Optimizing Machine Learning Models for Accurate Nutritional Value Prediction

Anika Kumar
[1]Gunn High School, [2]Foothill College

INSPIRATION

- **Goal**: Minimize food waste, assist individuals with special health conditions such as diabetes and high blood pressure
- **Solution**: personalized nutrition, and thus, accurate nutritional value prediction
- **Existing research**: meal descriptions and large language models or image inputs, rather than simply names of words and convolutional neural networks
  - BERT and other commonly used semantic analyzation techniques are often used for classification rather than numerical regression
- **This research:** Usage of a single word as an input rather than the conventional illustrative meal descriptions with neural networks
  - Convenience to users
  - Innovative solution to a novel problem

RESEARCH METHODOLOGIES

1. **Multi-Output Regressor with Custom Neural Network**
- Multi-Output Regressor extends the parameterized model type over the desired thirty-four nutritional values such as those shown in *Figure 1*
- Parameterized model fits over the preprocessed training input values and scaled nutritional values
  - Employs the KerasRegressor with custom layers and activation function
- Input layer: eight neurons after padding, afterwards embedded and flattened
- Output layer: one neuron after scaling

2. **Neural Network with Directly 34 Outputs (no Multi-Output Regression)**
- Input layer: eight neurons
- Output layer, thirty-four neurons, each representing a nutritional value
  - 34 outputs from one network rather than one output from 34 networks

3. **GridSearchCV (no Multi-Output Regression)**
- Built-in search for the most optimal parameters over a cross-fold of 10
- Utilizing Multi-Layer Perceptron Regressor
  - Differed from the custom neural network architecture earlier, which implemented the model leveraging Keras
- Beneficial platform to validate assumptions made in the previous models: layers of the model, activation functions, and solvers

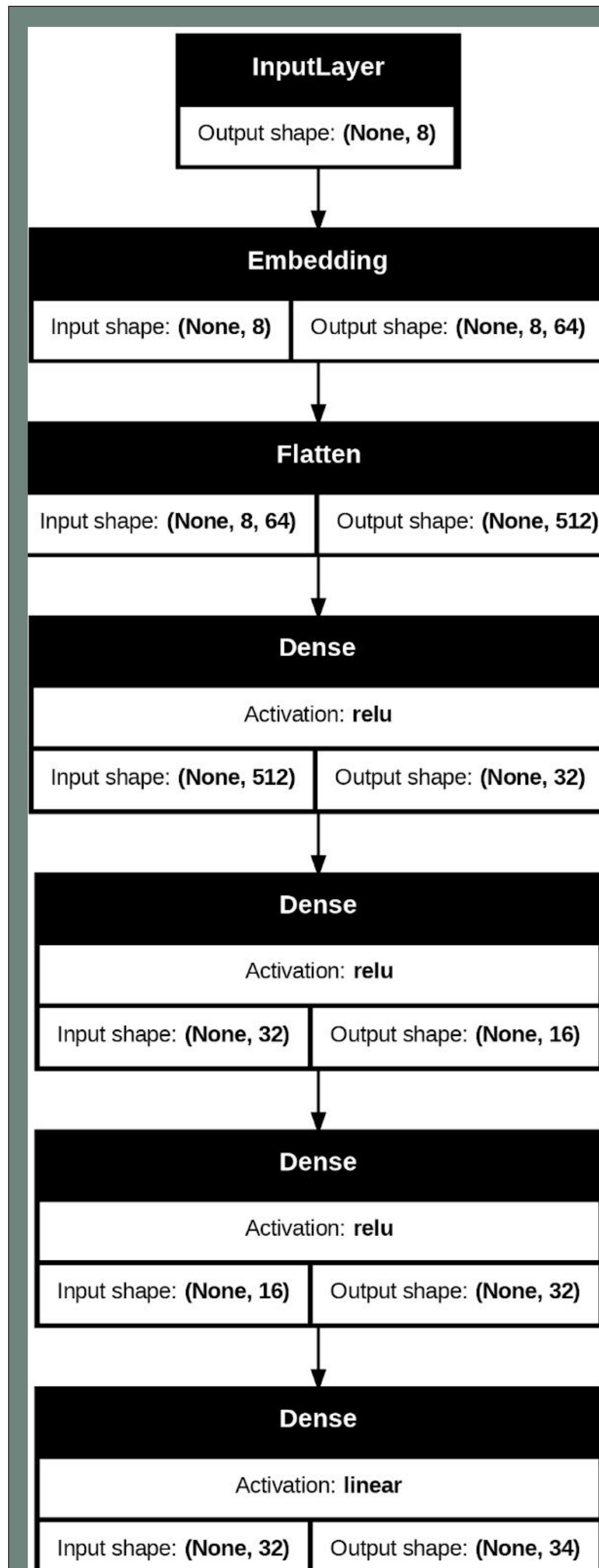
DATASET AND FINDINGS

- **Dataset:** 2,395 rows containing the name of the food as well as numerical values for macronutrients and vitamins, assuming a 100g serving size
- **Preprocessing:**
  - Tokenization using vocabulary size of 50000, ensuring encapsulation of all possible tokens
  - Padded using 'post'
  - Preprocessing outputs: StandardScaler from scikit-learn
  - Training data was obtained from 80% of the entire dataset
- Each model had three parameters independently changed: the number of epochs for training, the layers for the neural network, and the optimizer/activation function
  - For each iteration, a parameter was changed based on conventions in machine learning or previous iterations' metrics.
  - Goal: minimize the mean absolute error, which served as a metric for its accuracy. Each of the mean absolute errors were averaged over the thirty-four outputs to gain a holistic understanding of how well the model performed. We refer to this mean "mean-absolute error" metric as MMAE.

1. **Multi-Output Regressor with Custom Neural Network**
   - **Solver:** Adam performed better than stochastic gradient descent (sgd)
   - **Layers:** [68, 34, 16, 8, 4, 1], [8, 4, 1], [16, 8, 4, 1], and [32, 16, 8, 4, 1]
     - Most optimal layer pathway was [68, 34, 16, 8, 4, 1], depicted in *Figure 2*.
   - **Epochs:** 2, 3, 4, and 5
     - 5 most optimal

2. **Neural Network with Directly 34 Outputs (no Multi-Output Regression)**
   - **Solver:** Adam performed better than stochastic gradient descent (sgd)
   - **Epochs:** 2, 3, 4, 5, 6, and 7
     - 5 most optimal
   - **Layers:** [32, 34], [32, 16, 32, 34]. [2, 4, 8, 16, 32, 34], [2, 4, 8, 16, 32, 32, 34]
     - Most optimal layer pathway was [32, 16, 32, 34], depicted in *Figure 3.*
   - **Embedding dimension size:** originally, it employed an embedding dimension of 1024, however, after testing sizes 64, 68, 177 (as in model 1), and 2048, the embedding dimension of 64 resulted in the least MMAE

3. **GridSearchCV (Multi-Layer Perceptron Regressor)**
   - Parameter grid shown in *Figure 4*
   - GridSearchCV exhaustively considers all possible parameter combinations provided and fits a model over each one, evaluating each and outputting the best combination
   - Adam better than the stochastic gradient descent (sgd)
   - The rectified linear unit (relu) function performed better than the logistic and tanh functions
     - Results backed assumption for utilizing relu in models 1 and 2
   - Utilizing the MLPRegressor built-into scikitlearn rather than the custom neural network models using Keras
     - Most optimized dense layers of the MLPRegressor ended up being [1, 3, 5, 7, 9, 11, 13, 15, 25, 30, 34].
   - Number of epochs was left untested with the max_iter parameter (1000) being the number utilized



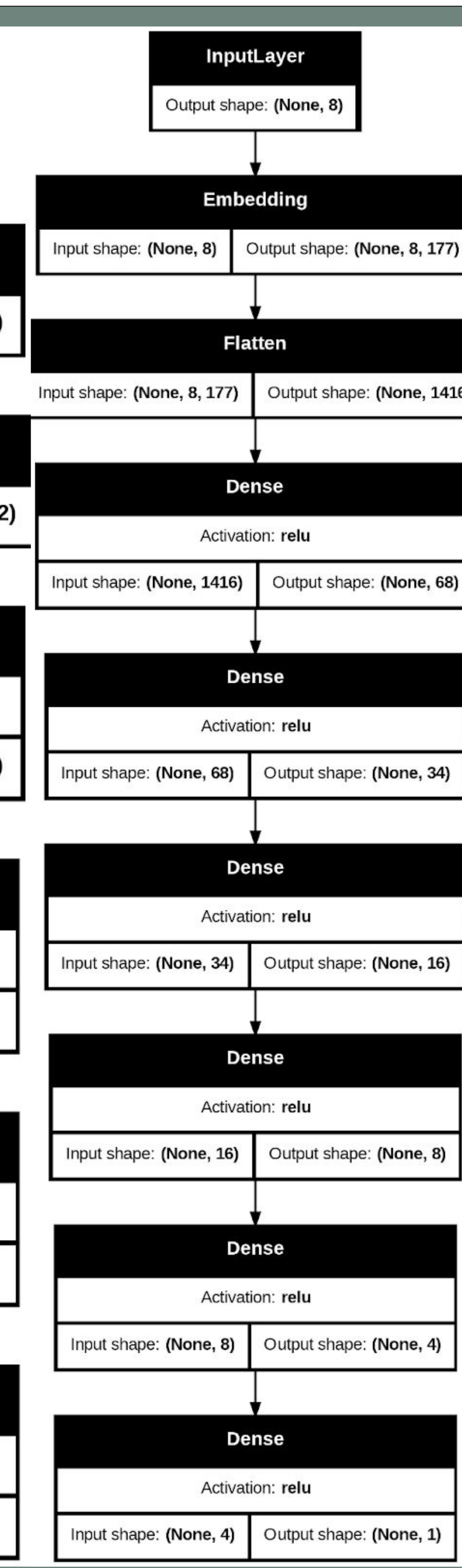


Figure 3: (left) Solely custom neural network most optimal tested layers pathway, represented graphically.
Figure 2 (right): Multi-Output Regressor's parameterized custom neural network most optimal tested layers pathway, represented graphically

food	Caloric Value	Fat	Saturated Fats	Monounsaturated Fats	Polyunsaturated Fats	Carl
cream cheese	51	5	2.9	1.3	0.2	
neufchatel cheese	215	19.4	10.9	4.9	0.8	
requeijao cremoso light catupiry	49	3.6	2.3	0.9	0.000	
ricotta cheese	30	2	1.3	0.5	0.002	
cream cheese low fat	30	2.3	1.4	0.6	0.042	
cream cheese fat free	19	0.2	0.1	0.091	0.075	
gruyere cheese	116	9.1	5.3	2.8	0.5	
cheddar cheese	113	9.3	5.3	2.6	0.3	
parmesan cheese	71	4.5	2.7	1.4	0.1	
romano cheese	19	1.3	0.9	0.4	0.035	
parmesan cheese grated	21	1.4	0.8	0.4	0.036	

Figure 1: A few data rows for example, cut off. There are 34 total columns, each corresponding to an output nutritional value.

IMPLICATIONS AND NEXT STEPS

- Expanding the dataset to encapsulate additional data rows for a breadth of training data and more accuracy
- Employing a custom neural network for each individual output—such as solely predicting sugars, carbohydrates, or caloric values based on the name of the food: rather than all thirty-four at once—may lead to better results.
  - MultiOutputRegressor model architecture took a step towards this, but utilized the same layer architecture for all outputs, leading to lack of customization.
- Employing a more diverse and methodical approach for testing optimal dense layers for the models
  - Possibly lead to smaller MMAEs
- Employ the large language approach for this model with the same data, inputs and outputs
  - See how these two different model architectures performed when compared to each other
- Similarly, employing BERT with a classification task by categorizing the outputs of the model to be intervals of a certain precision/width rather
  - Could be another model to compare

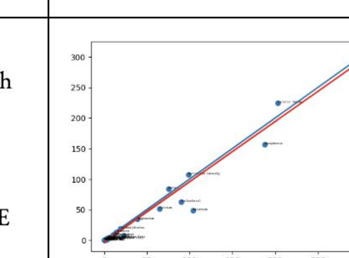
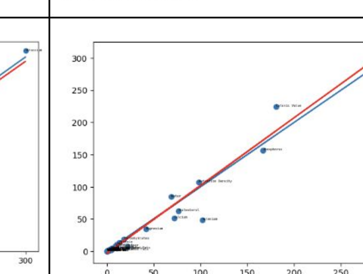
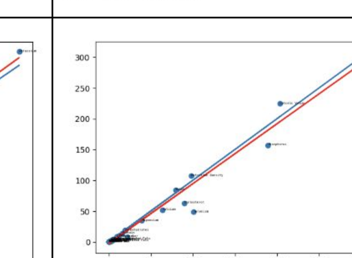
Metric	Model 1 (MultiOutput)	Model 2 (Custom CNN)	Model 3 (MLPRegressor)
MMAE	38.02783952	35.84121751	38.33789413
Training vs Testing Graph $\hat{y}$ is the predicted training MAE given x, the testing MAE, each over a nutritional value output	 *LSRL Equation:* $\hat{y} = -3.045 + 0.986x$	 *LSRL Equation:* $y = -3.230 + 1.052x$	 *LSRL Equation:* $y = -2.812 + 0.973x$

Figure 5: MMAE as defined earlier for each model as well as graphs for (testing MAE, training MAE) over each of the output variables. The red line on each graph represents the least squares regression line, and the blue line represents the line y=x, for reference.

hidden layer sizes	[8, 32, 16, 32, 34], [32, 16, 32], [8, 16, 32, 24], [1, 2, 3, 4, 34], [1, 3, 5, 7, 9, 11, 13, 15, 25, 30, 34]
Activation	relu, logistic, tanh
Solver	adam, sgd

Figure 4: GridSearch CV parameter grid setup.

CONCLUSION

Given these three models' optimization, we will now compare results across each one overall (See *Figure 5*). The custom neural network model outperformed both the MultiOutput and MLPRegressor in terms of accuracy given by the MMAE. However, it is interesting to see how close this metric ended up being, despite quite different architectures across the three models. Additionally, it appears that all three models generally did well in not overfitting with the training data, as the slopes of the least-squares-regression-line for the training MAE versus testing MAE graphs below were quite close to 1, indicating that the model predicted to about the same accuracy regardless of whether it had seen the input data or not. Although these slopes are close enough to 1 for a fair conclusion that all three models did not overfit, it appears that model 1 slightly outperformed the other two models, with a delta from the slope of 1 of 0.014 as compared to Model 2's 0.052 and Model 3's 0.027.

ACKNOWLEDGEMENTS AND REFERENCES

Special thanks to Ms. Rachael Kaci, Mr. CheeTung Leong, and Mr. Omar Ramos for the encouragement and feedback throughout my research.

Andong et al. (2024, July 4). *NutriBench: A Dataset for Evaluating Large Language Models in Carbohydrate Estimation from Meal Descriptions*. arXiv.org. https://arxiv.org/abs/2407.12843.

Arnav R. (2022, March 22). Scikit-Learn Solvers explained. Medium. https://medium.com/@arnavr/scikit-learn-solvers-explained-780a17bc322d

Michelle et al. (2024, October 21). *NutrifyAI: An AI-powered system for real-time food detection, nutritional analysis, and personalized meal recommendations*. arXiv.org. https://arxiv.org/abs/2408.10532.

Shafaat J. Rokon et al. (2022, March 13). Food Recipe Recommendation Based on Ingredients Detection Using Deep Learning. Arxiv. https://arxiv.org/pdf/2203.06721

Keras Documentation: *Keras 3 API documentation*. https://keras.io/api/.

Numpy reference. NumPy reference - NumPy v2.2 Manual. (n.d.) https://numpy.org/doc/stable/reference/index.html.

Programme, U. N. E. (2024, March 27). Food Waste Index Report 2024. Think Eat Save: Tracking Progress to Halve Global Food Waste. UN Environment Document Repository Home. https://wedocs.unep.org/handle/20.500.11822/45230
Scikit-learn API Reference. https://scikit-learn.org/stable/api/index.html.