Image Captioning for Food Datasets: A Deep Learning Approach

Agustina Ghelfi
Universitat Autònoma de Barcelona
Barcelona, España
agustina.ghelfi@autonoma.cat

María José Millán
Universitat Autònoma de Barcelona
Barcelona, España
mariajose.millan@autonoma.cat

Laila Aborizka
Universitat Autònoma de Barcelona
Barcelona, España
lailamohamed.aborizka@autonoma.cat

Abstract

Image captioning bridges computer vision and NLP by generating textual descriptions from images. This project explores methods ranging from encoder-decoder models (ResNet + LSTM/GRU) to transformer-based architectures (ViT-GPT2, Qwen-2.5, ViT-Llama) using Food Ingredients and Recipes Dataset . We evaluate performance with BLEU, METEOR, and ROUGE metrics, highlighting the impact of model choice on caption quality. Our findings demonstrate the effectiveness of attention mechanisms and multimodal pretraining, while identifying challenges in generalization and evaluation.

Keywords*: Image Captioning, Transformers, Multimodal Models, Deep Learning.*

1. Introduction

Image captioning is a fundamental task in computer vision and natural language processing that involves generating descriptive text based on visual input. It combines deep learning techniques from both domains, requiring models to extract meaningful visual features from images and translate them into coherent and contextually relevant text. Over the years, various approaches have been developed, ranging from traditional encoder-decoder architectures to more advanced methods incorporating attention mechanisms and transformer-based models.

The objective of this project is to explore different methodologies for image captioning, starting from the basics and gradually incorporating more sophisticated techniques. The project follows a structured learning path, beginning with data preprocessing and baseline models, and advancing towards modifications and improvements that align with state-of-the-art approaches. This structured approach allows us to assess model performance both quantitatively and qualitatively while iteratively improving the results.

For this study, we use the Food Ingredients and Recipes Dataset, which consists of 13,582 images of various dishes. The primary goal is to develop models capable of predicting dish titles based on image inputs. The dataset is divided into 80% training, 10% validation, and 10% test, ensuring a robust evaluation framework.

Throughout this project, we aim to gain deeper insights into the key components that contribute to effective image captioning systems and push the boundaries of traditional methods by exploring more advanced techniques.

2. Related work

Image captioning is a multidisciplinary research area that intersects computer vision and natural language processing (NLP) to generate textual descriptions of images. Over the years, various approaches have emerged, evolving from handcrafted rules and retrieval-based methods to modern deep learning-based architectures.

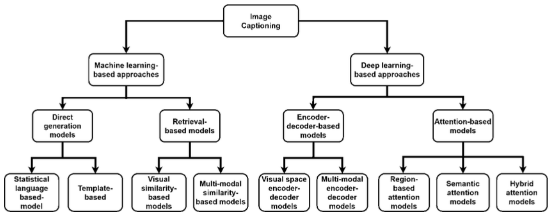


Image Captioning
Machine learning-based approaches
Deep learning-based approaches
Direct generation models
Retrieval-based models
Encoder-decoder-based models
Attention-based models
Statistical language based-model
Template-based
Visual similarity-based models
Multi-modal similarity-based models
Visual space encoder-decoder models
Multi-modal encoder-decoder models
Region-based attention models
Semantic attention models
Hybrid attention models

Figure 1. A comprehensive taxonomy of image captioning [8].

Early methods in image captioning primarily relied on template-based and retrieval-based techniques. In retrieval-based captioning, captions were either generated using predefined sentence structures or retrieved from an existing pool based on image similarity. The process typically involved identifying visually similar images from a dataset, and using methods like Markov Random Fields and Lin similarity to measure semantic distance between the query and training images. Some approaches directly retrieved captions, while others generated new ones based on the retrieved images. Techniques like Analysis of Kernel Canonical Correlation and cosine similarity were used to rank and select the most relevant phrases [9]. However, these approaches lacked flexibility and struggled to generalize well to unseen images, often failing to produce meaningful captions for novel object combinations or complex scenarios. While they generated grammatically correct captions, they were limited in their ability to adapt to new image contexts, making them less effective for more diverse or dynamic image sets [11].

With the advent of deep learning, encoder-decoder architectures became the dominant paradigm. The first deep learning-based models utilized Convolutional Neural Networks (CNNs) for feature extraction and Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) networks, for sequential text generation [12]. While these models significantly improved caption quality, they struggled with long-range dependencies and lacked a mechanism to focus on specific image regions.

To address this, attention mechanisms were introduced, allowing models to dynamically focus on relevant parts of an image while generating each word. Notable variants include soft attention, which learns a probabilistic weight distribution over image regions, and hard attention, which stochastically selects regions at each time step. More recently, self-attention and transformer-based architectures, such as Vision Transformers (ViTs) and multimodal transformers, have further improved captioning performance by capturing long-range dependencies and integrating vision and language more effectively [1, 11].

Another important development in image captioning is the use of reinforcement learning techniques, such as self-critical sequence training (SCST), which optimizes non-differentiable evaluation metrics like BLEU, CIDEr, and METEOR. These approaches enhance fluency and coherence beyond traditional maximum likelihood estimation [12]. Additionally, some works integrate graph-based representations and external knowledge bases to enrich captioning models with structured semantic relationships, leading to more contextually aware descriptions.

The field has also expanded into domain-specific applications, particularly in medical image captioning. Unlike general image captioning, where datasets such as MS COCO and Flickr30k are commonly used, medical image captioning requires specialized datasets like IU X-ray and MIMIC-CXR. These domain-specific models incorporate structured medical knowledge and leverage hierarchical learning techniques to generate accurate and clinically meaningful reports [1, 12].

Despite these advancements, several challenges remain. Current models still struggle with generating captions that are truly human-like in terms of coherence, diversity, and commonsense reasoning. Many image captioning models exhibit biases toward frequent objects and fail to describe rare or unseen entities effectively. Furthermore, evaluation metrics for image captioning remain a challenge, as automated scores often do not fully capture the quality of human-like descriptions [11].

Future research directions include enhancing multimodal pretraining techniques, leveraging large-scale vision-language models, and improving explainability in caption generation. The integration of external knowledge graphs, commonsense reasoning, and more sophisticated linguistic structures will also be crucial in advancing the field further.

3. Methodology

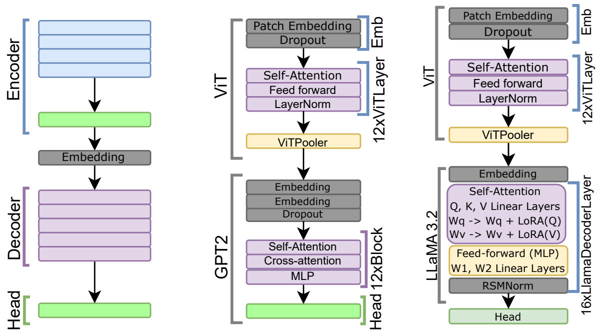
3.1. Base Architecture for Caption Generation

The baseline model was designed with an encoder-decoder architecture (Fig. 2a). In image captioning, the encoder is responsible for extracting meaningful visual representations from the input image, typically using CNNs to process and output high-level feature maps. The decoder interprets the visual features provided by the encoder and generates a corresponding natural language description, word by word. RNNs are commonly employed for this task due to their ability to model sequential data. For the baseline model, the encoder utilized a ResNet-18, and the decoder was implemented with a GRU. Various alternatives for the encoder and decoder were explored to improve performance, and the results of these modifications were compared to find the most effective configuration.

3.1.1. Encoders

ResNet-18 Architecture. CNN composed of 18 layers, including convolutional layers and fully connected layers. It follows the original residual learning framework introduced by He et al., where identity shortcuts (skip connections) allow the model to learn residual functions instead of direct mappings [4]. This design significantly alleviates the vanishing gradient problem and enables the training of deeper networks. The architecture is structured as follows:

- An initial 7×7 convolutional layer with 64 filters and



(a) Baseline encoder decoder architecture.

(b) ViT-GPT2 architecture.

(c) ViT-LLaMA 3.2 architecture.

Figure 2. Model architectures used for image captioning. (a) A flexible encoder-decoder architecture where the encoder can be ResNet18 or ResNet50, and the decoder can be GRU, LSTM, or Transformer. (b) A multimodal ViT-GPT2 model: the Vision Transformer encodes the image into patch embeddings, and GPT2 decoder generates text using self- and cross-attention layers. (c) A ViT-LLaMA 3.2 model enhanced with LoRA and advanced decoder layers for efficient and scalable caption generation.

a stride of 2, followed by a 3×3 max pooling layer.

- Four residual stages, each consisting of 2 basic residual blocks. Each basic block includes two 3×3 convolutional layers with batch normalization and ReLU activation.
- A global average pooling layer at the end aggregates spatial information.
- A fully connected layer produces the final classification scores.

ResNet-50 Architecture. Deeper and more expressive variant than ResNet-18, comprising 50 layers. It utilizes bottleneck residual blocks to enhance representational capacity while maintaining computational efficiency [4]. Its design reduces the computational cost while enabling deeper models:

- Initial 7×7 convolution and max pooling.
- Four stages, each containing a different number of bottleneck blocks, with the final stage having 2048 output filters. Each bottleneck block has:
  - A 1×1 convolution to reduce dimensionality.
  - A 3×3 convolution to process spatial features.
  - Another 1×1 convolution to restore dimensionality.

3.1.2. Decoder

LSTM Architecture. The Long Short-Term Memory network is a type of recurrent neural network designed to better capture long-term dependencies by mitigating the vanishing gradient problem [2]. It does so through a memory cell and a system of gating units that regulate the flow of information. The **input gate** controls how much of the new input is added to the cell state, the **forget gate** determines the amount of past information to retain or discard from the cell state, and the **output gate** regulates the exposure of the internal cell state to the output.

The decoder is composed of one or more stacked LSTM layers, each processing the output of the previous layer. During decoding, the final hidden state from the encoder is used to initialize the LSTM decoder's hidden and cell states. The output at each time step is passed through a linear transformation followed by a softmax function to produce a probability distribution over the target vocabulary.

GRU Architecture. The Gated Recurrent Unit is a variant of the LSTM that combines the memory and gating mechanisms into a simpler structure [2]. It was introduced to improve training efficiency while maintaining competitive performance. Each GRU unit contains two gates, an **update gate** that controls the degree to which the unit updates its activation or content, and a **reset gate** that determines how much of the past information to forget.

The decoder uses stacked GRU layers, with each layer processing sequential information and producing outputs passed to the next layer. Initial hidden states for the decoder are typically derived from the final encoder states. The output of each time step is projected through a linear layer followed by softmax activation for token prediction.

Transformer Decoder Architecture. Provides a non-recurrent alternative capable of modeling long-range dependencies through attention mechanisms. Its architecture consists of several key components:

- **Self-attention mechanism:** allows the decoder to weigh the importance of previous tokens in the sequence, ensuring that each token is generated in the context of the entire caption generated so far.
- **Decoder layers:** Each layer of the Transformer Decoder includes multi-head self-attention, a feed-forward network, residual connections, and layer normalization. These components help capture long-range dependencies in the generated sequence.
- **Positional encoding:** Since the Transformer lacks a built-in notion of token order, positional encodings are added to the input embeddings to inform the model of the token's position in the sequence.

Teacher Forcing: A commonly used technique during the training of decoders based on RNNs is *teacher forcing*. In this approach, during training, instead of using the model's previous predictions to generate the next word, the actual word from the training set is fed as input for the next prediction. This helps the model by providing more accurate information, preventing the accumulation of errors over time. Also allows for faster convergence and improves performance during training. However, it can introduce a discrepancy between training and inference behavior, as during inference, the model must rely solely on its own predictions.

3.2. ViT-GPT2 Model

Vision Transformer (ViT) serves as the visual encoder, extracting image features, while GPT-2 acts as the text decoder, generating captions in natural language (Fig. 2b).

ViT Architecture: Model for image processing based on the Transformer architecture, transforms an image into a sequence of patches, processes it using a Transformer to capture global relations, and uses a classification token for image representation [3].

- **Image preprocessing**: divides the image into non-overlapping patches, flattens them, and projects them into a fixed-dimensional embedding space. A special classification token preceeds the sequence to serve as a global representation of the image. Positional embeddings are also added to retain spatial information across patches.
- **Transformer encoder**: Composed of several layers that include multi-head self-attention and feedforward MLP blocks. Each layer applies Layer Normalization before the main operation and uses residual connections afterward. The final output of the classification token serves as the overall image representation.
- **Inductive Bias**: incorporates weaker inductive biases compared to CNNs. It does not inherently capture local structures or enforce translation invariance, instead relying on the patch-based input and positional embeddings to model spatial relationships.

GPT-2 Architecture: Decoder-only Transformer model for text generation, trained to predict the next word in a sequence using causal self-attention and MLP layers for processing concept vectors [6].

- **Word Embeddings:** Words are mapped to high-dimensional vectors that are nearly orthogonal to each other, acting as concept vectors.
- **Causal Self-Attention:** Each position can only attend to previous ones, ensuring that predictions are not influenced by future tokens.
- **MLP Layers:** Projects the embeddings to a higher dimension and back. This step captures non-linear relationships between concepts.
- **Word Prediction:** The final output is projected back to the word embeddings space, and a dot product is computed with each word embedding to predict the next token using a softmax function.

3.3. Qwen-2.5 7B Multimodal

QWEN 2.5 is a Transformer-based model optimized for complex NLP and multimodal tasks.

- **Self-Attention:** Multi-head self-attention captures long-range dependencies, with dynamic scaling for improved focus on relevant tokens.
- **Adaptive Attention Span:** Dynamically adjusts attention span for efficiency, reducing redundant computations.
- **Position Encoding:** Combines absolute/relative position encodings and Rotary Position Embeddings (RoPE) for better sequence handling.
- **Multi-Modal Processing:** A cross-attention mechanism integrates text and images using dual encoders (ViT for images, Text Transformer for language).
- **Efficient Training & Inference:** Sparse attention and gradient checkpointing reduce computational cost, with Flash Attention for faster inference on accelerators.

3.4. ViT-LLaMA 3.2 Model

LLaMA 3.2 is a state-of-the-art language model developed by Meta, designed for high-performance, large-scale language understanding and generation. Built on a Transformer architecture similar to GPT-3, it features multiple layers of self-attention to capture long-range dependencies and contextual relationships. For image captioning, LLaMA 3.2 was conditioned on visual embeddings from a ViT encoder (Fig. 2c), enabling cross-modal learning where image features guide the text generation process. The model was evaluated in both its 1B and 3B parameter variants.

Efficient Fine-Tuning with LoRA. To specialize LLaMA 3.2 for the captioning task, we employed Low-Rank Adaptation (LoRA) [5], a parameter-efficient fine-tuning method that inserts trainable low-rank matrices into attention and MLP layers without modifying the full model weights. This enabled efficient adaptation with reduced memory and compute demands, allowing for task-specific tuning while maintaining the model's original capabilities.

4. Experimental Settings

4.1. Dataset

The **Food Ingredients and Recipes Dataset** (licensed under **CC BY-SA 3.0**) contains 13,582 images of food dishes [10]. Each image has a corresponding entry in a CSV file, which has the following columns:

- **Title:** The dish title.

- **Ingredients:** The list of ingredients (raw data).
- **Instructions:** Cooking instructions for the dish.
- **Image Name:** Corresponding image filename.
- **Cleaned Ingredients:** Processed ingredients list.

Data cleaning involved removing 5 rows with missing titles and deleting rows with invalid image names, resulting in 13,501 valid image-caption pairs. Figure 3 shows a word cloud analysis of the dish titles, which reveals frequent terms such as *"Salad"* and *"Chicken"*.

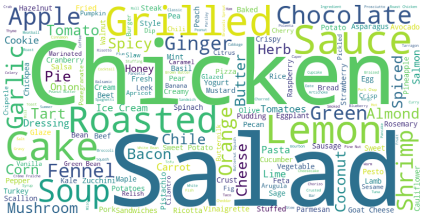


Figure 3. Word cloud analysis of the 'Food Ingredients and Recipes Dataset'.

4.2. Metrics

- **BLEU** (Bilingual Evaluation Understudy) is a metric for evaluating machine-generated translations based on n-gram precision, with a modified unigram precision. It includes a brevity penalty to balance word choice, order, and length, penalizing overly long translations while rewarding more balanced outputs. For this project, we used BLEU-1 and BLEU-2, which measure the precision of unigrams (single words) and bigrams (two consecutive words), respectively, between the generated and reference captions.
- **METEOR** (Metric for Evaluation of Translation with Explicit ORdering): enhances BLEU by emphasizing recall, considering exact matches, stemmed words, and synonyms between candidate and reference sentences. It is designed to provide a more holistic evaluation by capturing semantic equivalences and not just n-gram matches [7].
- **ROUGE** (Recall-Oriented Understudy for Gisting Evaluation): initially developed for text summarization, focuses on n-gram recall, measuring the overlap between candidate and reference n-grams. It is particularly useful for tasks where recall of important words and phrases is crucial [7].

4.3. Implementation Details

4.3.1. Environment and Setup

The programming language used for this project is Python (version 3.13.2), with PyTorch (version 2.6.0+cu124) and Hugging Face Transformers (version 4.49.0) as the primary frameworks. The hardware consists of an NVIDIA RTX 3090 GPU, with 24 GB of GPU memory. The operating system is Ubuntu 20.04.6 LTS (Focal Fossa).

4.3.2. Model-Specific Details

A. Base Architecture for Caption Generation The encoder can be configured to use either ResNet-18 or ResNet-50. The decoder can be configured as either GRU, LSTM or Trasnformer Decoder. During training, teacher forcing can be applied, where the actual ground-truth token is fed into the decoder at each time step instead of its previous prediction. This helps improve convergence and reduces error accumulation.

Depending on configuration, the optimizer can be Adam, AdamW, or SGD, with hyperparameters (learning rate, weight decay) specified externally. An early stopping mechanism monitors the validation loss (with a delta threshold of 0.001) to prevent overfitting.

B. ViT-GPT2 Model Images are preprocessed using a `ViTImageProcessor`, which resizes, normalizes, and splits them into patches before passing them through the encoder to extract features. The text decoder is integrated via a `VisionEncoderDecoderModel`, and captions are tokenized using the corresponding `AutoTokenizer`. GPT-2 generates text autoregressively, enabling caption generation conditioned on visual features.

The model supports selective freezing to reduce trainable parameters. In the ViT encoder, patch embeddings and any number of initial layers can be frozen. Likewise, in the GPT-2 decoder, token (wte) and positional (wpe) embeddings, as well as early transformer blocks, can be optionally frozen. This flexibility helps focus learning on task-specific features.

Training uses the AdamW optimizer with a learning rate of 2e-5. Batch sizes and number of epochs are set via command-line arguments. Early stopping (patience of 5 epochs, threshold of 0.001) is applied to prevent overfitting. A `CustomTrainer` handles training, extending the Hugging Face `Trainer` to compute loss, return model outputs if needed, and apply different learning rates to encoder and decoder.

C. Qwen-2.5 7B Multimodal For caption generation, the `Qwen2_5_VLForConditionalGeneration` class is used alongside its corresponding `AutoProcessor`, which applies a chat template and handles image inputs using custom preprocessing functions. Images are loaded from a predefined dataset and formatted into message structures that include both the image and a short textual prompt.

During inference, messages are tokenized, images are preprocessed, and captions are generated autoregressively with a maximum of 128 tokens. The model outputs a single descriptive phrase per image, which is then cleaned and stored in a CSV file. This setup allows the model to generate visually grounded text without requiring additional fine-tuning, leveraging the instruction-following capabilities of the base model.

D. ViT-Llama 3.2 Model The ViT encoder processes images by dividing them into 16×16 patches and extracting visual features using a 12-layer transformer with a hidden size of 768 and 12 attention heads. For text generation, Llama 3.2 is used in its 1B and 3B parameter variants, both configured as causal language models with 12 layers and a hidden size of 1024.

To integrate both modalities, the visual features extracted by ViT are projected into the Llama embedding space and concatenated with text embeddings, enabling joint processing by the decoder. The model is fine-tuned using LoRA (Low-Rank Adaptation), which introduces low-rank updates to reduce the number of trainable parameters. Specifically, LoRA is applied with rank 4, alpha 16, and dropout 0.1 for the 1B version, and with rank 4, alpha 8, and dropout 0.1 for the 3B version. Training is performed with the AdamW optimizer at a learning rate of 1e-4, over 10 epochs, with a batch size of 4 for training and 2 for validation. The combined ViT and Llama components are implemented in the custom `ViTLlamaForCaptioning` class, designed to handle multimodal input efficiently.

5. Results

5.1. Base Architecture for Caption Generation

We evaluated the multiple combinations of image encoders and text decoders, using different text representations: word-level, word-piece-level, and character-level. Table 1 summarizes the performance of the best representation for each model. The ResNet-18 + Transformer architecture with word-piece representation clearly outperformed all configurations, achieving significantly higher scores in every metric.

Resnet-18 with GRU and LSTM

In the ResNet-18 + GRU setup, word-level representation achieved the lowest loss and fastest convergence, making it the most effective. Word-piece representation had higher loss but generalized well. Character-level representation had the highest loss, likely due to the increased difficulty in capturing long-range dependencies. Similar trends were observed with ResNet-18 + LSTM, with word-level remaining the most efficient.

ResNet-50 with GRU and LSTM

All three text representations in the ResNet-50 + GRU configuration exhibited clear overfitting, which was not observed in the ResNet-18 + GRU setup. This could be because ResNet-50's detailed features increase the model's capacity to memorize training data rather than generalizing to the validation set.

With LSTM, overfitting persisted, especially for character- and word-level representations, but appeared to be less severe. This suggests that LSTM's ability to retain longer-term dependencies helps mitigate overfitting to some extent.

Teacher forcing

To further improve training stability and performance, we applied the Teacher Forcing technique to the best-performing models for each text representation. No noticeable improvements were observed for character- and word-level representations, but a slight performance gain was seen with the word-piece representation across all metrics.

Transformer Decoder Architecture

In experiments using a Transformer decoder with ResNet-18, word-level representation once again achieved the lowest training loss and fastest convergence. However, despite this promising behavior during training, it did not lead to the best performance on the test set. Word-piece-level representation outperformed others across all evaluation metrics, suggesting it offers better generalization in this setup.

Qualitative results

Table 2 shows predicted captions for two example images. Models with GRU and LSTM often generate repetitive or incomplete outputs (e.g., "with with", "chicken with with"), failing to capture full dish names. R50+GRU, despite being the best performing model quantitatively, still shows repetition generating outputs like "grilled grilled with". In contrast, R50+GRU (TF) produces more distinct phrases (e.g., "Chicken with -", "cchini - yogurt"), suggesting benefits from word-piece tokenization and improved decoding. These examples reflect how architectural choices influence caption quality.

Table 1. Performance comparison of models according to the metrics Bleu-1 (B1), Rouge-L (RL) and METEOR (MTR).

Model	text-level	B1	RL	MTR
R18+GRU	word	0.07	0.12	0.05
R18+LSTM	word	0.10	0.13	0.06
R50+GRU	**word**	**0.13**	**0.16**	**0.06**
R50+LSTM	word	0.09	0.11	0.04
R18+TRANSF	word-piece	0.12	0.09	0.06
R50+GRU (TF)	word-piece	0.13	0.11	0.06

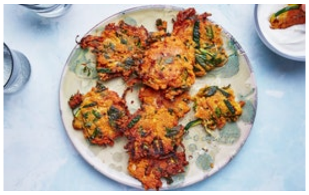
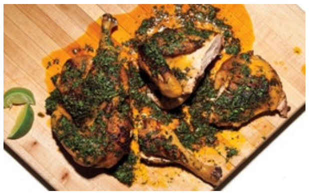
		
Ground truth	**Zucchini-Lentil Fritters With Lemony Yogurt**	**Grilled Chicken with Board Dressing**
R18 + GRU	with with	grilled and
R18 + LSTM	with with	grilled and with
R50 + GRU	grilled grilled with	chicken with with
R50 + LSTM	grilled with	chicken with with
R18 + TRANSF	chicken - chicken chicken chicken	grille - chicken chicken with with with
R50 + GRU (TF)	cchini - yogurt	Chicken with -
ViT ft GPT fz	pucchini andFemoniled-illedter with Carong andurt	ailled Chicken with Greeningressing
ViT fz GPT ft	Cucchini andandimeil Chickenritters Baconong Butterurt	Grilled Chicken with Lemon-ressing and
ViT ft GPT ft	Cucchini-Stimeil Friedritters Baconony Yogurt	Grilled Chicken with Lemon-ressing
Qwen (prompt 3)	Zucchini and Feta Fritters	Herb-Crusted Roasted Chicken with Garlic and Lime
Llama 3.2 1B	Pucchini andemonil Fritters Lemonemony Yogurt	Roilled Chicken with Lemonering
Llama 3.2 3B	Grucchini andentil Saladritters with Yogemony Yogurt Sauce	Grilled Chicken with Lemon Buttering

Table 2. Predicted dish names for two ground truth food images using various model architectures and configurations. Image encoders include ResNet-18 (R18), ResNet-50 (R50), and ViT (Vision Transformer), paired with GRU, LSTM, Transformer decoders, or GPT-based language models. ”ft” and ”fz” indicate whether components were fine-tuned or frozen, and ”TF” refers to teacher forcing during training. Outputs are shown for each configuration, along with baselines from Qwen and LLaMA-3 models.

5.2. ViT + GPT-2 Fine-tuning Strategies

In these experiments, word-piece level representation was used for the textual input. Table 3 summarizes the performance of the experiments with best results for each strategy.

ViT Fine-tuned and GPT-2 Frozen

GPT-2 was kept completely frozen while ViT was gradually unfrozen block by block. Validation loss highlighted two strong models: one with one ViT block frozen (less overfitting) and another with five blocks frozen (lowest validation loss). Considering both loss and evaluation scores, the best balance between generalization and caption quality came from freezing the first five ViT blocks.

ViT Frozen and GPT-2 Fine-tuned

ViT remained frozen while GPT-2 layers were gradually unfrozen. The best performance was achieved by freezing GPT-2 up to block 7, which minimized overfitting and yielded the best overall metrics. Compared to a fully fine-tuned GPT-2, this model showed significantly less overfitting, with only a minor drop in performance (0.01).

ViT Fine-tuned and GPT-2 Fine-tuned

In the final set of experiments, both ViT and GPT-2 were fine-tuned. Two learning rate strategies were tested: using a shared learning rate for both encoder and decoder, and using separate learning rates for each. The latter led to increased overfitting, but yielded better results regarding validation metrics.

Several layer freezing combinations were tried. Denoting as (NViT, NGPT-2) the number of frozen blocks in each component, configurations with low values like (2,2) and (6,6) quickly overfit, whereas (10,10) showed minimal overfitting and a better balance between training and validation loss, achieving also the best metrics.

Across all three strategies, the jointly fine-tuned ViT+GPT-2 model with (10,10) freezing and separate learning rates performed best. This underscores the value of carefully tuning trainable layers, using differentiated learning rates for encoder and decoder components, and prioritizing evaluation metrics over validation loss when selecting the optimal model. Qualitative results in Table 2 further support these findings. Captions generated by the jointly fine-tuned ViT+GPT-2 model are closer to the ground truth, both semantically and syntactically, compared to the other two strategies.

Table 3. Performance comparison of models according to the metrics Bleu-1 (B1), Bleu-2 (B2), Rouge-L (RL) and METEOR (MTR).

ViT	GPT-2	B1	B2	RL	MTR
fine-tune	freeze	0.16	0.06	0.12	0.08
freeze	fine-tune	0.20	0.09	0.18	0.11
fine-tune	**fine-tune**	**0.22**	**0.11**	**0.20**	**0.13**

5.3. Qwen2.5-VL-7B-Instruct

In this section, we evaluate the performance of the Qwen model, when experimenting with three prompting strategies, using word-piece level representation. Table 4 summarizes the impact of the different prompts on the quality of the generated outputs.

Prompt 1 (Baseline): *"Give 1 phrase describing the food in the image from its key ingredients."*

Prompt 2 (Instructional) *"Give a recipe title describing the food in the image. If no title is found give an empty string."*

Prompt 3 (Descriptive + Examples): *"Generate exactly 1 title for the recipe/food/drink based on its key ingredients. Each title should be clear, catchy, and descriptive, highlighting the most important or unique ingredients in the dish. For example, if the key ingredients are potatoes and seasoning, a title might be 'Crispy Salt and Pepper Potatoes'. For a dish with mac and cheese and Thanksgiving flavors, a title could be 'Thanksgiving Mac and Cheese'. You can also include elements that reflect the style or theme of the recipe (e.g., 'Italian Sausage and Bread Stuffing'). Return only the title."*

Prompt 3 outperforms the other prompting strategies, highlighting that the detailed guidance and examples lead to more descriptive and specific outputs with richer and more appealing descriptions that are closer in spirit to the ground truth. Results from Table 2 provide additional insight into the model's capabilities: Qwen predictions show certain ability to capture key ingredients.

Table 4. Performance comparison of prompt strategies on Qwen2.5-VL-7B-Instruct.

Prompt	B1	B2	RL	MTR
Prompt 1	0.139	0.064	0.205	0.177
Prompt 2	0.237	0.125	**0.228**	0.176
Prompt 3	**0.244**	**0.126**	0.221	**0.180**

5.4. Llama 3.2 -1B and -3B + LoRA

All experiments in this section use word-piece level representations. As shown in Table 5, LLaMA 3.2–3B model outperforms its 1B counterpart, though differences are modest. This indicates that, when fine-tuned with LoRA, the smaller 1B model can achieve nearly the same level of performance as the much larger 3B model, which overall yielded the best results among all architectures tested.

Table 5. Comparison between LLaMA 3.2–1B and 3.2–3B models using BLEU, ROUGE-L, and METEOR metrics.

Model	B1	B2	RL	MTR
LLaMA 3.2–1B	0.260	0.134	0.223	0.155
LLaMA 3.2–3B	**0.269**	**0.141**	**0.228**	**0.160**

Table 2 results show that even though both models exhibit some spelling and grammatical issues, Llama 3.2 1B and 3B show a more coherent and accurate labeling of dishes, particularly when compared to earlier architectures. These improvements showcase the models' ability to capture and better reflect the intended dish names, despite some remaining inaccuracies, such as "Grilled Chicken with Lemon Buttering".

6. Conclusions

This study demonstrates that both model architecture and input representation significantly influence the quality of image-to-text generation. Encoder-decoder models like ResNet with GRU, LSTM or transformer-based decoders often produced repetitive or incomplete captions. Fine-tuning large multimodal models such as ViT-GPT2 and LLaMA 3.2—with careful control over layer freezing and learning rates—yielded substantial gains. Additionally, prompt engineering in the Qwen model showed that well-crafted instructions can guide the model toward more descriptive and accurate captions. Overall, results highlight the effectiveness of attention mechanisms, multimodal pretraining, and adaptive tuning strategies in generating high-quality food image descriptions.

References

[1] Lakshita Agarwal and Bindu Verma. From methods to datasets: A survey on image-caption generators. *Multimedia Tools and Applications*, 83:28077–28123, 2024. 2

[2] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014. 3

[3] 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. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2010.11929. 4

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 770–778, 2016. 2, 3

[5] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*, 2022. 4

[6] Johannes Knittel, Tushaar Gangavarapu, Hendrik Strobelt, and Hanspeter Pfister. Gpt-2 through the lens of vector symbolic architectures. In *2nd Workshop on Attributing Model Behavior at Scale (ATTRIB), NeurIPS*, 2024. Workshop paper. 4

[7] Sara Sarto, Marcella Cornia, and Rita Cucchiara. Image captioning evaluation in the age of multimodal llms: Challenges and future perspectives, 2025. 5

[8] Himanshu Sharma and Devanand Padha. A comprehensive survey on image captioning: from handcrafted to deep learning-based techniques, a taxonomy and open research issues. *Artificial Intelligence Review*, 56:1–43, 2023. 1

[9] Himanshu Sharma, Manmohan Agrahari, Sujeet Kumar Singh, Mohd Firoj, and Ravi Kumar Mishra. Image captioning: a comprehensive survey. In *2020 International Conference on Power Electronics & IoT Applications in Renewable Energy and its Control (PARC)*, pages 325–328. IEEE, 2020. 2

[10] Manan Sharma. Food ingredients and recipe dataset with images. https://www.kaggle.com/datasets/pes12017000148/food-ingredients-and-recipe-dataset-with-images, 2020. Accessed: 2025-04-05. 4

[11] Matteo Stefanini, Marcella Cornia, Lorenzo Baraldi, Silvia Cascianelli, Giuseppe Fiameni, and Rita Cucchiara. From show to tell: A survey on deep learning-based image captioning. *arXiv preprint*, 2021. 2

[12] Liming Xu, Quan Tang, Jiancheng Lv, Bochuan Zheng, Xianhua Zeng, and Weisheng Li. Deep image captioning: A review of methods, trends, and future challenges. *Neurocomputing*, 546:126287, 2023. 2