussion

The results demonstrate the effectiveness of NutriFusionNet (**NFN**), particularly with **Graph-Attentive Ingredient Embedding (GAIE)**.

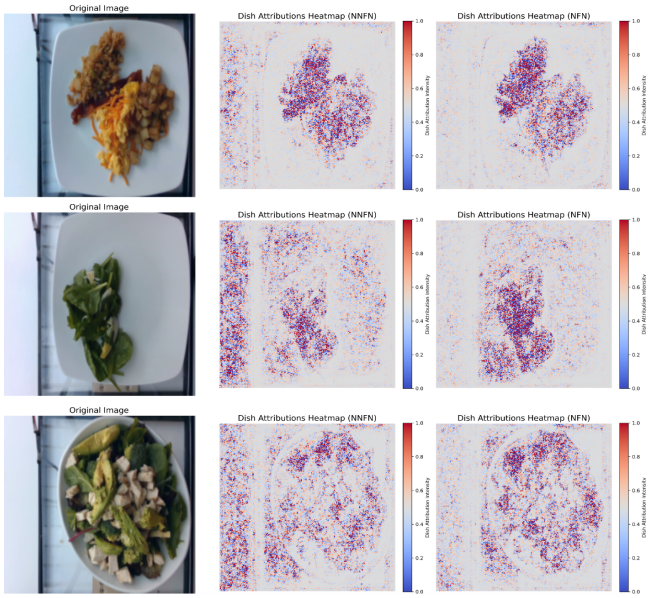

Figure 17: Visualization of Attention Mechanisms

NFN achieves the lowest Mean Absolute Error (MAE) of 24.66, representing a 61.38% improvement over the baseline. Unlike existing methods that use multiple image angles and RGB-D data, NFN relies solely on overhead images, highlighting the robustness of its VQA framework.

Compared to NNFN, NFN consistently outperforms in metrics such as classification accuracy, calories MAE, and mass MAE, showcasing better alignment with ground truth labels. Integrated gradients visualizations (Figure 17) reveal that NFN focuses on food items, reducing noise and irrelevant associations, unlike the Naive NutriFusionNet.

Overall, NFN's integration of GAIE, multimodal attention, and VQA enhances its ability to predict nutritional values accurately while maintaining model interpretability. Further refinements could address minor gaps in metrics like precision and carbohydrates MAE.

6 CONCLUSION

This study introduces NutriFusionNet (**NFN**), a novel framework for estimating nutritional information from overhead food images. By integrating a Visual Question Answering (VQA) framework, Graph-Attentive Ingredient Embedding (**GAIE**), and multimodal encoder-decoder attention mechanisms, NFN achieves state-of-the-art performance in predicting ingredient masses and overall nutritional values.

Our results demonstrate that NFN significantly improves over existing models, achieving the lowest Mean Absolute Error (MAE) of 24.66, a 61.38% improvement over the baseline. NFN outperforms prior approaches, such as those in the Nutrition5k paper, which require multiple image angles and RGB-D data. Using only overhead images, NFN effectively aligns visual and ingredient features, enabling accurate predictions while maintaining interpretability through integrated gradients and attention visualizations.

The comparison of embedding methods shows the superiority of GAIE over BERT in this task, highlighting the importance of task-specific embeddings for ingredient relationship modeling. GAIE enhances the model's ability to focus on co-occurring ingredients, improving predictions for calories, mass, and other critical nutritional categories while reducing irrelevant connections.

To conclude, this work underscores the potential of combining multimodal attention and domain-specific embeddings to address complex nutritional analysis tasks. NutriFusionNet lays a strong foundation for scalable, interpretable, and efficient dietary tools, paving the way for advancements in computational nutrition and personalized health applications.

6.1 Ethics

The Nutrition5k dataset was collected using an automated system that scans dishes to retrieve overhead images and video, minimizing the need for excessive human labor in the data collection process. However, there are some potential concerns about the representativeness of the dataset. The dataset is not globally diverse, it primarily includes dishes commonly found in the United States. For instance, it lacks representation of cuisines from Asia and Africa, so it might limits its utility for users from different cultural background. Additionally, the USDA Food and Nutrient Database, is also U.S. focused, potentially includes a regional bias for the nutrition prediction.

The primary stakeholders for this project are individuals who intend to use this tool to manage their daily diets. These users will rely on the tool to obtain a measure of the nutritional facts about their meals. Potential consequences of mistakes made by the algorithm include user consuming more calories then their intended target. Such mistake could cause dietary adjustments such as running for an extra kilometer, the stakes are relatively low. Because the MAE for each nutritional facts in our model are not excessively high, there shouldn't be any health concerns.

6.2 Division of labor

Yancheng Liu

- **Dataset Reconstruction:** Inspected the dataset, identified issues, selected valid samples, and reconstructed the dataset programmatically. (Section 3) Implementation is documented in the Data Reconstruction Notebook.
- **Preprocessing:** Designed image transformations and label normalization procedures. (Section 3.1) Codes are available in the Preprocess Script and Data Preparation Notebook.
- **Baseline Models:** Implemented direct prediction models, including paper methods and backbone LSTMs. (Section 2) Earlier versions can be reviewed in the GitHub repository. These methods are just for testing, not used later.
- **Mass Prediction Algorithm:** Developed the algorithm to predict ingredient masses and compute nutritional values using the USDA dataset. (Section 1 and Section 5)
- **Model Development:** Constructed baseline (CNN + MLP), Backbone + LSTM, and Naive NutriFusionNet (pre-trained model - Resnet + Bert with VQA). (Section 4 and Section 5) Codes are in the Models Script.
- **Loss Function Design:** Designed Weighted MSE with L2 regularization and tested alternatives (e.g., combined loss). Turns out only weighted MSE is stable for training. (Section 4.3.1) Implementation is available in the Training Script.
- **Training and Evaluation:** Wrote modular training, evaluation, and inference scripts. Trained models using SLURM (~150 runs) and generated loss plots and results. Scripts are located in the Training Scripts Directory.
- **Embedding Issues:** Identified issues with BERT embeddings (e.g., top 100 similarity pairs). (Section 4.2.1) Analysis is in the BERT Failure Notebook.
- **GAIE and GNN:** Designed the co-occurrence matrix and trained a Graph Neural Network (GNN) with attention layers. (Section 4.2.2) Evaluated GNN performance on top 100 pairs. (Section 4.2.3) Codes are in Co-Occurrence Data Creation and Training CoOccurrenceGNN.
- **Model Updates:** Updated NutriFusionNet with GAIE and evaluated structures with varying LSTM and attention layers. (Section 4.3) Developed pooling-based feature extractors to reduce computational overhead (Pooling-Based Extractor) (Section 4.1.1).
- **Model Interpretability:** Used integrated gradients and Grad-CAM to visualize feature importance, comparing BERT vs. GNN and ResNet vs. ViT. (Section 5.2) Visualizations are in the Integrated Gradients Notebook and Grad-CAM Notebook.
- **Downstream Analysis:** Conducted parameter tuning, threshold selection, and backbone testing (e.g., ConvNeXt, EfficientNet, InceptionV3) with comparable results. (Section 4.3 and Section 5)
- **User Interface:** Developed a user interface for faster inference and improved usability for presentations. Codes are in User Interface.
- **Project Check-In:** Authored the project check-in document.
- **Presentation Contributions:** Prepared slides for the introduction, dataset overview, model structure, issues with BERT embeddings, and solutions (VQA + GNN embeddings), and introduced the final structure.
- **Report Contributions:** Wrote the methodology section and revised other parts of the report.

David Ning

- Teacher model construction and testing with llama + GPT.
- Write scripts to auto-generate outputs from the teacher model.
- Prompt-engineering for the teacher model.
- Directly apply GNN between dishes and predict mass.
- GNN attention layer output visualization.
- GNN to create ingredient embedding.
- Slides and report.

Yunqi Li

- Utilized SAM to segment and mask dish images, retaining only the ingredients in each dish.
- Constructed the Llama 3.2 teacher model.
- Optimized teacher model prompts to generate accurate nutritional data and recommendations.

- Compiled a comprehensive summary of the project workflow and progress.
- Slides and report.

6.3 Reflection

Overall, the project evolved significantly from the base goal. Initially, our goal is to use distillation to guide a smaller model. However, during the development, we found that there is no model, such as GPT and Llama, that can accurately predict the mass. The result generated are similar or worse than the student model. As a result, we decided to change our approach, we decided to utilize the VQA modal with multimodal attention mechanism. This new approach was more complex than the proposed base goal but delivered a better performance.

Generally speaking, our model performed as expected, as we achieved a better / similar result compared to the original paper. Before, we used direct prediction to estimate all the nutritional facts for a dish, but we decided to focus on mass prediction first, since the model makes the highest mistakes in predicting mass. We achieved a better result with this new approach. Subsequently, we introduced the co-occurence graph with GNN to create ingredient embeddings, which were utilized in the multimodal encoder-decoder layer, we used the GNN's ability to model relationships between neighboring nodes, ingredients that are likely to be appeared together, to enhance the model's performance.

For Yancheng, one of the biggest takeaway from this project is that complexity does not always guarantee better results. For a smaller dataset like Nutrition5k, complex models are prone to overfitting. Techniques like data augmentation and simpler, well-designed models often yield better performance. This project deepened my understanding of multimodal frameworks, attention mechanisms, and graph neural networks, providing valuable insights into balancing model complexity with practical effectiveness.

The biggest takeaway for David was learning the process of applying deep learning to a task from the very beginning, all the way from the data preprocess and to build the layers. This project provided an opportunity to apply the knowledge learned in the class to a real-world scenario task.

7 FUTURE WORKS

Expanding the dataset to include a more diverse range of cuisines from various cultural backgrounds could significantly enhance the model's ability to generalize. Currently, the dataset is primarily focused on dishes common in the United States, which limits its applicability to a global audience. To address this, additional images and ingredient information could be collected using Python-based web scraping tools. This would allow the inclusion of underrepresented cuisines, improving the model's robustness and relevance for diverse dietary contexts.

Incorporating 3D representations of dishes could also improve the model's performance. By utilizing the device specifications used to capture the images, it would be possible to reconstruct 3D models of dishes. This would provide additional insights into the volume of food items, aiding in more accurate mass estimation. Such an enhancement could significantly boost the accuracy of downstream nutritional predictions.

Introducing multi-task learning into the framework could further improve performance. Adding a complementary task, such as ingredient presence detection, would allow the model to leverage shared representations, enhancing its ability to understand ingredient-specific features. This approach would not only refine mass prediction but also provide a more holistic analysis of the dish composition.

Finally, addressing biases in the pre-trained BERT embeddings could significantly enhance the model's output quality. The current embeddings may introduce noise or irrelevant relationships due to their general-purpose pretraining. By de-biasing and fine-tuning these embeddings for the specific task, the model could achieve more accurate representations of ingredient relationships, resulting in improved performance and reduced errors.

REFERENCES

[1] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual Question Answering. *CoRR* abs/1505.00468 (2015). arXiv:1505.00468 http://arxiv.org/abs/1505.00468

[2] Chongqing Chen, Dezhi Han, and Jun Wang. 2020. Multimodal Encoder-Decoder Attention Networks for Visual Question Answering. *IEEE Access* 8 (2020), 35662–35671. https://doi.org/10.1109/ACCESS.2020.2975093

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL] https://arxiv.org/abs/1810.04805

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv:2010.11929 [cs.CV] https://arxiv.org/abs/2010.11929

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.03385 [cs.CV] https://arxiv.org/abs/1512.03385

[6] V Balaji Kasyap and N. Jayapandian. 2021. Food Calorie Estimation using Convolutional Neural Network. In *2021 3rd International Conference on Signal Processing and Communication (ICPSC)*. 666–670. https://doi.org/10.1109/ICSPC51351.2021.9451812

[7] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG] https://arxiv.org/abs/1412.6980

[8] Min Lin, Qiang Chen, and Shuicheng Yan. 2014. Network In Network. arXiv:1312.4400 [cs.NE] https://arxiv.org/abs/1312.4400

[9] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781 [cs.CL] https://arxiv.org/abs/1301.3781

[10] G. Ramkumar, Venkatramanan C B, V. Manonmani, S. Palanivel, Sakthisaravanan. N, and Ata Kishore Kumar. 2023. A Real-time Food Image Recognition System to Predict the Calories by Using Intelligent Deep Learning Strategy. In *2023 International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication Engineering (RMKMATE)*. 1–7. https://doi.org/10.1109/RMKMATE59243.2023.10369431

[11] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. 2020. DropEdge: Towards Deep Graph Convolutional Networks on Node Classification. arXiv:1907.10903 [cs.LG] https://arxiv.org/abs/1907.10903

[12] Robin Ruede, Verena Heusser, Lukas Frank, Alina Roitberg, Monica Haurilet, and Rainer Stiefelhagen. 2020. Multi-Task Learning for Calorie Prediction on a Novel Large-Scale Recipe Dataset Enriched with Nutritional Information. arXiv:2011.01082 [cs.CV] https://arxiv.org/abs/2011.01082

[13] Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data* 6, 1 (2019), 1–48.

[14] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. arXiv:1703.01365 [cs.LG] https://arxiv.org/abs/1703.01365

[15] Quin Thames, Arjun Karpur, Wade Norris, Fangting Xia, Liviu Panait, Tobias Weyand, and Jack Sim. 2021. Nutrition5k: Towards Automatic Nutritional Understanding of Generic Food. arXiv:2103.03375 [cs.CV] https://arxiv.org/abs/2103.03375

[16] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. arXiv:1710.10903 [stat.ML] https://arxiv.org/abs/1710.10903

[17] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. 2018. Non-local Neural Networks. arXiv:1711.07971 [cs.CV] https://arxiv.org/abs/1711.07971

[18] Muhan Zhang and Yixin Chen. 2018. Link Prediction Based on Graph Neural Networks. arXiv:1802.09691 [cs.LG] https://arxiv.org/abs/1802.09691