Image Captioning of Recipes

Luis Domene, Eric López, Marino Oliveros
UAB, 17-12-2024

Abstract
Image captioning continues to push the boundaries of visual understanding, but current solutions struggle to interpret domain-specific images and contexts. In this project, we explore the use of Transformer-based Vision Encoder-Decoder models to enhance caption generation for recipe images. By using a dataset dedicated to recipes, we show that these advanced methods can learn to reproduce detailed food descriptions, but still struggle to match them with the actual content of the images.

Keywords: Image captioning, Transformers, Vision Language Models, Vision Encoder Decoder Models, BLIP, Recipes dataset

Contents

1 Introduction

Image captioning is the task of recognising what is going on in an image, from visual input to textual output. This task is evaluated with different metrics compared to other AI tasks that normally use confusion matrices, accuracy or F1 scores. We use other metrics that take into consideration how similar the produced text is to the reference text instead. These metrics (that we will dive further in later on in the paper) include BLEU, ROUGE, METEOR and their derivatives.

Image captioning tasks are usually performed in very universal fields that include all types of action and objects, to break the norm and to advance further in the image captioning field we will focus on a certain field in this paper: food. More specifically we will be dealing with photos of meals and their corresponding ingredients and recipes. We will be using a transformer-based vision encoder decoder architecture to caption these images. We will train a model in this dataset that is able to caption from spaghetti bolognese to nigiri sushi.

Being an image captioning task we will be using methods such as our baseline, ViT-GPT2, based on a CLIP-ViT approach introduced in [Anderson et al., 2018] chosen for its modularity and strong pertaining foundation as it provides a solid point of comparison for exploring more advanced architectures. And other proposals such as DINO-GPT2 and BLIP (Bootstrapped Language-Image Pretraining) for its unified architecture and the ability to understand and generate contextually rich and detailed captions. Our models will be fine tuned to assure the best possible results. We will compare and discuss the results and configuration of our models and additionally explain the struggle that these models have to match the accurately generated food descriptions with the actual content of the images in this dataset and scenario. The workflow we will follow in this project is depicted in Figure 1.

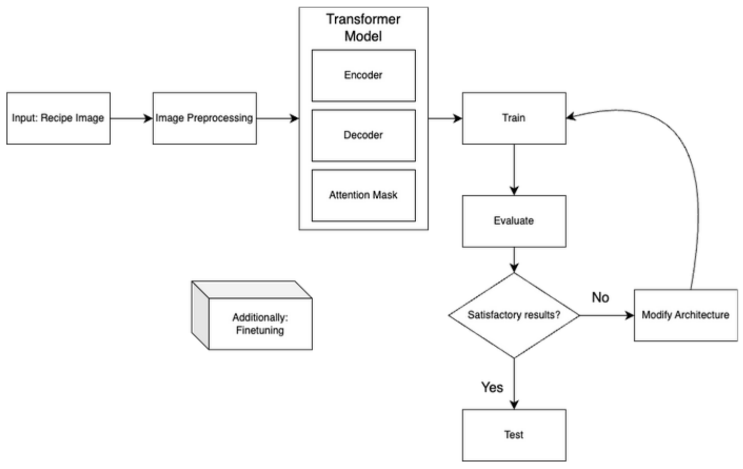


Figure 1: Workflow Flowchart

2 Related Work

The current State of the Art (SoTA) for image captioning taks begins with an exploration of advancements in computer vision and natural language processing. Over the year, numerous approaches have been developed to tackle this problem from traditional methods to state of the art transformer-based models.

Early methods for image captioning relied on template-based techniques, where predefined sentence structures were filled with visual content identified in images. These methods lacked flexibility and struggled with complex images. With deep learning, 'encoder-decoder' frameworks became the new rule. A notable example is the Neural Image Captioning (NIC) model, which marked a significant step forward.

The introduction of attention mechanisms greatly improved image captioning by allowing models to focus on specific regions of an image relevant to the current word being generated. Key advancements include:

- Bottom-Up and Top-Down Attention [Anderson et al., 2018]
- Attention on Attention (AoA) Networks [Anderson et al., 2018]
- X-Linear Attention Networks (X-LAN) [Pan et al., 2020]

Transformer based approaches. The shift from RNNs to transformers revolutionized image captioning by enabling better global context modeling and parallelization. The current Transformer-based methods that dominate the current state of the art would be:

- VisualGPT [Chen et al., 2022]
- PromptCap [Y. Hu et al., 2023]
- In-Context Learning [Qin et al., 2024]

In-Context Learning uses few-shot learning capabilities to demonstrate how in-context examples can guide the model to generate more accurate and context-aware captions.

These SoTA models are widely used and commonly evaluated on datasets such as MS-COCO, Flickr30k and Conceptual Captions. Evaluated on metrics such as BLEU, METEOR and ROUGE.

In this project we will use the aforementioned transformer-based approach, explained thoroughly in the methodology section.

3 Methodology

In order to achieve a model that could provide accurate captions of the recipes images, which we observed were very specific in terms of image content and their corresponding caption, we thought of starting with a baseline architecture, and gradually customize it by replacing its parts, adding new ones or changing the parameters.

The dataset used was already preprocessed and in a tabular form, so the only cleaning required was to remove the samples that contained empty captions or empty image paths. We divided the data into train, validation and test splits of 70-10-20%, respectively, and for each new customization, we trained the model on the train set and evaluated rigorously on the validation set. This allowed us to have an idea of the improvement each change had on our baseline.

The following sections explain in detail the baseline architecture used, and the iterative modifications

3.1 Baseline

ViT-GPT2

The baseline model follows the architecture of a Vision Language model, similar to PaliGemma [Beyer et al., 2024], but uses a ViT encoder and GPT-2 decoder, like in [Luo et al., 2022]. It is not trivial to join a pretrained image encoder and a large language model (LLM), since they have been trained in different domains and tasks, but now have to work together in the same task. A simple approach to achieve that is to project the outputs from the encoder, which are in the image space, to the language space, so the LLM can understand them. Therefore, the strategy employed is passing the visual tokens to a linear projection, concatenating them with the text tokens and passing them to GPT-2 as input embeddings.

The text input was the first words from the caption; the ground truth ones during training (teacher forcing), and the generated so far during inference. Also, we encapsulate the ground truth caption between beginning <bos>and end of sequence <eos>tokens, and at inference time, the text input starts being only a <bos>token. The image encoder used was the pretrained Google's vit-base-patch16-224 [Dosovitskiy et al., 2021]. We believed this was a good option to provide basic understanding of the images, so we initially kept it frozen, and only trained the linear projection and fine tuned GPT-2.

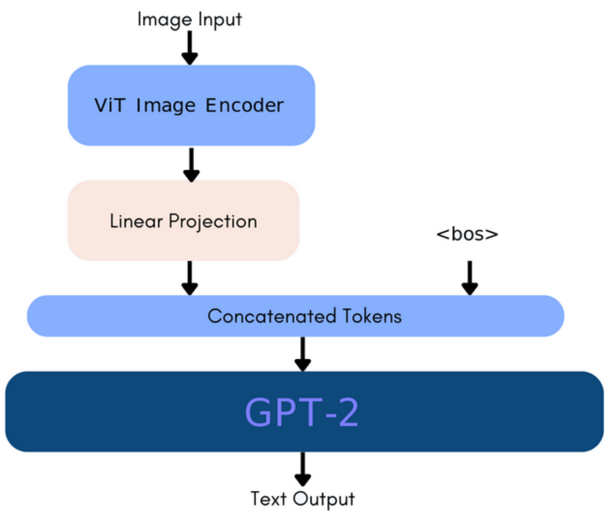


Figure 2: Baseline model architecture, a vision language model similar to PaliGemma

The image encoder, a ViT, works in the following way: the image is splitted into a sequence of square patches, embeds them with a linear projection, similar to text tokens, but concatenates them with a CLS token, and then applies standard transformer encoder layers. In Figure 3 bottom we can see the architecture of a ViT. The resulting embeddings from the CLS token are then passed to the decoder, GPT-2.

This decoder-only transformer usually takes text as input, but we can replace text embeddings with the projected output embeddings from the image encoder. Then, the decoder will predict the first text token of the caption. For the next ones, the predicted token embeddings so far will be concatenated to the image embeddings.

3.2 Modifications

DINO-GPT2

The baseline model did not yield satisfactory results on the dataset, and we thought that could be because of the image encoder not being powerful enough, so we decided to change it to Meta's dinov2-small [Oquab et al., 2024]. This model had an unsupervised training and can capture robust features from images. We also increased the linear projection to 2 layers with a ReLU activation function, as we believed the projection could be also the bottleneck of the system.

Additionally we experimented with some minor changes, like finetunning DINO, adding contrastive loss, etc. but these are explained in more detail in the experiments section.

DINO-SmolLM

After changing the image encoder, we decided to also change the language model with a more modern one, HuggingFace's SmolLM2-135M [Allal et al., 2024]. This is proved to demonstrate significant advances in reasoning and knowledge over previous LLMs. However, we discovered the bottleneck was not in the language model we used, as the performance didn't improve.

DINO-GPT2-VED

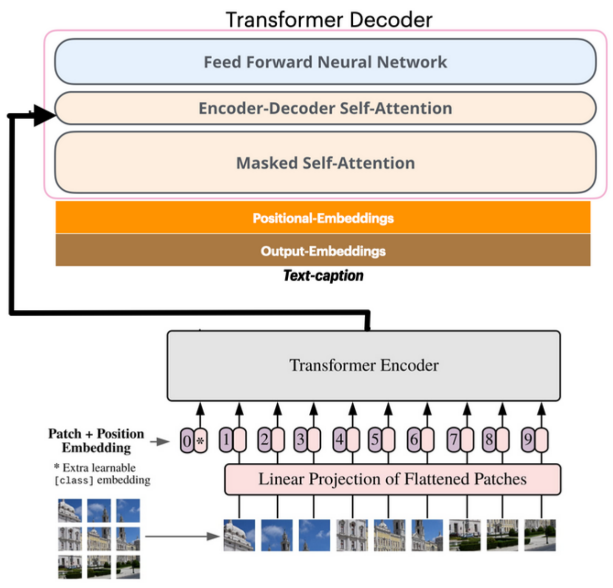

Transformer Decoder
Feed Forward Neural Network
Encoder-Decoder Self-Attention
Masked Self-Attention
Positional-Embeddings
Output-Embeddings
Text-caption
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
Linear Projection of Flattened Patches

Figure 3: Architecture of a Vision Encoder Decoder model, using a ViT as an encoder

At this point, the results obtained were far from good. As we will see later in the results, the generated text was not related at all with the image. We needed a change of paradigm, so we decided to change the way we join the encoder and the decoder together: we implemented a Vision Encoder Decoder (VED) model. The main difference between a VED and the previous approach lies in the way the encoder passes its embeddings to the decoder: cross-attention layers are introduced into the decoder architecture to receive the encoder output. These then need to be trained, usually together with the fine tuning of the encoder and the decoder.

We maintained DINO as the encoder, and switched the LLM back to GPT-2 since SmolLM didn't yield improvements.

ViT-GPT2-VED

Until now, we had been used encoders and decoders pretrained separately, and fine tuning them together. This time, we used a VED model employing a ViT and GPT2 fine tuned on image captioning, NLPConnect's vit-gpt2-image-captioning [nlpconnect, 2024]. However, it was not enough, since the recipes dataset is very specific, so we further fine tuned it on our data. Also, we tried to reinitialize the weights of the encoder to train it from scratch, and changed some hyperparameters of the architecture to attempt better scores. For the first, we used Xavier uniform initialization.

BLIP

Given that the task of image captioning involves the understanding of the image but also the generation of text, the approach followed by BLIP (Bootstrapping Language-Image Pre-training) [Li et al., 2022] seems to be more accurate than the previous approaches in our case. From the full architecture of BLIP, shown in Figure 4, in our case that we want to do image captioning, we will only use the part seen in Figure 5. BLIP is a vision-language model designed to handle both understanding and generation tasks for image-text data. To address the challenge of noisy web data, BLIP introduces the Captioning and Filtering (CapFilt) method during his pre-training, which includes:

- A captioner that generates synthetic captions for web images to enhance training data quality.
- A filter that removes noisy or irrelevant captions, ensuring the model learns from accurate image-text pairs.

BLIP pre-trains on both vision-language understanding tasks (like image-text retrieval) and generation tasks (like image-captioning). The petraining model use three different losses for the training:

- Image-Text Contrastive Learning (ITC): Aligns image and text representations in a shared space to distinguish between matching and non-matching pairs.

- Image-Text Matching (ITM): Trains the model to identify whether a given image and text pair is semantically aligned.
- Language Modeling (LM): Enables the model to generate coherent text based on visual inputs, essential for tasks like image captioning.

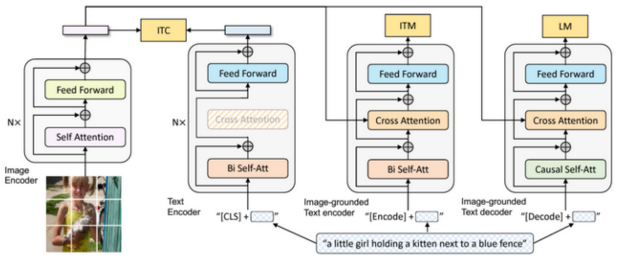


Figure 4: Architecture of BLIP

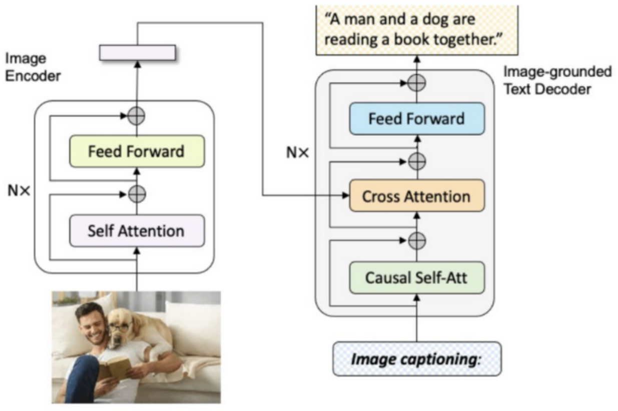


Figure 5: Blip Image Captioning

Given that the second version of BLIP also existed we look at it and try to compare the differences between both in our task. BLIP-2 improves upon BLIP by incorporating a lightweight query mechanism to align the vision encoder and language decoder more efficiently. This reduces computational cost while enhancing performance. The drawback we've found through the implementation of BLIP-2 is that probably due to the lack of a significant amount of data the training of BLIP-2, though promising, didn't give us the expected results being surpassed by BLIP-1.

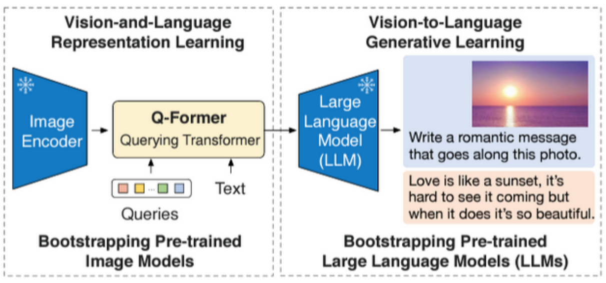

Vision-and-Language Representation Learning
Vision-to-Language Generative Learning
Image Encoder
Q-Former
Querying Transformer
Queries
Text
Large Language Model (LLM)
Write a romantic message that goes along this photo.
Love is like a sunset, it's hard to see it coming but when it does it's so beautiful.
Bootstrapping Pre-trained Image Models
Bootstrapping Pre-trained Large Language Models (LLMs)

Figure 6: Blip2 Image Captioning

LoRA (Low-Rank Adaptation) [E. J. Hu et al., 2021] and PEFT (Parameter-Efficient Fine-Tuning) [Xu et al., 2023] are techniques designed to optimize the fine-tuning of large pre-trained models. LoRA works by injecting low-rank matrix updates into the weights of a frozen pre-trained model, allowing fine-tuning to focus on a small subset of parameters. This approach drastically reduces memory and computational requirements while maintaining performance, making it ideal for adapting models like BLIP and BLIP-2.

In BLIP from the 252.869.948 parameters, 5.455.872 were trained which means that we have fine tuned the model and get better results compared to zero-shot test just using 2.1% of the parameters which is a huge saving in time and resources.

In BLIP-2 the number of parameters is much bigger because of the LLM 3B parameters and the training was done in 5,242,880 parameters which is just 0.1%. In this case the fine tuned model works better than zero shot but cannot be considered as a good result. Perhaps in this case the amount of learnable parameters could have affected the low performance of the model.

4 Experimental Design

4.1 Dataset Statistical Analysis

Our dataset, the 'Food Ingredients and Recipes Dataset with Images', was obtained from Kaggle [Goel et al., 2020].

We conducted a statistical analysis on it to obtain some more information of what we were working with, for this we explored our data obtaining the following relevant information.

Figure 7: Dataset sample image '3 ingredient cacio e pepe pasta with cheese and pepper'

The dataset is divided in an image folder and a .csv file containing the recipes, captions and image ids that include the name of the images. There are a total of 13,582 images with 13,306 captions.

Image width	Min	274
	Mean	274.031
	Max	702
Image height	Min	169
	Mean	169.041
	Max	722
Title length	Min	3
	Mean	32.761
	Max	112
Lexicon	Common words in Titles	with, and, Salad, Chicken, Sauce, Grilled...
	Common words in Ingredients	1, Cup, Teaspoon, 2, Tablespoons, fresh. . .
	Rare words in Titles	Board, Thread, Sinigang, Bihon, Eureka. . .
	Rare words in Ingredients	Nougat, cup/95, Gluten-free, oolong. . .

Table 1: Overview of the dataset statistics

The mean image size is 169x274 pixels, as we can see there are some clear outliers with the maximum image values being 722 and 702 pixels for height and width respectively. The mean title length for a caption would be 32.76 characters with 3 as minimum and 112 at maximum.

In the lexicon part of the table we have a summary of the most common words from the Title/Caption section of the csv file, these make sense as are connector words and names of very common plates of food. Same with ingredients –proportions. On the rarer words we chose words that are only repeated once, things like 'Bihon' in titles or 'Nougat' in ingredients, we did not include typos.

Dataset anomalies and dealing with them: unmatched images, duplicate captions and missing captions: As aforementioned there are 13,582 images with 13,306 captions which creates an imbalance in our dataset. Also there are only 13,472 unique image names. We conducted additional analysis to conclude what could be wrong with our dataset and how to fix it. This is all available in our Github repo under the name of Statistical Analysis.

Missing images
We found 30 missing images, these are images that are listed in the 'Image_Name' column of the dataset but not found in the image folder. The cause for this could be that either the files were deleted or never added, the impact on our dataset is the reduction of its size, which being only 30 images is something that won't affect performance as much. If the recovery of these images is not possible we will exclude the rows from the csv file.

Unmatched images
111 unmatched images were identified, there are extra images files found in the folder but not referenced in the dataset (no corresponding captions). These images are not usable as they lack captions and can lead to some confusion during data preprocessing. To solve this issue we can either remove or label these images; we settled for the latter choice.

Duplicate captions
Some captions are associated with multiple images (different angles or versions of the same dish), this is not a problem but it needs to be handled correctly to avoid biases in training. We will retain the duplicate captions. We found 162 instances of this occurrence.

Missing captions
Only 5 missing captions were found, they have 'NaN' or empty values in the 'Title' column, as there are only 5 we will exclude these rows from the total dataset.

We hope that by addressing these issues, our dataset will be ready for training in a successful, unbiased manner.

4.2 Metrics

The metrics used to asses our model's performance are employed because they evaluate natural language generation in a way that allows to objectively measure how well the generated caption align with the human-written reference caption (recipes from the dataset), they are as follow:

BLEU (Bilingual Evaluation Understudy) Scores
BLEU is a precision-based metric that evaluates how closely a generated caption matches the reference captions. BLEU calculates n-gram overlap between the predicted captions and reference captions, with BLEU-1, BLEU-2, BLEU-3, and BLEU-4 referring to the overlap of unigrams, bigrams, trigrams, and four-grams, respectively. In our evaluation, we focus on BLEU-1 and BLEU-2:

- BLEU-1: Measures unigram precision, basically the percentage of individual words in the predicted caption that appear in the reference captions.
- Measures bigram precision, the percentage of consecutive word pairs (bigrams) in the generated/predicted caption that appear in the reference captions.

These scores provide an indication of the lexical similarity between the generated captions and human references. High BLEU scores suggest that the model generates captions that closely follow the wording and phrase structure of the reference captions.

METEOR (Metric for Evaluation of Translation with Explicit ORdering)
METEOR is a more comprehensive metric that tries to improve upon BLEU by considering things like synonymy, stemming, and word order. It computes a weighted average of precision and recall, with a greater emphasis on recall, encouraging models to generate diverse, relevant content. METEOR also incorporates a penalty for word order mismatches, making it more sensitive to the structure of the caption. The score ranges from 0 to 1, with 1 indicating a perfect match to the reference captions.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation)
ROUGE is a family of metrics used primarily for the evaluation of summary generation tasks but is also applicable to image captioning. The ROUGE score focuses on recall, measuring the overlap of n-grams, word sequences, or word pairs between the predicted and reference captions.

ROUGE is particularly useful in assessing the fluency and syntactic integrity of generated captions, as it rewards models that produce captions with a coherent sequence of words.

4.3 Split & Training Configuration

As previously mentioned, the dataset is splitted into train, validation and test sets. We train the model and different experimentations in the train set, and the goal is to obtain the highest metrics in the test set, using the model configuration that resulted in highest metrics in the validation set.

Internally, the training is done with classic teacher forcing, where the model is given the first tokens of the caption and needs to predict the next token. The logits are compared with the one-hot vector of vocab. size with 1 in the ground truth token, and the objective is to minimize the cross-entropy loss. Additionally, we experimented with the combination by addition of this loss and a contrastive one, where the image and the caption's text latents are aimed to get together, using a simple matrix multiplication of both latents, and later a CLIP-like contrastive loss.

Intensive experimentation was done with hyperparameters. The most relevant are shown in Table 2: the contrastive loss weight, the inference mode (greedy, sampling or beam search), the choice of using pretrained weights of the encoder and decoder, and whether to finetune them for both. Other parameters for inference were tuned, but didn't affect the result much. These and their final values were: maximum new tokens (20), the repetition penalty (2.0), length penalty (3.0), top k (50) and top probability (0.95) for sampling inference, and number of beams (5) for beam search. The experimentation in hyperparameters was mostly done on the first modifications of the model architecture; once we found stable values, these were kept for the last models (BLIP).

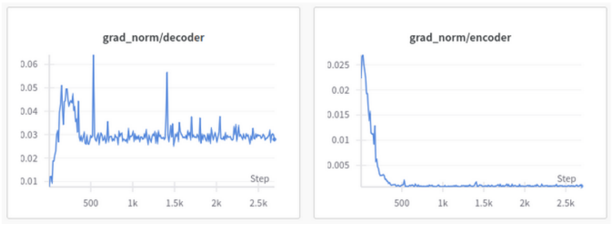


Figure 8: Gradient norm monitoring in the finetunning of the decoder and the encoder for DINO-GPT2

The continuous monitoring of the model training was crucial to know where to apply the modification. We kept track of the training and validation total loss, but also of cross-entropy and contrastive loss, and of the gradient norm of the encoder, projection and decoder separately. For instance, Figure 8 shows that the gradients for the encoder rapidly became negligible, meaning the encoder was not learning properly, that's where we decided to use the VED approach.

We also monitored the cosine similarity between text and image latents when contrastive loss was used, in order to ensure the loss was strong enough or we had to increase its weight. Moreover, keeping track of the distinct k-grams (with k=1,2) in the validation generations was also useful to know that the captions were each time less random.

Finally, finetunning with LoRA, previously explained, was also an experiment we tried. Finetunnings were made in one Nvidia Titan X GPU with 12 VRAM, except for BLIP2, which we required to parallelize in five of these GPUs, since it was a bigger model.

For the reported metrics in the results section, we evaluated the epoch checkpoint that resulted in highest metrics in average in the validation set.

5 Results

Model Name	**Use pre-trained Encoder**	**Use pre-trained Decoder**	**Train Encoder**	**Train Decoder**	**Contrastive Loss**	**Inference mode**	**BLEU-1**	**BLEU-2**	**ROUGE-L**	**METEOR**
ViT-GPT2	✓	✓		✓		Greedy	0.075	0.009	0.054	0.083
DINO-GPT2	✓	✓		✓		Greedy	0.058	0.012	0.054	0.070
	✓	✓	✓	✓		Greedy	0.076	0.010	0.031	0.055
	✓	✓	✓	✓	x0.1	Greedy	0.078	0.015	0.062	0.089
	✓	✓	✓	✓	x0.1	Sampling	0.049	0.005	0.054	0.063
	✓	✓	✓	✓	x1	Sampling	0.053	0.000	0.043	0.069
	✓	✓	✓	✓	x10	Sampling	0.042	0.008	0.038	0.057
	✓	✓	✓	✓	x0.1	Beam search	0.096	0.009	0.063	0.081
DINO-SmolLM	✓	✓	✓	✓	x0.1	Beam search	0.003	0.000	0.008	0.014
DINO-GPT2-VED	✓	✓	✓	✓	x0.1	Sampling	0.058	0.000	0.081	0.062
ViT-GPT2-VED	✓	✓	✓	✓	x10	Beam search	0.074	0.016	0.107	0.080
	✓	✓	✓	✓	x10 (clip)*	Beam search	0.089	0.020	0.109	0.084
			✓	✓	x10 (clip)*	Beam search	0.049	0.006	0.064	0.054
		✓	✓	✓	x10 (clip)*	Beam search	0.096	0.018	0.105	0.069
BLIP	✓	✓			x1	Sampling	0.117	0.040	0.134	0.107
	✓	✓	✓	✓	x1	Sampling	**0.262**	**0.132**	**0.236**	**0.179**
BLIP2	✓	✓			x1	Sampling	0.078	0.003	0.185	0.175
	✓	✓	✓	✓	x1	Sampling	0.074	0.036	0.131	0.149

Table 2: Table of the results in the Recipes dataset, showing the baseline model and all the modifications and ablation studies, and the final model family, BLIP. Checks indicate whether for the model a pretrained encoder/decoder was loaded and finetunned.
*CLIP-like contrastive loss

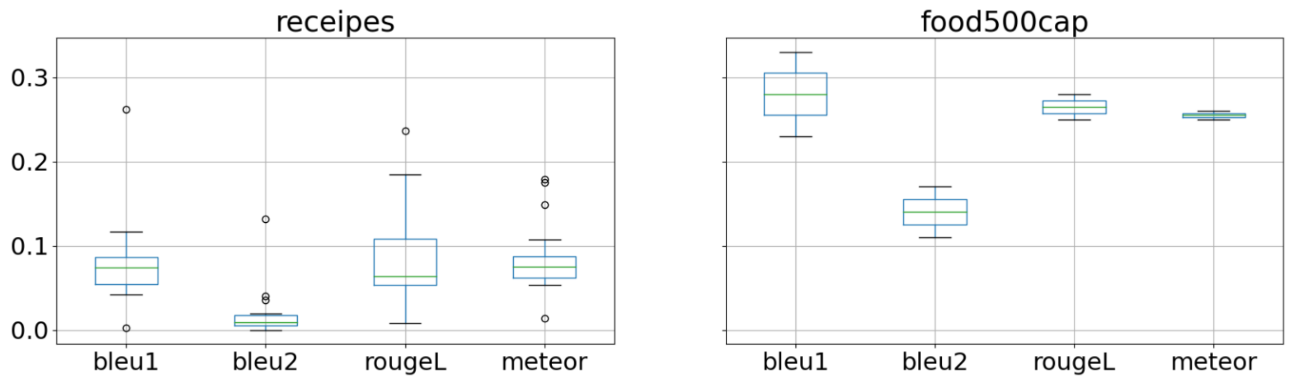


Figure 9: Distribution of metrics for all experiments done in the target dataset (recipes) and the alternative dataset (food500cap)

GT: Dungeness Crab and Heirloom Bean Brandade
Greedy: Chilled-Cheddar-Roasted with Cheese-Salted
Sampling: Earle Pie with Bambouette Sauce Zizzo and Brittle and Crumble Tangerines Save Recipe
Beam search: Gluten-Free Pancetta With Soy Sauce Braised Clams, Greens and Goat Cheese Ra

GT: Chocolate Marble Cheesecake
Greedy: Chilled Chich-Cheddar Cheese
Sampling: Severines (Sea Salt Beer Pizza) Fresh Squash and Poblano's Crisp Tacos Tikka-Tsak
Beam search: Spring Bread Crumb Fattousal-Saccéry (nascotata)

Figure 10: DINO-GPT Example captions with different inference modes

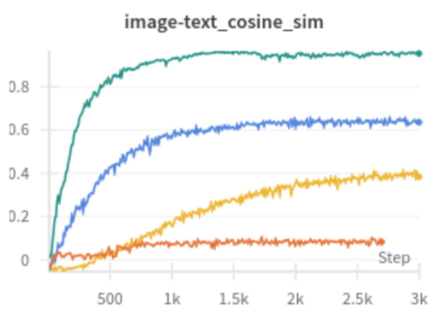
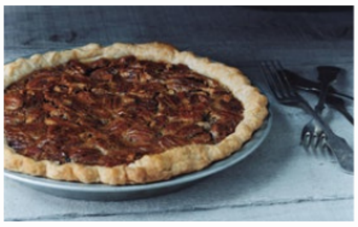


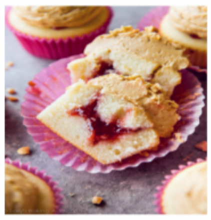
Figure 11: Cosine similarity of image and text latents, with x10 contrastive loss weight, of DINO-GPT (orange), ViT-GPT-VED (blue), ViT-GPT-VED with LoRA and 16 patch size (Yellow) and ViT-GPT-VED without LoRA but 32 patch size (green)



VIT-GPT2-VED: "Grilled Chicken Thighs with Fennel, Peas, and Cucumber Salad ("
BLIP: "pecan - caramel tart"

GT: "Bittersweet Chocolate Pecan Pie"

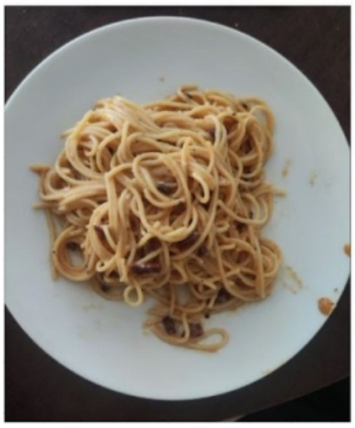
Figure 12: Example comparison BLIP vs previous modification



BLIP: "a muffin with a bite of strawberry jam on top of it, and a few muffins on the bottom"

GT: "red juicy strawberry jam was stuffed into baked golden cakes."

Figure 13: Example caption on a Food500Cap sample



BLIP: Pasta with bacon and parmesan
BLIP2: Chili Noodles with Sp (Noodles)(Noodles)
BLIP2+Prompt: saucepapollo, a traditional italian sauce, with dried tomatoes, onions and spices

Figure 14: Real Life Example

6 Discussion

Results in Table 2 show that most of the modifications were not successful at increasing the performance of the image captioning model, although BLIP was the approach that increased metrics the most. The baseline model, VIT-GPT2, only achieved 0.075 BLEU-1, and 0.083 METEOR. Changing the encoder to DINO and finetunning it didn't improve.

Greedy inference mode seems to inflate the metrics, but that's because it learns the most common words and always generates them, Figure 10 shows two examples. Sampling and Beam search instead add more variety, there seems to be no significant improvement in metrics by choosing one or the other, but Sampling is faster and more memory efficient, that's why we used it for the later models.

As explained in previous sections, our aim with contrastive loss was to get together image and text representations, and this is evaluated with the cosine similarity, as shown in Figure 11. While it's harder to achieve that with the first Vision Language models, even when raising the contrastive loss weight to 10, it is easier with VED architectures. Another observation is that LoRA finetunning can't achieve this aim as well as traditional finetunning, and increasing the image patch size for the ViT encoder highly increases the similarity. Changing the contrastive loss to the one used by CLIP did not bring much difference. However, this was not enough to raise the metrics significantly, suggesting that bringing together the latents of different modalities was not the main problem.

Finetunning a pretrained decoder and training the image encoder from scratch was the best option for the VIT-GPT2-VED approach. Regarding BLIP and BLIP-2, the following table summarizes them:

	Bleu-1	Bleu-2	Rouge-L	Meteor
BLIP	0.117	0.040	0.134	0.107
BLIP-FINE TUNED	**0.262**	**0.132**	**0.236**	**0.179**
BLIP-2	0.078	0.003	0.185	0.175
BLIP-2-FINETUNED	0.074	0.036	0.131	0.149

The evaluation metrics BLEU-1, BLEU-2, ROUGE-L, and METEOR provide insight into the performance of image captioning models. Some examples are shown in Figures 12, 13, 14. Comparing the results, BLIP FINE TUNED achieves the highest scores across all metrics, showing significant improvements over its base model, BLIP, particularly in BLEU-1 (0.262 vs. 0.117) and METEOR (0.179 vs. 0.107). On the other hand, BLIP-2 exhibits lower scores overall, with BLEU-1 and BLEU-2 showing minimal performance (0.078 and 0.003, respectively). However, fine-tuning BLIP-2 slightly improves METEOR (from 0.175 to 0.149), though BLEU-1 and BLEU-2 scores remain comparable to the base BLIP-2. The amount of data to train BLIP-2 seems to not be enough and the LLM tend to hallucinate a lot of words and make the caption excessively long that lead to the poor performance observed in BLEU metrics. Seeing this problem we tried to guide the model through the use of a small prompt that was "The ingredients of the food are" this way the result seems to improve a little, being more specific, but they weren't substantial improvements. This can be seen in Figure 14.

In general, results shown by BLIP are the best performant. The generated captions have a satisfactory quality and detail at describing the content in the images, but this is not reflected in the metrics, since the dataset ground truth is rather usually the technical name of the dish or recipe. Instead, it was easier for all the models to fit the Food500Cap dataset. As we can see in Figure 9, the overall metrics distribution of models in Food500Cap was much higher than in Receipes. Figure 13 also supports this observation.

7 Limitations

The results obtained from our experiments highlight several limitations and challenges encountered during the project.

Architecture Changes and Results
Despite multiple modifications to the baseline architecture, including replacing the image encoder with DINO and the language model with SmolLM, significant improvements were not observed.

Component Integration
Integrating a vision encoder and a pretrained lan-

guage model does not automatically ensure seamless collaboration. Misalignment in latent representations between the encoder and the decoder can result in suboptimal captions, as the decoder struggles to interpret visual embeddings.

ViT-GPT2 Bottlenecks
The baseline ViT-GPT2 architecture struggled to adapt to the complexity of the recipes dataset. Attempts to join the vision encoder with the GPT2 decoder were not fully optimized, leaving room for further refinement.

Dataset Complexity
The 'Recipes' dataset presented unique challenges due to its technical nature. Captions often consist of dish names rather than descriptive sentences, which limits the ability of models to generalize and generate detailed captions.

Performance of Large Models
Models like BLIP and BLIP-2 significantly outperformed our custom configurations. This can be attributed to their robust pretraining on diverse and large-scale datasets, enabling strong zero-shot performance. Also we had way too many parameters to finetune BLIP2 to a good standard.

Metrics Limitations
While metrics such as BLEU and METEOR were used to evaluate the models, they may not fully capture qualitative improvements or subtle semantic alignments between generated and ground truth captions. Alternative metrics or human evaluations might better reflect performance for niche datasets like ours. BLEU could have been tried at a word level rather than a character one.

8 Conclusions

In this project, we explored multiple transformer-based architectures for image captioning, focusing on the 'Recipes' dataset, which presented unique challenges due to its specificity and lack of descriptive captions. While our modifications, including the use of different vision encoders and language models, aimed to improve results, we found that integrating pretrained components like ViT and GPT-2 was less effective than anticipated. The misalignment of latent representations between these modules proved to be a bottleneck for performance. In contrast, models like BLIP and BLIP-2 achieved significantly better results, leveraging large-scale pretraining and unified architectures that excel in zero-shot scenarios and especially in when those models are fine tuned. Additionally, while metrics such as BLEU-1 and BLEU-2 provided quantitative insights, they often failed to capture qualitative improvements or subtle semantic nuances in the generated captions. Future efforts should focus on developing tailored solutions, such as custom tokenizers and prompt engineering, as well as incorporating qualitative evaluations to better assess caption quality and alignment with ground truth.

Future Work: To address the limitations and improve upon the current outcomes, we propose the following future directions:

1. **Custom Tokenizer:** Develop and train a custom tokenizer on the 'Recipes' dataset to better capture the specific vocabulary and semantics associated with food names and ingredients, and less on other irrelevant vocabulary not appearing in the dataset.

2. **Prompt Engineering:** Further investigate prompt engineering techniques, particularly for models like BLIP-2, to improve performance and better align generated captions with the dataset. This could range from a simple prior question to guide the answer, to few-shot learning where we provide similar examples before letting the model produce the actual caption.

3. **Enhanced Integration:** Focus on optimizing the collaboration between vision encoders and language decoders. Techniques such as shared latent space training could mitigate the integration challenges.

4. **Human Evaluation:** Incorporate qualitative assessments through human evaluation to better gauge the quality and accuracy of generated captions beyond current metrics.

By addressing these areas, we can build on the foundations established in this project and improve the performance of image captioning models made for domain-specific datasets.

References

Allal, Loubna Ben et al. (2024). *SmolLM2 - with great data, comes great performance.*

Anderson, Peter et al. (2018). "Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6077–6086. DOI: 10.1109/CVPR.2018.00636.

Beyer, Lucas et al. (2024). *PaliGemma: A versatile 3B VLM for transfer.* arXiv: 2407.07726 [cs.CV]. URL: https://arxiv.org/abs/2407.07726.

Chen, Jun et al. (2022). *VisualGPT: Data-efficient Adaptation of Pretrained Language Models for Image Captioning.* arXiv: 2102.10407 [cs.CV]. URL: https://arxiv.org/abs/2102.10407.

Dosovitskiy, Alexey et al. (2021). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.* arXiv: 2010.11929 [cs.CV]. URL: https://arxiv.org/abs/2010.11929.

Goel, Sakshi et al. (2020). *Food Ingredients and Recipes Dataset with Images.*

Hu, Edward J. et al. (2021). *LoRA: Low-Rank Adaptation of Large Language Models.* arXiv: 2106.09685 [cs.CL]. URL: https://arxiv.org/abs/2106.09685.

Hu, Yushi et al. (2023). *PromptCap: Prompt-Guided Task-Aware Image Captioning.* arXiv: 2211.09699 [cs.CV]. URL: https://arxiv.org/abs/2211.09699.

Li, Junnan et al. (2022). *BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation.* arXiv: 2201.12086 [cs.CV]. URL: https://arxiv.org/abs/2201.12086.

Luo, Ziyang et al. (2022). *A Frustratingly Simple Approach for End-to-End Image Captioning.* arXiv: 2201.12723 [cs.CV]. URL: https://arxiv.org/abs/2201.12723.

nlpconnect (2024). *Vit GPT2 Image Captioning.*

Oquab, Maxime et al. (2024). *DINOv2: Learning Robust Visual Features without Supervision.* arXiv: 2304.07193 [cs.CV]. URL: https://arxiv.org/abs/2304.07193.

Pan, Yingwei et al. (2020). "X-Linear Attention Networks for Image Captioning". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10968–10977. DOI: 10.1109/CVPR42600.2020.01098.

Qin, Chengwei et al. (2024). *In-Context Learning with Iterative Demonstration Selection.* arXiv: 2310.09881 [cs.CL]. URL: https://arxiv.org/abs/2310.09881.

Xu, Lingling et al. (2023). *Parameter-Efficient Fine-Tuning Methods for Pretrained Language Models: A Critical Review and Assessment.* arXiv: 2312.12148 [cs.CL]. URL: https://arxiv.org/abs/2312.12148.