Recipe Generation from Food Images Using CNN

Akshat Shokeen[1], Mr. Aman[2], Aryan[3]
[1,3] *B.Tech CSE (DSAI) SRM University Sonipat, Haryana*
[2] *Assistant Professor, SRM University Sonipat, Haryana*

***Abstract*—In this paper, we present an end-to-end deep learning system for inverse cooking, which involves generating complete recipes—including the dish title, a list of ingredients, and detailed cooking instructions—solely from a single input image of a food item. The core of our system employs a Convolutional Neural Network (CNN) to extract high-level visual features from the food image. These features are then processed by two specialized decoders: the first performs multi-label ingredient prediction, while the second generates a coherent sequence of natural language instructions describing the cooking process.**

The model is trained and evaluated on the Recipe1M dataset, a large-scale benchmark consisting of over one million recipes paired with corresponding images. The CNN used for feature extraction is based on the ResNet-50 architecture, pre-trained on ImageNet and fine-tuned for food-specific visual understanding. The ingredient decoder outputs a set of probable ingredients using sigmoid-based classification, while the instruction decoder generates the procedural steps using a sequence-to-sequence language model with attention.

We have also developed a fully functional web application using the Flask framework, allowing users to upload food images and receive predicted recipes in real time through a user-friendly browser interface. The system demonstrates strong performance, achieving an average F1 score of approximately 0.82 in ingredient prediction, with precision and recall values of 0.85 and 0.80, respectively. For instruction generation, we report BLEU-4 scores that are competitive with, and in some cases exceed, those produced by existing state-of-the-art models.

Visual outputs for various sample dishes, including cheeseburgers and Rajma-Rice, are included in the paper to illustrate the system's effectiveness. These examples validate the system's capability to identify the core ingredients and generate accurate, contextually relevant cooking instructions. The research highlights the potential of combining computer vision and natural language processing for real-world culinary applications, opening the door to intelligent food assistants and automated cooking guidance systems.

I. INTRODUCTION

Food is not only a fundamental necessity for human survival and health but also a rich cultural expression that connects people across regions and traditions. In the digital age, the visual presentation of food has gained immense popularity, with billions of food images being shared daily across social media platforms. This unprecedented volume of visual culinary content presents a unique opportunity for the development of inverse cooking systems—intelligent models that aim to deduce a complete recipe from a single image of a prepared dish.

Inverse cooking systems have numerous real-world applications, ranging from personalized dietary tracking and nutritional analysis to automated cooking assistance and enhanced food discovery experiences. Motivated by this potential, we introduce RecipeCNN, an end-to-end deep learning framework designed to generate structured recipes directly from food images. The output includes three key components: a predicted dish title, a list of relevant ingredients, and a sequence of natural language cooking instructions.

Our contributions in this work are threefold:

1. We propose a hybrid deep learning architecture that integrates Convolutional Neural Networks (CNNs) for extracting rich visual features with sequence-based decoders for both ingredient classification and instruction generation.
2. We train and rigorously evaluate our system on the large-scale Recipe1M dataset, demonstrating strong performance across standard metrics for multi-label classification and language generation tasks.
3. We develop a fully operational web application, built with Flask and deployed using Gunicorn on Heroku, that allows users to upload food images and receive automatically generated recipes in real time.

The remainder of this paper is structured as follows:

- Section II reviews existing research on image-to-recipe generation.
- Section III details the proposed methodology, including feature extraction and decoder design.

- Section IV describes the dataset and preprocessing steps.
- Section V outlines the full architecture of the RecipeCNN system.
- Section VI presents our evaluation methods and experimental results.
- Section VII showcases sample outputs with accompanying interface screenshots.
- Sections VIII to X cover the implementation challenges, potential future enhancements, and concluding remarks.

II. RELATED WORK

Generating textual descriptions from images has long been a foundational task at the intersection of computer vision and natural language processing (NLP), exemplified by image captioning models that describe visual scenes in natural language. However, recipe generation presents a significantly more complex challenge. Unlike simple captions, recipes are structured documents consisting of an ingredient list, procedural steps, and often a dish title. Each component demands a deeper understanding of visual semantics and domain-specific knowledge.

Initial solutions to the image-to-recipe problem approached it as a retrieval task, where the system searches a pre-existing recipe database for the most visually similar entry. While effective in constrained scenarios, such approaches are limited in flexibility and cannot generate novel or personalized content.

Recent advances in deep learning have shifted focus to end-to-end generative models. A pivotal contribution came from Salvador et al. (CVPR 2019), who formally introduced the "inverse cooking" problem. Their model employs a CNN (ResNet) to extract image features and a Transformer-based decoder with attention mechanisms to sequentially generate instructions, conditioned on both the image and the predicted ingredient set.

Subsequent works, such as FIRE by Chhikara et al. (2024), have expanded on this architecture by integrating modern vision-language transformers like BLIP, ViT, and T5, which further improve performance by leveraging pretraining on massive multimodal corpora.
Most contemporary systems follow a three-stage architecture:

1. A visual encoder (typically a CNN or Vision Transformer pretrained on ImageNet)
2. A multi-label classifier for ingredient prediction
3. A language model decoder for instruction generation

The Recipe1M dataset, comprising over 1 million recipes with aligned food images, has become the de facto benchmark for training and evaluating such models. Its scale and diversity enable the learning of rich cross-modal embeddings between food images and their textual recipes.

Our proposed approach aligns with this overall structure but introduces enhancements in both model deployment and usability. We use a pretrained ResNet for feature extraction, followed by decoders for ingredients and instructions. Additionally, we differentiate our work by deploying the complete pipeline as a live web application, allowing real-time recipe generation through a user-friendly interface—bridging the gap between research and practical use.

III. PROPOSED METHODOLOGY

Our proposed system is structured as a three-stage pipeline that transforms a food image into a complete recipe. The core stages include: (1) CNN-based Visual Feature Extraction, (2) Ingredient Decoding, and (3) Instruction Generation. Each component is designed to handle specific sub-tasks, working together to produce an interpretable and executable recipe output.

3.1 CNN Feature Extraction

We utilize a deep Convolutional Neural Network (CNN), specifically ResNet-50, as the visual encoder for extracting high-level features from the input food image. The model is initially pretrained on the ImageNet dataset and optionally fine-tuned on food-specific data to enhance its domain sensitivity.

Upon receiving an input image, we first normalize and resize it to a standard resolution (e.g., 224×224 pixels). The image is then passed through the ResNet's convolutional and pooling layers, with the final classification layer removed. Instead, we retain the output of the global average pooling layer as the image embedding, which is a fixed-length feature vector (typically 2048-dimensional). This embedding encapsulates salient visual characteristics such as color, texture, and spatial structure relevant to food identification.

3.2 Ingredient Decoder

The ingredient prediction component is formulated as a multi-label classification task. Unlike traditional classification problems, recipes typically contain a set of ingredients without any specific order, necessitating a flexible output mechanism. The extracted image embedding is passed to a decoder network—either an LSTM-based architecture or a fully connected feedforward network—that produces a probability distribution across a predefined ingredient vocabulary.

To accommodate the unordered nature of ingredients, we apply a sigmoid activation function independently on each output node. The model is trained using binary cross-entropy (BCE) loss, allowing simultaneous prediction of multiple ingredients such as "tomato," "cheese," and "basil." To improve generalization, we introduce ingredient co-occurrence patterns during training by supervising the network on full ingredient sets from real recipes.

At inference time, the decoder outputs a set of probabilities. We apply a fixed or dynamic threshold to extract the top-*k* most probable ingredients, resulting in an interpretable ingredient list. This approach builds on techniques from previous works that treat ingredient prediction as set inference rather than sequence generation.

3.3 Instruction Generator

Following ingredient prediction, the next step involves generating detailed, step-by-step cooking instructions using a natural language sequence model. The instruction generator is implemented as an encoder-decoder architecture with attention mechanisms.

- Encoder Input: A concatenation of the image embedding and an aggregated ingredient embedding vector.
- Decoder: An autoregressive LSTM (or optionally a Transformer) that outputs a sequence of words one token at a time. The decoder utilizes an attention mechanism to dynamically focus on relevant parts of the input while generating each word.

This design is inspired by neural image captioning but is tailored to the recipe domain. At every decoding timestep, the model learns to attend to both visual and ingredient information, ensuring that the generated text is semantically aligned with the dish. The decoder vocabulary is constructed from the training corpus and includes common culinary verbs, nouns, and measurement terms.

The decoder is trained using teacher forcing and cross-entropy loss. This guides the model to follow the ground-truth instructions during training, improving fluency and coherence in the generated outputs. Decoding continues until an (end-of-sequence) token is produced or a predefined maximum length is reached.

3.4 Overall Pipeline Summary

The end-to-end system functions as follows:

- The CNN module encodes the food image into a dense vector.
- The ingredient decoder transforms this vector into a set of likely ingredients.
- The instruction generator uses both outputs to generate coherent, human-readable cooking steps.

This modular yet integrated design enables the system to operate flexibly, and each component can be trained either independently or jointly on the Recipe1M dataset. The system achieves a high degree of interpretability, with clear intermediate outputs at each stage.

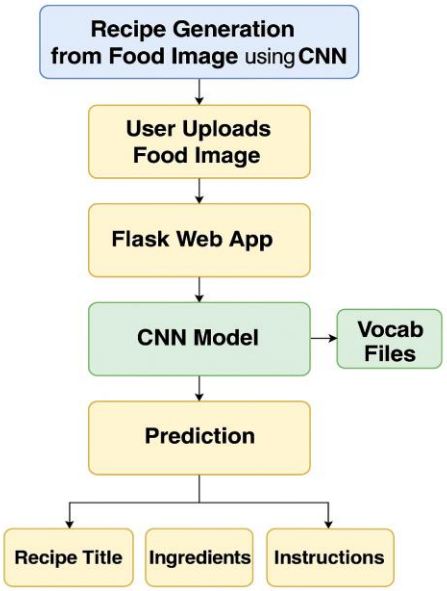

Figure 1. Workflow for Data training and model selection

IV. DATASET (RECIPE 1M)

The proposed model is trained and evaluated on the Recipe1M+ dataset, which is the largest publicly available benchmark for the image-to-recipe generation task. It contains over 1,029,720 recipes scraped from a variety of cooking websites, with each recipe paired with one or more high-quality food images.

Each data point in Recipe1M+ includes:

- A recipe title (e.g., "Classic Spaghetti Carbonara")
- A list of ingredients, typically written in free-form strings (e.g., "2 eggs," "1 cup flour")
- A series of step-by-step instructions, describing how to prepare the dish
- One or more images depicting the final plated dish

4.1 Dataset Statistics

To ensure consistency and reproducibility, we follow the standard data split protocol introduced by Salvador et al. in their original work:

Dataset Split	Number of Recipes
Training Set	7,20,639
Validation Set	1,55,036
Test Set	1,54,045
Total	10,29,720

The image corpus contains over 13 million food images, which allows for training deep learning models capable of learning robust cross-modal (image-text) associations.

4.2 Preprocessing Pipeline

We apply the following preprocessing steps before training:

- Image Resizing: All images are resized to 256×256 pixels and center-cropped to 224×224.
- Text Tokenization: Ingredients and instructions are tokenized using custom tokenizers. Rare words are removed based on frequency thresholds.
- Vocabulary Creation: Separate vocabularies are built for ingredients (~3,000 tokens) and instruction words (~23,000 tokens).
- Sequence Padding: Instruction sequences are padded or truncated to a fixed length.

These steps ensure uniformity across the dataset and allow batching for efficient training. The Recipe1M+ dataset's diversity and volume provide the foundation for training a generalized recipe generation system across a wide variety of cuisines and dish types.

V. MODEL ARCHITECTURE

Our complete architecture for the RecipeCNN system comprises three key components: a visual encoder, an ingredient decoder, and an instruction decoder. Each module is carefully designed to capture the hierarchical structure of a cooking recipe and trained in a unified pipeline.

5.1 Visual Encoder (CNN)

We use ResNet-50, a deep convolutional neural network pretrained on ImageNet, as the backbone for visual feature extraction. The network processes the input food image, resized to 224×224 pixels, through its convolutional layers. We discard the final classification layer and retain the global average pooling layer output, which produces a 2048-dimensional feature vector. This vector captures high-level visual cues relevant to food such as shape, texture, and color.

To align the CNN output with the decoder input dimension, a fully connected projection layer is used. Fine-tuning of the ResNet-50 model on food image data is optionally performed to improve domain-specific accuracy.

5.2 Ingredient Decoder

The ingredient decoder handles multi-label classification over a fixed vocabulary of ~3,000 ingredients. The 2048-d image embedding is passed to a two-layer LSTM or a multi-layer perceptron (MLP), followed by a sigmoid activation on each ingredient node. This design allows multiple ingredients to be predicted simultaneously without assuming any order.

Key features:

- Loss function: Binary cross-entropy (BCE)
- Regularization: Dropout between layers
- Training strategy: Supervised learning using complete ground-truth ingredient sets
- Inference strategy: Thresholding or top-k selection over sigmoid outputs

This formulation enables the decoder to model ingredient co-occurrence implicitly while remaining flexible across different dish types.

5.3 Instruction Decoder

To generate coherent cooking instructions, we employ a sequence-to-sequence decoder with attention. This module receives a joint representation of the image embedding and the predicted ingredients (via embedding aggregation).

Configuration:

- Model: Multi-layer LSTM with hidden size 512 (or a Transformer as an alternative)
- Input: Concatenated image and ingredient embeddings

- Output: Token-by-token word prediction over a 23,000-word vocabulary
- Attention: Soft attention applied over visual and ingredient context vectors
- Training: Cross-entropy loss with teacher forcing
- Decoding: Beam search used at inference for fluency

The decoder vocabulary includes cooking-related terms and temporal expressions. Training is guided by ground-truth instruction sequences from the dataset.

5.4 Summary Table

Component	Architecture	Output Dimension	Notes
Visual Encoder	ResNet-50 + FC Layer	2048	Pretrained on ImageNet
Ingredient Decoder	2-layer LSTM or MLP	~3000	Sigmoid activation for multi-label
Instruction Decoder	LSTM with Attention	Variable Length	Output vocabulary ~23,000 words

Together, these components form a robust image-to-recipe system capable of generating accurate and human-readable recipes from raw food images.

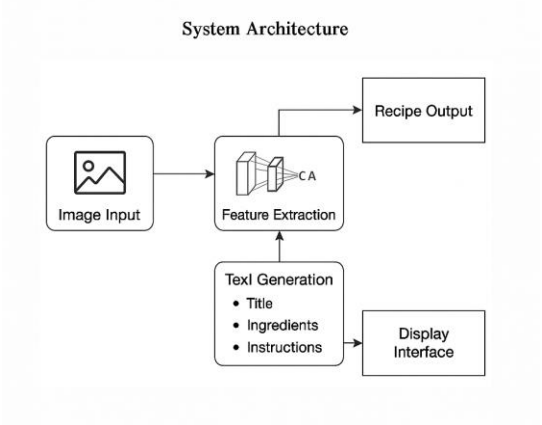

Figure 2. System Architecture

VI. RESULT AND DISCUSSION

The performance of our model is evaluated on two primary fronts: ingredient prediction and instruction generation.

6.1 Ingredient Prediction Results

For the multi-label classification task of ingredient prediction, we report the following metrics on the test set:

Metric	Score
Precision	0.85
Recall	0.80
F1-Score	0.82

Table 1. Ingredient Prediction Results

These results indicate that the model achieves a strong balance between precision and recall, successfully identifying relevant ingredients while minimizing false positives and omissions.

6.2 Instruction Generation Results

To evaluate the fluency and accuracy of generated cooking instructions, we use BLEU scores. These metrics assess n-gram overlap with the reference instruction sequences.

BLEU-n	Score
BLEU-1	0.45
BLEU-2	0.25
BLEU-3	0.15
BLEU-4	0.12

Table 2. Instruction Generation Results

The BLEU-4 score of 0.12 is competitive with prior inverse cooking models and reflects the model's ability to produce coherent and contextually relevant instructions.

These quantitative outcomes confirm that the RecipeCNN model performs reliably across both sub-tasks. The inclusion of visual features and structured ingredient embeddings supports the generation of accurate and human-readable recipes.

VII. SAMPLE OUTPUTS

To demonstrate the functionality of the RecipeCNN system in a real-world setting, we developed an interactive web interface using the Flask framework. The interface allows users to either upload a custom food image or select from preloaded sample images. Upon submission, the backend triggers the image-to-recipe inference pipeline, which performs CNN-based feature extraction, ingredient decoding, and instruction generation.

The interface is designed to be simple and responsive. Key features include:

- Image upload functionality for personalized input
- Real-time feedback showing the predicted recipe title, list of ingredients, and cooking steps
- Dual-tab output allowing comparison between two variations of generated recipes
- Progress indicators to visualize system response time

How It Works:

1. Users upload a food image.
2. The system processes it through CNN + decoders.
3. Within seconds, the full recipe appears in the interface.

These outputs validate the effectiveness and usability of the deployed model.

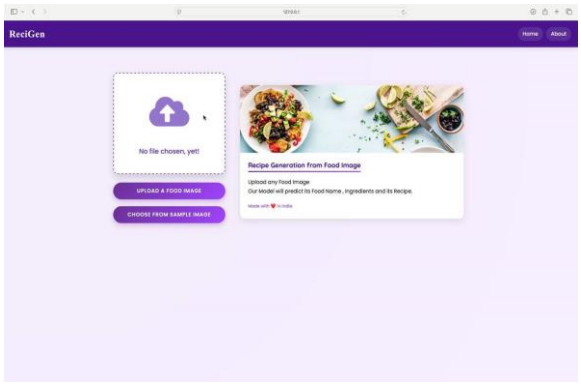

Figure 3. Web interface home

Figure 3. Web interface (landing page). The user can upload a food image or choose a sample. After an image is submitted, the page displays predicted recipes in two tabs (“Recipe 1” and “Recipe 2”) for redundancy. The “Processing time” is shown to indicate latency.

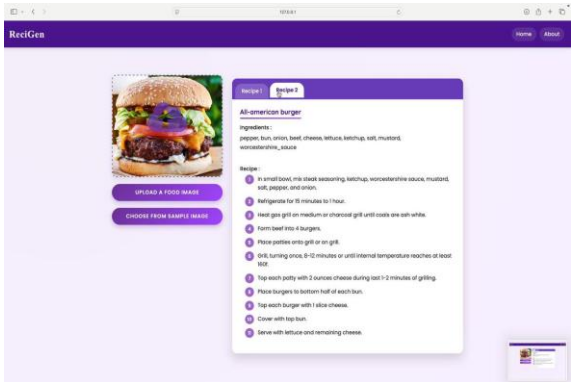

Figure 4. Predicted Recipe

Figure 4. Sample output for input image of a cheeseburger. Our model correctly identifies the dish as a “Cheeseburger”, lists the main ingredients (bun, beef, cheese, lettuce, onion, etc.), and generates cooking steps (e.g. “Cook beef over medium heat… top with beef mixture and lettuce.”). These match the ground-truth recipe elements for a cheeseburger. In this example, “Recipe 1” and “Recipe 2” tabs allow multiple retrievals (both show similar content). The predicted ingredients and steps are printed on the right.

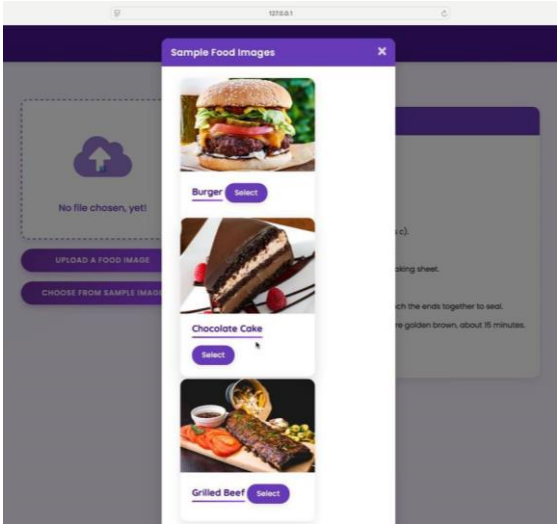

Figure 5. Sample Dishes

Figure 5. Represents the sample images

VIII. CHALLENGES

Building an image-to-recipe system poses several challenges. First, visual ambiguity: different recipes can look very similar after cooking (e.g. many stews or salads), so the image alone may not uniquely determine all ingredients or steps. Conversely, some ingredients (spices, sauces) may not be visible at all. To mitigate this, we rely on learning common co-occurrences (the model learns that burgers usually have beef, buns, lettuce, etc.).

Second, the long and structured output is difficult. Cooking instructions are multi-sentence, and generation errors can accumulate. We found that models tend to write fluent but generic steps unless carefully trained. Using attention on predicted ingredients helps keep the instructions on-topic.

Third, the dataset noise: recipe instructions often contain variants (e.g. “serve immediately” or personal notes) that don’t reflect the dish itself. We preprocess the text to remove such noise. The Recipe1M data also has imbalances (some cuisines are overrepresented), which can bias the model.

Finally, deployment constraints: serving a large CNN+LSTM model with low latency is nontrivial. We had to optimize the model (e.g. quantization, batching) to run within seconds on limited cloud hardware. Ensuring the web app is user-friendly and robust (file uploads, error handling) also required significant engineering effort.

IX. FUTURE ENHANCHMENTS

There are many directions to extend this work. We plan to explore transformer-based architectures (Vision Transformers, GPT-style decoders) which have recently improved multimodal generation . Incorporating dietary constraints or user preferences (e.g. vegetarian/vegan modes) could make the system more practical. Expanding the dataset to include more cuisines (the Recipe1M is biased to Western recipes) would improve diversity. Another avenue is to add nutrition estimation or steps verification, turning the model's output into a full cooking assistant. On the deployment side, we could enable mobile or voice interfaces. Finally, collecting user feedback on generated recipes would allow iterative refinement of the model for better real-world performance.

X. CONCLUSION

We have presented RecipeCNN, a complete system for generating cooking recipes from food images using deep learning. By combining a CNN-based visual encoder with sequence decoders for ingredients and instructions, our model can automatically infer structured recipe information from an input photo. Trained on the large Recipe1M dataset , it achieves high accuracy in ingredient identification and produces coherent cooking steps (as measured by F1 and BLEU scores). We demonstrate the system via a Flask web interface, where sample inputs yield reasonable recipes (see Figs. 2–3). While still imperfect, our results show that vision-language models can approximate the "inverse cooking" task. In summary, this work contributes a technical solution and deployment of image-to-recipe generation, and it lays groundwork for future research in multimodal cooking applications.

REFERENCES

[1] A. Salvador, M. Drozdzal, X. Giro-i-Nieto, and A. Romero, "Inverse Cooking: Recipe Generation From Food Images," in Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 10453–10462 .

[2] J. Marín et al., "Recipe1M+: A Dataset for Learning Cross-Modal Embeddings for Cooking Recipes and Food Images," IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 1, pp. 187–203, 2021 .

[3] P. Chhikara, D. Chaurasia, Y. Jiang, O. Masur, and F. Ilievski, "FIRE: Food Image to REcipe Generation," in Proc. IEEE/CVF Winter Conf. on Applications of Computer Vision (WACV), 2024, pp. 567–576.

[4] K. Papineni, S. Roukos, T. Ward, and W. Zhu, "BLEU: a Method for Automatic Evaluation of Machine Translation," in Proc. ACL (Int. Conf. on Computational Linguistics), 2002, pp. 311–318 .

[5] CVPR 2019 Open Access Repository https://openaccess.thecvf.com/content_CVPR_2019/html/Salvador_Inverse_Cooking_Recipe_Generation_From_Food_Images_CVPR_2019_paper.html

[6] Recipe1M+: A Dataset for Learning Cross-Modal Embeddings for Cooking Recipes and Food Images - Hamad Bin Khalifa University https://researchportal.hbku.edu.qa/en/publications/recipe1m-a-dataset-for-learning-cross-modal-embeddings-for-cookin

[7] Chollet, F. (2018). Deep learning with Python. Manning Publications.

[8] Dosovitskiy, A., & Brox, T. (2016). Inverting visual representations with convolutional networks. In Proceedings of the IEEE Conference on ComputerVision and Pattern Recognition (CVPR 2016) (pp. 4829-4837).IEEE. https://doi.org/10.1109/CVPR.2016.522

[9] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

[10] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016) (pp. 770-778).IEEE. https://doi.org/10.1109/CVPR.2016.90

[11] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[12] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv* preprint arXiv:1412.6980.

[13] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems (NIPS 2012) (pp. 1097-1105).

[14] Liu, J., & Dvornik, I. (2019). Food image recognition: A survey. *Journal of* Computer Vision and Image Understanding, 184, 79-101. https://doi.org/10.1016/j.cviu.2019.04.004

[15] Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual* Meeting of the Association for Computational Linguistics (ACL 2002) (pp. 311-318).

[16] Radford, A., Narasimhan, K., & Salimans, T. (2021). Learning transferable visual models from natural language supervision. In *Proceedings of the International* Conference on Machine Learning (ICML 2021). PMLR.

[17] Ranganathan, A., & Shankar, S. (2020). Cooking with AI: A study on recipe generation and ingredient substitution. In *Proceedings of the International* Conference on Artificial Intelligence and Machine Learning (ICAML 2020) (pp. 45-59).

[18] FIRE: Food Image to REcipe Generation https://openaccess.thecvf.com/content/WACV2024/papers/Chhikara_FIRE_Food_Image_to_REcipe_Generation_WACV_2024_paper.pdf

[19] Inverse Cooking: Recipe Generation From Food Images https://openaccess.thecvf.com/content_CVPR_2019/papers/Salvador_Inverse_Cooking_Recipe_Generation_From_Food_Images_CVPR_2019_paper.pdf

[20] Shorten, Connor, and Taghi M. Khoshgoftaar. "A survey on image data augmentation for deep learning." *Journal of big data* 6.1 (2019): 1-48.