Learning joint Embeddings of Cooking Recipes and Food Images: A Contrastive Learning Approach

Antonin Wattel
December 7, 2022

Abstract

In this project, we aim to perform image to recipe retrieval tasks using computer vision and natural language processing techniques. We propose a model combining texts of recipes titles, ingredients and instructions, and their associated food image. This allows us to retrieve a recipe from a list of known recipes given an image query, and in reverse, retrieve an image from a list of known images given a text recipe query. The model consists of an image encoder and a text encoder, trained together using contrastive loss functions to learn joint embeddings of both modalities.

1. Introduction

Food is an important part of our lives. Imagine an AI agent that can look at a dish and recognize ingredients and reliably reconstruct the exact recipe of the dish, or another agent that can read, interpret and execute a cooking recipe to produce our favorite meal. The Computer Vision community has long studied image-level food classification [2, 6], and only recently focused on understanding the mapping between recipes and images using multi-modal representations [10, 14, 16, 17, 22].

The goal of this project is to retrieve a recipe (from a list of known recipes) given an image query and, in reverse, to retrieve an image (from a list of known images) given a text recipe. For this, we propose a simple model consisting of an image encoder and Text encoder, using standard architectures borrowed from the Computer Vision and Natural Language Processing Fields. The model will be be trained in a contrastive manner to learn a joint embedding of text recipes and food images. Both encoders output vectors in a common feature space, where positive pairs are close to each other. This allows for the retrieval task we are aiming at. We experiment on the effect of different image encoders, and the use of different text information such as title, instructions and ingredients.

This project was realized in the context of the Computer Vision course (INF634) by Vicky Kalogeiton and Robin Courant, given at Ecole Polytechnique . The code for this project is available at `https://github.com/antonin-wattel/Contrastive-Food`.

1.1. Related Work

Cross-modal recipe retrieval is a prevalent task in the field of food computing. The task of earning joint representations for textual and visual modalities in the context of food images and cooking recipes has been studied extenively. As stated in [16], recently, works focusing on this task [3, 4, 17, 20, 22] use the idea of learning embeddings for recipes and images, and projecting them into a joint embedding space that is optimised using contrastive or triplet loss functions. Other advances introduce new complex models and loss functions, , such as cross-modal attention [11], adversarial networks [20, 22], the use of auxiliary semantic losses [3, 17, 20, 22], multi-stage training [9, 17], and reconstruction losses [11].

2. Learning Image-Recipe Embeddings

2.1. General approach

We adopt a model architecture similar to that of CLIP [15]. The idea is to jointly train an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. Given a batch of N (image, text) pairs, CLIP is trained to predict which of the $N \times N$ possible (image, text) pairings across a batch actually occurred. To do this, CLIP learns a multi-modal embedding space by jointly training an image encoder and text encoder to maximize the cosine similarity of the image and text embeddings of the N real pairs in the batch while minimizing the cosine similarity of the embeddings of the $N^2 - N$ incorrect pairings. In order to perform retreival tasks, we simply need to encode the query and database examples, compute their similarities, and choose the closest point in the embedding space. We illustrate this approach in Figures 1 and 6.

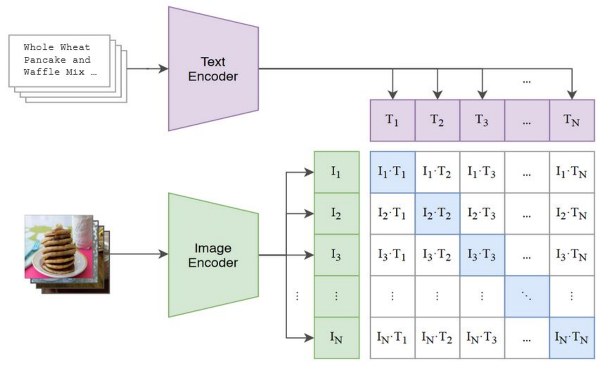

Whole Wheat Pancake and Waffle Mix ...
Text Encoder
Image Encoder
T_1 T_2 T_3 ... T_N
I_1 I_2 I_3 I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$

Figure 1. Summary of our approach - Contrastive pre-training (image adapted from [15])

Similarly as in [15], we don't make use of a non-linear projection between the representations from the encoders and the contrastive embedding space, as popularized in [1, 5]. This motivation comes from the fact that CLIP authors do not notice a difference in training efficiency between the two versions and speculate that non-linear projections may be co-adapted with details of current image only in self-supervised representation learning methods. In our approach, we simply modify the head of these encoders, and add a linear layer to output vectors in an embedding space of chosen dimension. We leverage the power of existing pretrained models known to have strong performance in the fields of computer vision and natural language processing.

Due to limited computational resources, we only experiment with two different pretrained image encoder architectures, a ResNet50 and a ViTB16. In CLIP, the author train series of 5 different ResNets and 3 different Vision Transformers from scratch.

2.2. Image Encoder

For the image encoder, we can use any standard Computer vision model, based on CNN and or Tranformer architetures. Here, we decide to make use of two different image encoder image architetures, Resnet50 [13] and ViTB16 [8]. We briefly recall the principle of these architecures.

2.2.1 ResNet

The main contribution of ResNet [13] resides in stacking residual blocks, allowing to skip some layers convolutional architectures. These allow to have networks that are easier to optimize, by solving vanishing gradient problems. These networks can gain accuracy from increased depth. In our experiments, we will use the ResNet50 architecture, made of 50 layers using these blocks.

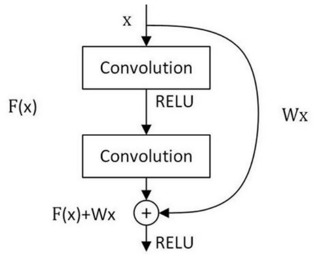


Figure 2. Residual block, used in the ResNet architecture

2.2.2 ViT

The Vision Transformer (ViT) [8], makes use of a transformer architecture. To do so, input images are split into patches, which are then linearly embeddeded, and treated as a sequence by adding position embeddings. This sequence can then be fed to a transformer encoder.

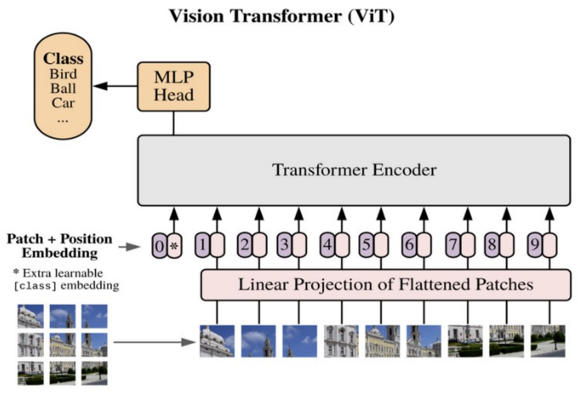

Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0 * 1 2 3 4 5 6 7 8 9
Linear Projection of Flattened Patches

Figure 3. An illustration of the ViT (Visual Transformer)

2.3. Text Encoder

For our text encoder, the idea is to use a transformer based language model. we choose to use a pretrained DistilBERT [18] model. This model, based on the originial BERT model [7], Knowledge distillation is performed during the pre-training phase, BERT model by 40%, while retaining 97% of its language understanding capabilities and being 60% faster.

[1]BERT [7], or Bidirectional Encoder Representations from Transformers, improves upon standard Transformers by removing the unidirectionality constraint by using a masked language model (MLM) pre-training objective. The masked language model randomly masks some of the tokens from the input, and the objective is to predict the original vocabulary id of the masked word based only on its context. Unlike left-to-right language model pre-training, the MLM objective enables the representation to fuse the left and the right context, which allows us to pre-train a deep bidirectional Transformer. In addition to the masked language model, BERT uses a next sentence prediction task that jointly pre-trains text-pair representations.

There are two steps in BERT: pre-training and fine-tuning. During pre-training, the model is trained on unlabeled data over different pre-training tasks. For fine-tuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters.

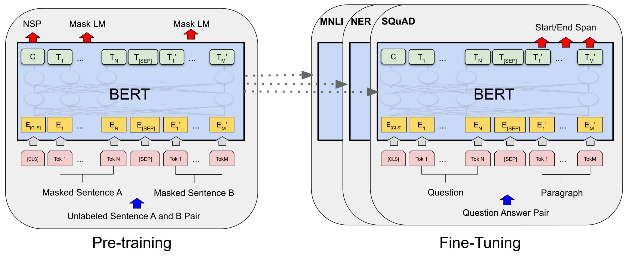


Figure 4. Illustration of Bert pre-training and fine-Tuning

2.4. Loss function

2.4.1 Contrastive Loss

The formula of contrastive loss, as first introduced in [12] is presented below.

$$\begin{aligned} L(W, Y, X_1, X_2) &= (1-Y) L_S(D_W^i) + Y L_D(D_W^i) \\ &= (1-Y)\frac{1}{2}(D_W)^2 + \frac{1}{2} Y \{\max(0, m - D_W)\} \end{aligned} \quad (1)$$

The label term Y specifies, whether the two given data points are similar or dissimilar. L_S refers to the loss for similar data points, which is simply the distance D_W between them. We can choose this distance metric, e.g to be cosine or euclidean. L_D refers to the loss for dissimilar data points, which is not simply the negative distance between points, but adds the concept of margin, denoted as m.

[1]We borrow the following paragraphs to https://paperswithcode.com/method/bert

The intuition of this function is that similar samples are penalized from being far from each other, while negative ones are penalized by being to close to each other. For negative samples, the concept of margin is introduced, used as a minimal distance that dissimilar points need to keep, such that they are penalized for being closer than the given margin.

2.4.2 InfoNCE losses

In this project, to conduct our experiments, we mainly use a loss function refered to as InfoNCE loss, popularized in the contrastive setting in [19], and first used in a multimodal text-image contrastive setting in the medical field [21] (from which the following paragraph is inspired), and also used in [15].

At training time, for each minibatch of N input pairs (x_v, x_u), denoting respectively image and text inputs, we compute their representations (I, T) in the embedding space. We use two similar loss functions for both image-to-text and text-to image similarities. These contrastive loss take the form of InfoNCE [19]. minimizing it leads to encoders that maximally preserve the mutual information between the true pairs under the representation functions. Intuitively, it is the log loss of an N -way classifier that tries to predict (I_i, T_i) as the true pair.

$$L_i^1 = -log \frac{exp(\langle I_i, T_i \rangle / \tau)}{\sum_{k=1}^N exp(\langle I_i, T_i \rangle / tau}$$

$$L_i^2 = -log \frac{exp(\langle T_i, I_i \rangle / \tau)}{\sum_{k=1}^N exp(\langle T_i, I_i \rangle / tau}$$

Here, $\langle I_i, T_i \rangle$ represents the cosine similarity, i.e $\langle I_i, T_i \rangle = v^T u / \|I_i\| \|T_i\|$. τ represents a temperature parameter. We then obtain our final loss as an average of these two, averaged over all positive pairs in our minibatch.

$$L = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} \left(L_i^1 + L_i^2 \right)$$

3. Experiments

3.1. Dataset

For our experiments, we use the *Food Ingredients and Recipes Dataset* [2], scraped from *epicurious.com*. The dataset consists of 13,582 images and each image comes with a corresponding title, a list of ingredients, and a list of instructions. On figure 5, we show an example from this dataset. We note than in the literature, datasets of larger scales, such as [6, 14] are often used.

[2]https://www.kaggle.com/datasets/pes12017000148/food-ingredients-and-recipe-dataset-with-images

Title :
```
Whole Wheat Pancake and Waffle Mix
```
Ingredients:
```
['6 cups white whole wheat flour', '1/3 cup
sugar', '2 tablespoons baking powder', '1
tablespoon baking soda', '2 teaspoons salt', '1
large egg, whisked', '1 cup buttermilk', '1
tablespoon vegetable or canola oil', '1 cup
pancake mix (above)']
```
Instructions:
```
1. Whisk the egg, buttermilk and oil in a large
bowl.
2. Whisk in the pancake mix until just combined.
3. Heat a large pan or griddle over medium heat
and grease with butter or oil.
4. Pour about 1 tablespoon of the pancake
mixture onto the griddle, making as many
pancakes as will fit and cook for 2 minutes.
5. Flip the pancakes and cook for one minute
longer and serve.
```

Figure 5. Example from the *Food Ingredients and Recipes Dataset*

3.2. quantitative study

In this subsection, we conduct a comparative study of different methods using a dataset that is split into training, validation, and test sets with proportions of 0.8, 0.1, and 0.1 respectively. Unless specified otherwise, we train our models for three epochs on a single RTX 3060 laptop GPU, using the InfoNCE loss variant presented previously. We use a batch size of 16. We choose an embedding space dimension of 100. To enhance the generalizability of our models, we apply basic data augmentation techniques for images. We optimize our models using stochastic gradient descent with momentum, using a learning rate of 0.001 and a momentum value of 0.9. The main part of the bert model is frozen during training, except for its head. We evaluate the performance of our models for image and text retrieval using metrics recall@1, recall@5 and medr on tasks of text-to-image and image-to-text retrieval on the entire test dataset, consisting of around 1.3k images.

3.2.1 image encoder

Encoder	Recall@1	Recall@5	Medr
TTI			
ViT B16	0.382	0.838	2.111
ResNet50	0.400	0.818	2.082
ITT			
ViT B16	0.381	0.842	2.171
ResNet50	0.393	0.823	2.170

Table 1. Comparison of the ViT and ResNet image encoders, pretrained on ImageNet. Top: Text-to-Image retrieval, Bottom : Image-to-text retrieval.

In our task, the ResNet50 seems to perform better than the ViT B16 , which may be due to the model size, the lack of data, or the low amount of epochs. The ViT also takes more time to train. In our case, for 3 epochs, the training time for the ViT is around 115m, while it is around 105 for the ResNet.

3.2.2 text information

Method	Recall@1	Recall@5	Medr
TTI			
1	0.400	0.818	2.082
2	0.428	0.872	1.900
3	0.498	0.903	1.523
ITT			
1	0.393	0.823	2.170
2	0.415	0.842	2.171
3	0.501	0.907	1.535

Table 2. Ablation study on the use of different textual recipe information. 1: Title, 2: Title + Ingredients, 3:Title + Ingredients + Instructions. ResNet50 encoder

The method we use for several text modalities is simple text concatenation. Before tokenizing, we simply concatenate all the information we have, with no additional text preprocessing. On Table 3, we can notice that the more information we use, the better our model performs. We could use other ways to preprocess this information, and other ways to incorporate them into our model. For instance, we could train separate encoders for recipes and other information, and sum their embeddings, as seen in [17].

3.2.3 effect of pretraining

Here, we highlight the importance of using pretrained models.

Method	Recall@1	Recall@5	Medr
TTI			
pretraining	0.382	0.838	2.111
no pretraining	0.093	0.328	7.4
ITT			
pretraining	0.381	0.842	2.171
no pretraining	0.0828	0.348	7.44

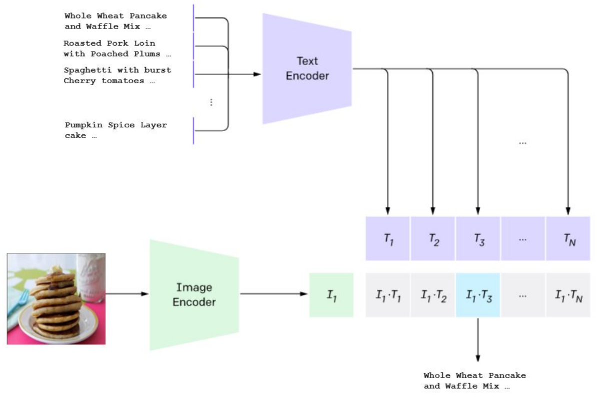
Table 3. Ablation study on the use pretraining the ResNet50 image encoder (on ImageNet). We still use a pretrained DistilBert encoder

Other studies, such as the effect of the embedding dimension, batch size, as well as the comparison with other methods would be useful. We also compare our method with zero-shot pretrained clip, not finetuned with our data, and this performs poorly, as can be seen briefly on Figure 10.

3.3. Qualitative results

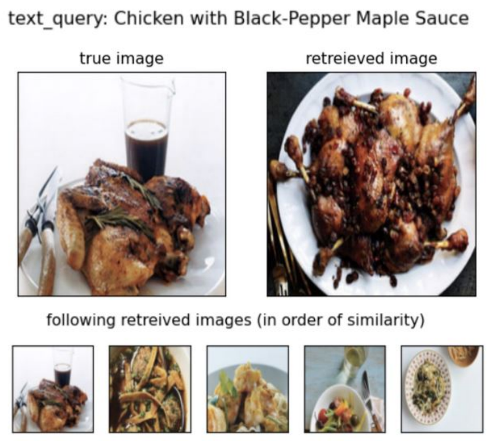
3.3.1 Text to image retrieval

In order to perform the retrieval task, we simply have to put our query in the image encoder, as well as all the texts in the text encoder. Then, we compute cosine similarities between the query encoder features and encoded features from the text database. Then, we choose the closest matching point. We illustrate this on Figure 6. On figure 7, we show qualitative results of this retrieval task.



Whole Wheat Pancake and Waffle Mix …
Roasted Pork Loin with Poached Plums …
Spaghetti with burst Cherry tomatoes …
Pumpkin Spice Layer cake …
Text Encoder
T_1
T_2
T_3
T_N
Image Encoder
I_1
$I_1 \cdot T_1$
$I_1 \cdot T_2$
$I_1 \cdot T_3$
$I_1 \cdot T_N$
Whole Wheat Pancake and Waffle Mix …

Figure 6. Retrieval task



text_query: Chicken with Black-Pepper Maple Sauce
true image
retreieved image
following retreived images (in order of similarity)

Figure 7. Qualitative example of Text to image retrieval

3.3.2 Image to text retrieval

This task is very similar as the text to image retrieval. Instad, we use a text query and an image database. In Figure 8, we show some qualitative results.

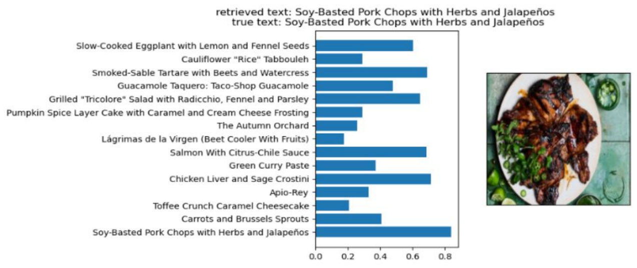


Figure 8. Qualitative example of Image to text retrieval

3.4. Additional Results

3.4.1 similarity matrix

On Figure 9, we can notice that the diagonal, corresponding to positive pairs, has high similarities. Moreover, we can generally notice sensible common points between pairs that seem close to each other in terms of food visuals and recipes. On Figure 10 We also show the same matrix using a pretrained CLIP model non finetuned on our data, which performs poorly.

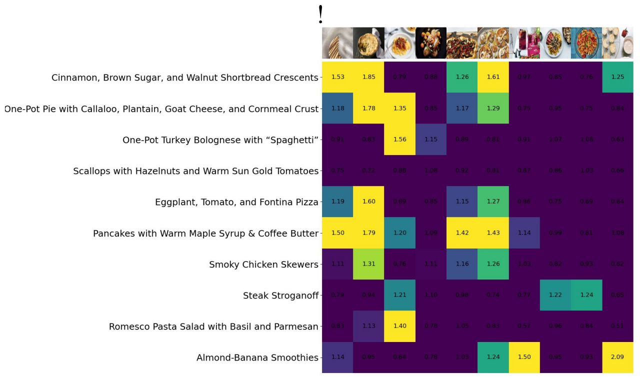


Figure 9. Similarity matrix between text and image features of a batch of test data. Model used: resnet, distilbert encoders, trained on all text information (3)

3.4.2 embedding space visualization

On Figure 11, we visualise embeddings using PCA. We can notice interesting patterns, where close dishes are close to each other in the embedding space, for instance meat dishes, cocktails, or cakes.

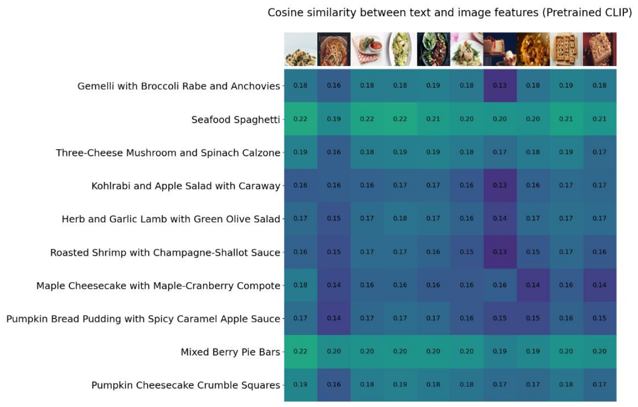


Figure 10. Similarity matrix between text and image features of a batch of test data. CLIP pre-trained model, not fine-tuned. Prompts used: 'Photo of a + *Title*'

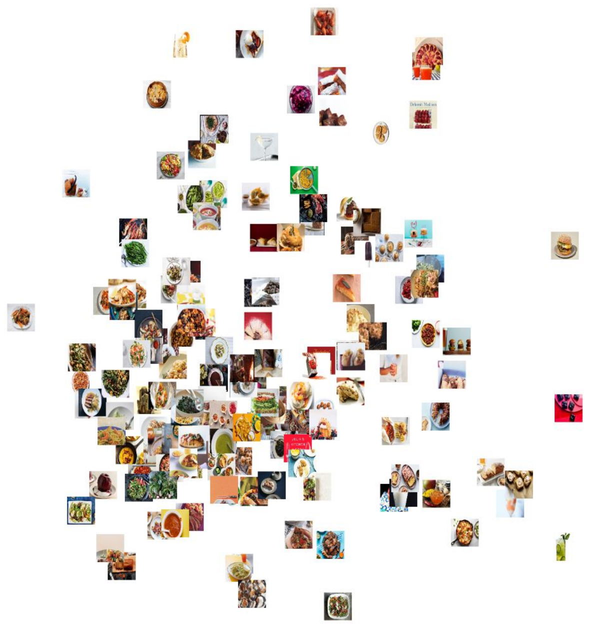


Figure 11. Visualization of the embedding space through PCA. Texts are encoded through the text encoder, and we perform PCA of their embeddings. Associated images are scattered in R2.

4. Conclusion

To conclude, in this project, we proposed a simple model combining text recipe information with associated images. This model made of a text encoder and an image encoder, is trained in a contrastive way. It allows us to perform image-to-text retrieval as well as text-to-image tasks. To improve over this model, it would be useful to explore new models, such as hierarchical transformers [16], treat the text information in a better way, and perform more quantitative studies to better understand the effect of certain parameters on the model's performance. A further step would consist in making use of the embedding space and associate it with a generative model to create a text-to-image generator.

References

[1] Philip Bachman, R. Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *CoRR*, abs/1906.00910, 2019. 2

[2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 446–461, Cham, 2014. Springer International Publishing. 1

[3] Micael Carvalho, Rémi Cadène, David Picard, Laure Soulier, Nicolas Thome, and Matthieu Cord. Cross-modal retrieval in the cooking context: Learning semantic text-image embeddings. *CoRR*, abs/1804.11146, 2018. 1

[4] Jingjing Chen, Chong-Wah Ngo, Fuli Feng, and Tat-Seng Chua. Deep understanding of cooking procedure for cross-modal recipe retrieval. *Proceedings of the 26th ACM international conference on Multimedia*, 2018. 1

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020. 2

[6] Xin Chen, Hua Zhou, and Liang Diao. Chinesefoodnet: A large-scale image dataset for chinese food recognition. *CoRR*, abs/1705.02743, 2017. 1, 3

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. 2, 3

[8] 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. *CoRR*, abs/2010.11929, 2020. 2

[9] Mikhail Fain, Andrey Ponikar, Ryan Fox, and Danushka Bollegala. Dividing and conquering cross-modal recipe retrieval: from nearest neighbours baselines to sota. *CoRR*, abs/1911.12763, 2019.

[10] Han Fu, Rui Wu, Chenghao Liu, and Jianling Sun. MCEN: bridging cross-modal gap between cooking recipes and dish images with latent variable model. *CoRR*, abs/2004.01095, 2020. 1

[11] Han Fu, Rui Wu, Chenghao Liu, and Jianling Sun. MCEN: bridging cross-modal gap between cooking recipes and dish images with latent variable model. *CoRR*, abs/2004.01095, 2020. 1

[12] Raia Hadsell, Sumit Chopra, and Yann Lecun. Dimensionality reduction by learning an invariant mapping. pages 1735 – 1742, 02 2006. 3

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 2

[14] Javier Marín, Aritro Biswas, Ferda Ofli, Nicholas Hynes, Amaia Salvador, Yusuf Aytar, Ingmar Weber, and Antonio Torralba. Recipe1m+: A dataset for learning cross-modal embeddings for cooking recipes and food images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):187–203, 2021. 1, 3

[15] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021. 1, 2, 3

[16] Amaia Salvador, Erhan Gundogdu, Loris Bazzani, and Michael Donoser. Revamping cross-modal recipe retrieval with hierarchical transformers and self-supervised learning. *CoRR*, abs/2103.13061, 2021. 1, 6

[17] Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marin, Ferda Ofli, Ingmar Weber, and Antonio Torralba. Learning cross-modal embeddings for cooking recipes and food images. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3068–3076, 2017. 1, 4

[18] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. 2

[19] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018. 3

[20] Hao Wang, Doyen Sahoo, Chenghao Liu, Ee-Peng Lim, and Steven C. H. Hoi. Learning cross-modal embeddings with adversarial networks for cooking recipes and food images. *CoRR*, abs/1905.01273, 2019. 1

[21] Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D. Manning, and Curtis P. Langlotz. Contrastive learning of medical visual representations from paired images and text. *CoRR*, abs/2010.00747, 2020. 3

[22] Bin Zhu, Chong-Wah Ngo, Jingjing Chen, and Yanbin Hao. R^{2}gan: Cross-modal recipe retrieval with generative adversarial network. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11469–11478, 2019. 1