Recipe1M+: A Dataset for Learning Cross-Modal Embeddings for Cooking Recipes and Food Images

Javier Marín[1]*, Aritro Biswas[1]*, Ferda Ofli[2], Nicholas Hynes[1], Amaia Salvador[3], Yusuf Aytar[1], Ingmar Weber[2], Antonio Torralba[1]
[1]Massachusetts Institute of Technology [2]Qatar Computing Research Institute, HBKU
[3]Universitat Politècnica de Catalunya
{abiswas,nhynes}@mit.edu, {jmarin,yusuf,torralba}@csail.mit.edu, amaia.salvador@upc.edu, {fofli,iweber}@hbku.edu.qa

Abstract—In this paper, we introduce Recipe1M+, a new large-scale, structured corpus of over one million cooking recipes and 13 million food images. As the largest publicly available collection of recipe data, Recipe1M+ affords the ability to train high-capacity models on aligned, multimodal data. Using these data, we train a neural network to learn a joint embedding of recipes and images that yields impressive results on an image-recipe retrieval task. Moreover, we demonstrate that regularization via the addition of a high-level classification objective both improves retrieval performance to rival that of humans and enables semantic vector arithmetic. We postulate that these embeddings will provide a basis for further exploration of the Recipe1M+ dataset and food and cooking in general. Code, data and models are publicly available.

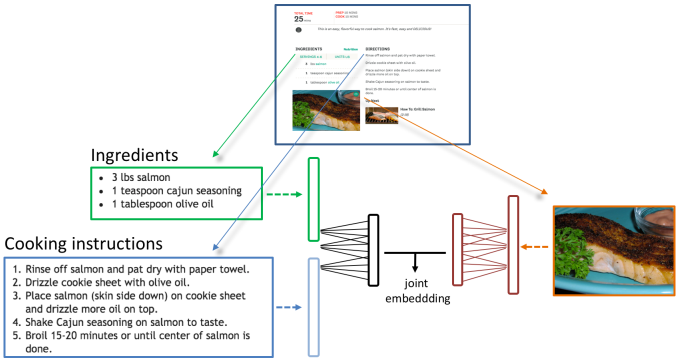
Index Terms—Cross-modal, deep learning, cooking recipes, food images

1 Introduction

There are few things so fundamental to the human experience as food. Its consumption is intricately linked to our health, our feelings and our culture. Even migrants starting a new life in a foreign country often hold on to their ethnic food longer than to their native language. Vital as it is to our lives, food also offers new perspectives on topical challenges in computer vision like finding representations that are robust to occlusion and deformation (as occur during ingredient processing).

The profusion of online recipe collections with user-submitted photos presents the possibility of training machines to automatically understand food preparation by jointly analyzing ingredient lists, cooking instructions and food images. Far beyond applications solely in the realm of culinary arts, such a tool may also be applied to the plethora of food images shared on social media to achieve insight into the significance of food and its preparation on public health [1] and cultural heritage [2]. Developing a tool for automated analysis requires large and well-curated datasets.

The emergence of massive labeled datasets [3], [4] and deeply-learned representations [5], [6], [7] have redefined the state-of-the-art in object recognition and scene classification. Moreover, the same techniques have enabled progress in new domains like dense labeling and image segmentation. Perhaps the introduction of a new large-scale food dataset–complete with its own intrinsic challenges–will yield a similar advancement of the field. For instance, categorizing an ingredient's state (e.g., sliced, diced, raw, baked, grilled, or boiled) provides a unique challenge in attribute recognition–one that is not well posed by existing datasets. Furthermore, the free-form nature of food suggests a departure from the concrete task of classification in favor of a more nuanced objective that integrates variation in a recipe's structure. Hence, we argue that food images must be analyzed together with accompanying recipe ingredients and instructions in order to acquire a comprehensive understanding of "behind-the-scene" cooking process as illustrated in Fig. 1.



- 3 lbs salmon
- 1 teaspoon cajun seasoning
- 1 tablespoon olive oil

1. Rinse off salmon and pat dry with paper towel.
2. Drizzle cookie sheet with olive oil.
3. Place salmon (skin side down) on cookie sheet and drizzle more oil on top.
4. Shake Cajun seasoning on salmon to taste.
5. Broil 15-20 minutes or until center of salmon is done.

Fig. 1. **Learning cross-modal embeddings** from recipe-image pairs collected from online resources. These embeddings enable us to achieve in-depth understanding of food from its ingredients to its preparation.

Existing work, however, has focused largely on the use of medium-scale image datasets for performing food categorization. For instance, Bossard et al. [8] introduced the Food-101 visual classification dataset and set a baseline of 50.8% accuracy. Even with the impetus for food image categorization, subsequent work by [9], [10] and [11] could only improve this result to 77.4%, 79% and 80.9%, respectively, which indicates that the size of the dataset may be the limiting factor. Although Myers et al. [10] built upon Food-101 to tackle the novel challenge of estimating a meal's energy content, the segmentation and depth information used in their work are not made available for further exploration.

*contributed equally.

In this work, we address data limitations by introducing the large-scale Recipe1M+ dataset which contains one million structured cooking recipes and their images. Additionally, to demonstrate its utility, we present the im2recipe retrieval task which leverages the full dataset–images and text–to solve the practical and socially relevant problem of demystifying the creation of a dish that can be seen but not necessarily described. To this end, we have developed a multimodal neural model which jointly learns to embed images and recipes in a common space which is semantically regularized by the addition of a high-level classification task. The performance of the resulting embeddings is thoroughly evaluated against baselines and humans, showing remarkable improvement over the former while faring comparably to the latter. With the release of Recipe1M+, we hope to spur advancement on not only the im2recipe task but also heretofore unimagined objectives which require a deep understanding of the domain and its modalities.

1.1 Related Work

Since we presented our initial work on the topic back in 2017 [12], several related studies have been published and we feel obliged to provide a brief discussion about them.

Herranz et al. [13], besides providing a detailed description on recent work focusing on food applications, propose an extended multimodal framework that relies on food imagery, recipe and nutritional information, geolocation and time, restaurant menus and food styles. In another study, Min et al. [14] present a multi-attribute theme modeling (MATM) approach that incorporates food attributes such as cuisine style, course type, flavors or ingredient types. Then, similar to our work, they train a multimodal embedding which learns a common space between the different food attributes and the corresponding food image. Most interesting applications of their model include flavor analysis, region-oriented food summary, and recipe recommendation. In order to build their model, they collect all their data from a single data source, i.e., Yummly[1], which is an online recipe recommendation system.

In another interesting study, Chang et al. [15] focus on analyzing several possible preparations of a single dish, like "chocolate chip cookie." The authors design an interface that allows users to explore the similarities and differences between such recipes by visualizing the structural similarity between recipes as points in a space, in which clusters are formed according to how similar recipes are. Furthermore, they examine how cooking instructions overlap between two recipes to measure recipe similarity. Our work is of a different flavor, as the features they use to measure similarity are manually picked by humans, while ours are automatically learned by a multimodal network.

Getting closer to the information retrieval domain, Engilberge et al. [16] examine the problem of retrieving the best matching caption for an image. In order to do so, they use neural networks to create embeddings for each caption, and retrieve the one whose embedding most closely matches the embedding of the original image. In our work, we aim to also use embeddings to retrieve the recipe matching an image, or vice versa. However, since our domain involves cooking recipes while theirs only involves captions, we account for two separate types of text – ingredients and cooking instructions – and combine them in a different way in our model.

Alternatively, Chen et al. [17] study the task of retrieving a recipe matching a corresponding food image in a slightly different way. The authors find that, although ingredient composition is important to the appearance of food, other attributes such as the manner of cutting and manner of cooking ingredients also play a role in forming the food's appearance. Given a food image, they attempt to predict ingredient, cutting and cooking attributes, and use these predictions to help retrieve the correct corresponding recipe. With our model, we attempt to retrieve the recipe directly, without first predicting attributes like ingredients, cutting and cooking attributes, separately. Furthermore, along with retrieving the recipe matching an image, our model also allow to retrieve the image matching a corresponding recipe.

The two most relevant studies to the current one are presented in [18] and [19]. Different from our work, Chen et al. [18] approach the image-to-recipe retrieval problem from the perspective of attention modeling where they incorporate word-level and sentence-level attentions into their recipe representation and align them with the corresponding image representation such that both text and visual features have high similarity in a multi-dimensional space. Another difference is that they employ a rank loss instead of a pairwise similarity loss as we do. These improvements effectively lead to slight performance increases in both image-to-recipe and recipe-to-image retrieval tasks.

On the other hand, building upon the same network architecture as in our original work [12] to represent the image and text (recipe) modalities, Carvalho et al. [19] improve our initial results further by proposing a new objective function that combines retrieval and classification tasks in a double-triplet learning scheme. This new scheme captures both instance-based (i.e., fine-grained) and semantic-based (i.e., high-level) structure simultaneously in the latent space since the semantic information is directly injected into the cross-modal metric learning problem as opposed to our use of classification task as semantic regularization. Additionally, they follow an adaptive training strategy to account for the vanishing gradient problem of the triplet losses and use the MedR score instead of the original loss in the validation phase for early stopping. We also find that using the MedR score as the performance measure in the validation phase is more stable. However, our work is orthogonal to both of these studies, i.e., their performances can be further improved with the use of our expanded dataset and the quality of their embeddings can be further explored with various arithmetics presented in this submission.

The rest of the paper is organized as follows. In Section 2, we introduce our large-scale, multimodal cooking recipe dataset and provide details about its collection process. We describe our recipe and image representations in Section 3 and present our neural joint embedding model in Section 4. Then, in Section 5, we discuss our semantic regularization approach to enhance our joint embedding model. In Section 6, we present results from our various experiments and conclude the paper in Section 7.

2 Dataset

Due to their complexity, textually and visually, (e.g., ingredient-based variants of the same dish, different presentations, or multiple ways of cooking a recipe), understanding food recipes demands a large, general collection of recipe data. Hence, it should not be surprising that the lack of a larger body of work on the topic could be the result of missing such a collection. To our knowledge, practically all the datasets publicly available in the research field either contain only categorized images [8], [10], [20], [21] or simply recipe text [22]. Only recently have a few datasets been released that include both recipes and images. For instance, Wang

[1] https://www.yummly.com/

et al. [23] released a multimodal food dataset which has 101k images divided equally among 101 food categories; the recipes for each are however raw HTML. In a later work, Chen and Ngo [24] presented a dataset containing 110,241 images annotated with 353 ingredient labels and 65,284 recipes, each with a brief introduction, ingredient list, and preparation instructions. Of note is that the dataset only contains recipes for Chinese cuisine.

Although the aforementioned datasets constitute a large step towards learning richer recipe representations, they are still limited in either generality or size. As the ability to learn effective representations is largely a function of the quantity (especially when learning features using deep architectures) and quality of the available data, we create and release publicly a new, large-scale corpus of structured recipe data that includes over 1M recipes and 13M images. In comparison to the current largest datasets in this domain, the Recipe1M+ includes twice as many recipes as [22] and 130 times as many images as [24].

We created the Recipe1M+ dataset in two phases. In the first phase, we collected a large dataset of cooking recipes paired with food images, all scraped from a number of popular cooking websites, which resulted in more than 1M cooking recipes and 800K food images (i.e., Recipe1M [12]). Then, in the second phase, we augmented each recipe in this initial collection with food images downloaded from the Web using a popular image search engine, which amounted to over 13M food images after cleaning and removing exact-and-near duplicates. In the following subsections, we elaborate further on these data collection phases, outline how the dataset is organized, and provide analysis of its contents.

2.1 Data Collection from Recipe Websites

The recipes were scraped from over two dozen popular cooking websites and processed through a pipeline that extracted relevant text from the raw HTML, downloaded linked images, and assembled the data into a compact JSON schema in which each datum was uniquely identified. As part of the extraction process, excessive whitespace, HTML entities, and non-ASCII characters were removed from the recipe text. Finally, after removing duplicates and near-matches (constituting roughly 2% of the original data), the retained dataset contained over 1M cooking recipes and 800K food images (i.e., Recipe1M [12]). Although the resulting dataset is already larger than any other dataset in this particular domain (i.e., includes twice as many recipes as [22] and eight times as many images as [24]), the total number of images is not yet at the same scale as the largest publicly available datasets such as ImageNet [3] and Places [25], which contain tens of millions of images, in the computer vision community. Therefore, in the next phase, we aimed to extend the initial collection of images by querying for food images through an image search engine.

2.2 Data Extension using Image Search Engine

Thanks to the latest technological infrastructure advances, half the population of the entire world have become Internet users[2]. Online services ranging from social networks to simple websites have grown into data containers where users share images, videos, or documents. Companies like Google, Yahoo, and Microsoft, among others, offer public search engines that go through the entire Internet looking for websites, videos, images and any other type of content that matches a text query (some of them also support image queries). Looking at the search results for a given recipe title (e.g., "chicken wings") in Fig. 2, one can say that the retrieved images are generally of very good quality. We also observed during the first phase of data collection from recipe websites that users were often using images from other recipes of the same dish (sometimes with slight differences) to visually describe theirs. Motivated by these insights, we downloaded a large amount of images using as queries the recipe titles collected from the recipe websites in the first phase.

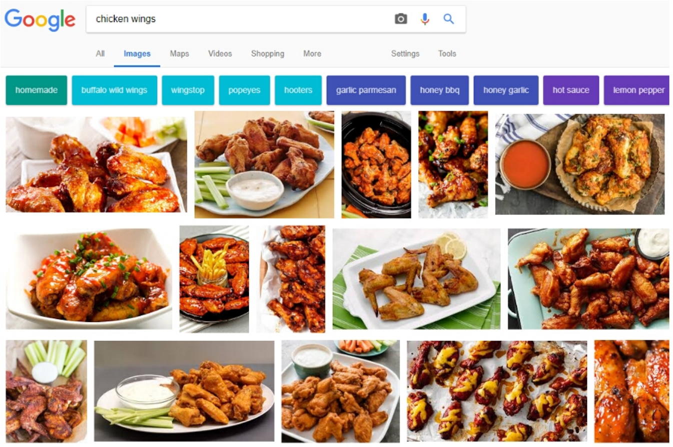


Fig. 2. **Google image search results.** The query used is *chicken wings*.

TABLE 1
Dataset sizes. Number of recipes and images in training, validation and test sets of each dataset.

		Recipe1M	intersection	Recipe1M+
Partition	# Recipes	# Images	# Images	# Images
Training	720,639	619,508	493,339	9,727,961
Validation	155,036	133,860	107,708	1,918,890
Test	154,045	134,338	115,373	2,088,828
Total	1,029,720	887,706	716,480	13,735,679

Data Download. We targeted collecting 50M images, i.e., 50 images per recipe in the initial collection. In order to amass such a quantity of images, we chose the Google search engine. As mentioned before, we used the title of each recipe as a query. Out of the Google search results, we selected the top 50 retrieved images and stored locally their image URLs. For this task, we used publicly available Python libraries on ten servers in parallel for several days. Then, to download images simultaneously, we made use of Aria2[3], a publicly available download utility. In the end, we managed to download over 47M images as some of the image URLs either were corrupted or did not exist any more.

Data Consolidation. One of the first tasks, besides removing corrupted or wrong format images, was eliminating the duplicate images. For this task, we simply used a pre-trained ResNet18 [7] as a feature extractor (by removing its last layer for classification) and computed pairwise euclidean distances between the collected images. During this cleanse process, we combined the initial set of images collected from recipe websites and the new ones collected via Google image search. After this first stage, we removed over

[2] https://www.internetworldstats.com/stats.htm

[3] https://aria2.github.io/

32M duplicate images (those with an euclidean distance of 0). We only kept one representative for each duplicate cluster. Later, we visually inspected the remaining images and realized that a significant amount of them were still either duplicates or near-duplicates. The main reason we could not detect some of these duplicates in the first stage was due to compression or rescaling operations applied to the images, which cause slight modifications to their feature representation. By using distances between them, and removing those that were close enough, we managed to eliminate these duplicates. Near-duplicates, instead, were due to distortions (i.e., aspect-ratio changes), crops, added text into the image, and other alterations. To remove near-duplicates, after trying different strategies, we chose a harsh distance threshold between images, which meant we had to eliminate a certain amount of good examples, as well. This strategy was used between different partitions (i.e., training, test and validation). That is, we allowed near-duplicates within a partition to a certain extent (using a relaxed threshold). Additionally, we ran a face detector over the images and removed those that had a face with high confidence. Thanks to computing distances, we also found non-recipe images such as images with nutritional facts. Images containing only text were close to each other within the feature space. In order to compute the distances between images, we used C++ over Python for efficiency purposes.

Regarding the recipes sharing the same title, we uniformly distributed the queried images for a particular non-unique title among the recipes sharing it. This helped us to avoid having different recipes with the exact same food images. In the last two paragraphs of Section 2.5, we describe an experiment performed by humans that supports the validity of spreading them uniformly.

In order to re-balance the dataset in terms of partitions, we slightly modified the images belonging to each partition. For a fair comparison between the Recipe1M and Recipe1M+ in our experiments, we created an *intersection* version of the initial dataset, which simply contains the images that were common between both of them. One would expect Recipe1M images to be a subset of Recipe1M+ images, but due to the re-balance and the cleanse of near-duplicates, which were not done in the original Recipe1M dataset, this was no longer true. Table 1 shows the small differences in numbers.

2.3 Nutritional Information

The ingredient lists in the recipes scraped from the recipe websites include the ingredient, quantity and unit information altogether in a single sentence in several cases. In order to simplify the task of automatically computing the nutritional information of a recipe, we decided to encapsulate these three different fields, i.e., (i) the ingredient, (ii) the units, and (iii) the quantity, separately in the dataset structure. After identifying different type of sentences that followed the ‘quantity-unit-ingredient’ sequence pattern in the recipe ingredient lists, we used a natural language processing toolkit[4] to tag every single word within each of these sentences (e.g., [(‘2’, ‘CD’), (‘cups’, ‘NNS’), (‘of’, ‘IN’), (‘milk’, ‘NN’)]). Every ingredient in the dataset that followed the sentence structure (e.g., ‘4 teaspoons of honey’) of one of those we identified, was selected for further processing. We then went through the unit candidates of these sentences and chose only the measurable ones (some non-measurable units are for instance, a *bunch*, a *slice* or a *loaf*). Table 2 shows the 20 different units we found. 103,152 unique recipes had measurable units and numerical quantities defined for all their ingredients. Regarding numerical quantities, these recipes contained 1,002 different ones.

TABLE 2
Recipe1M+ units. The 20 measurable units isolated in the dataset.

units
bushel, cup, dash, drop, fl. oz, g, gallon, glass, kg, liter, ml, ounce, pinch, pint, pound, quart, scoop, shot, tablespoon, teaspoon

Once we finished the previous stage, we matched thousands of ingredient names with a publicly available nutrient database [26] assembled by the United States Department of Agriculture (USDA). This database provides the nutritional content of over 8,000 generic and proprietary-branded foods. In order to facilitate the matching process, we first reduced the ingredient list to contain only the first word within the sentence (after removing quantities and units), obtaining a total of 6,856 unique words. Then, for each unique ingredient we picked, when available, the second word of the sentence. Due to multiple different sentences having the same first word, we did only take one example out of the possible ones. We went through each single bigram and only selected those that were food ingredients, e.g., *apple juice* or *cayenne pepper*. If the second word was nonexistent, e.g., $1/2$ *spoon of sugar*, or was not part of a standard ingredient name, e.g., 1 *cup of water at* $40\,^{\circ}\mathrm{C}$, we only selected the first word, i.e., *sugar* and *water*, respectively. We created a corpus of 2,057 unique different ingredients with their singular and plural versions, and, in some cases, synonyms or translations, e.g., *cassava* can be also called *yuca*, *manioc* or *mandioca*. We found ingredient names from different nationalities and cultures, such as Spanish, Turkish, German, French, Polish, American, Mexican, Jewish, Indian, Arab, Chinese or Japanese among others. Using the ingredient corpus we assigned to each ingredient sentence the closest ingredient name by simply verifying that all the words describing the ingredient name were within the original ingredient sentence. We found 68,450 recipes with all their ingredients within the corpus. The matching between the USDA database and the new assigned ingredient names, similarly as before, was done by confirming that all the words describing the ingredient name were within one of the USDA database food instances. We inspected the matching results to assure the correctness. In the end, we obtained 50,637 recipes with nutritional information (mapping example: *American cheese* ⇒ *cheese, pasteurized process, American, without added vitamin d*). In Fig. 3, we can see a 2D visualization of the embeddings of these recipes that also include images, using t-SNE [27]. Recipes are shown in different colors based on their semantic category (see Section 5). In Fig. 4, we can see the same embedding but this time showing the same recipes on different colors depending on how healthy they are in terms of sugar, fat, saturates, and salt. We used the traffic lights[5] definition established by the Food Standards Agency (FSA).

2.4 Data Structure

The contents of the Recipe1M dataset can logically be grouped into two layers. The first layer (i.e., Layer 1) contains basic information including a title, a list of ingredients, and a sequence of instructions

[4] http://www.nltk.org/

[5] https://www.resourcesorg.co.uk/assets/pdfs/foodtrafficlight1107.pdf

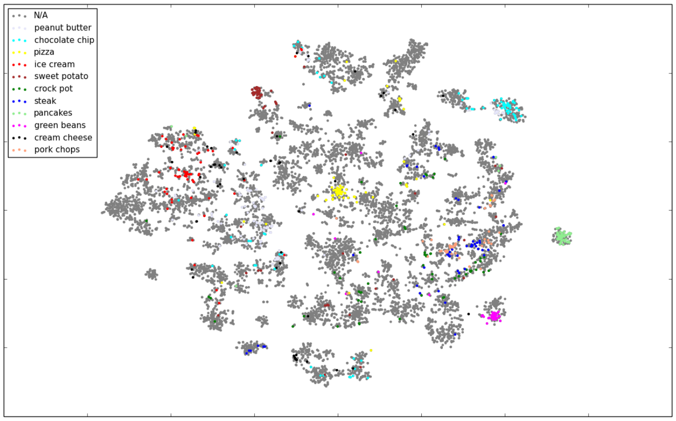


Fig. 3. **Embedding visualization using t-SNE**. Legend depicts the recipes that belong to the top 12 semantic categories used in our semantic regularization (see Section 5 for more details).

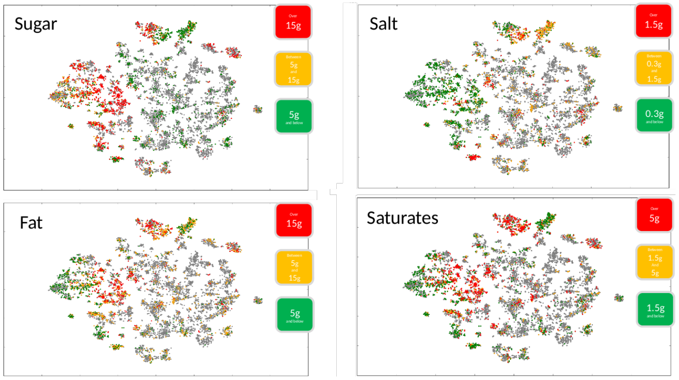


Fig. 4. **Healthiness within the embedding.** Recipe health is represented within the embedding visualization in terms of sugar, salt, saturates, and fat. We follow FSA traffic light system to determine how healthy a recipe is.

for preparing a dish; all of these data are provided as free text. Additional fields such as unit and quantity are also available in this layer. In cases where we were unable to extract unit and quantity from the ingredient description, these two fields were simply left empty for the corresponding ingredient. Nutritional information (i.e., total energy, protein, sugar, fat, saturates, and salt content) is only added for those recipes that contained both units and quantities as described in Section 2.3. FSA traffic lights are also available for such recipes. The second layer (i.e., Layer 2) builds upon the first layer and includes all images with which the recipe is associated–these images are provided as RGB in JPEG format. Additionally, a subset of recipes are annotated with course labels (e.g., appetizer, side dish, or dessert), the prevalence of which are summarized in Fig. 5. For Recipe1M+, we provide same Layer 1 as described above with different partition assignments and Layer 2 including the 13M images.

2.5 Analysis

Recipe1M (hence Recipe1M+) includes approximately 0.4% duplicate recipes and, excluding those duplicate recipes, 20% of recipes have non-unique titles but symmetrically differ by a median of 16 ingredients. 0.2% of recipes share the same ingredients but are relatively simple (e.g., spaghetti, or granola), having a median of six ingredients. Approximately half of the recipes did not have any images in the initial data collection from recipe websites. However, after the data extension phase, only around 2% of the recipes are left without any associated images. Regarding the experiments, we carefully removed any exact duplicates or recipes sharing the same image in order to avoid overlapping between training and test sets. As detailed earlier in Table 1, around 70% of the data is labeled as training, and the remainder is split equally between the validation and test sets. During the dataset extension, as we mentioned earlier, we also created an *intersection* dataset in order to have a fair comparison of the experimental results on both the initial and the extended versions of the dataset.

According to Fig. 5, the average recipe in the dataset consists of nine ingredients which are transformed over the course of ten instructions. One can also observe that the distributions of data are heavy tailed. For instance, of the 16k ingredients identified as unique (in terms of phrasing), only 4,000 account for 95% of occurrences. At the low end of instruction count–particularly those with one step–one will find the dreaded *Combine all ingredients*. At the other end are lengthy recipes and ingredient lists associated with recipes that include sub-recipes.

A similar issue of outliers exists also for images: as several of the included recipe collections curate user-submitted images, popular recipes like chocolate chip cookies have orders of magnitude more images than the average. Notably, the number of unique recipes that came with associated food images in the initial data collection phase was 333K, whilst after the data extension phase, this number reached to more than 1M recipes. On average, the Recipe1M+ dataset contains 13 images per recipe whereas Recipe1M has less than one image per recipe, 0.86 to be exact. Fig. 5 also depicts the images vs recipes histogram for Recipe1M+, where over half million recipes contain more than 12 images each.

To evaluate further the quality of match between the queried images and the recipes, we performed an experiment on Amazon Mechanical Turk (AMT) platform[6]. We randomly picked 3,455 recipes, containing at most ten ingredients and ten instructions, from the pool of recipes with non-unique titles. Then, for each one of these recipes, we showed AMT workers a pair of images and asked them to choose which image, A or B, was the best match for the corresponding recipe. The workers also had the options of selecting ‘both images’ or ‘none of them’. Image A and image B were randomly chosen; one from the original recipe (i.e., Recipe1M) images and the other one from the queried images collected during the dataset expansion for the corresponding recipe title. We also changed the order of image A and image B randomly. We explicitly asked the workers to check all the ingredients and instructions. Only master workers were selected for this experiment.

Out of 3,455 recipes, the workers chose 971 times the original recipe image (28.1%); 821 times the queried one (23.8%); 1581 times both of them (45.8%); and 82 times none of them (2.4%). Given the difference between the original recipe image vs. the queried image is less than 5%, these results show that the extended dataset is not much noisier than the original Recipe1M.

3 Learning Embeddings

In this section, we describe our neural joint embedding model. Here, we utilize the paired (recipe and image) data in order to learn a common embedding space as illustrated in Fig. 1. Next, we discuss recipe and image representations, and then, we describe our neural joint embedding model that builds upon recipe and image representations.

[6] http://mturk.com

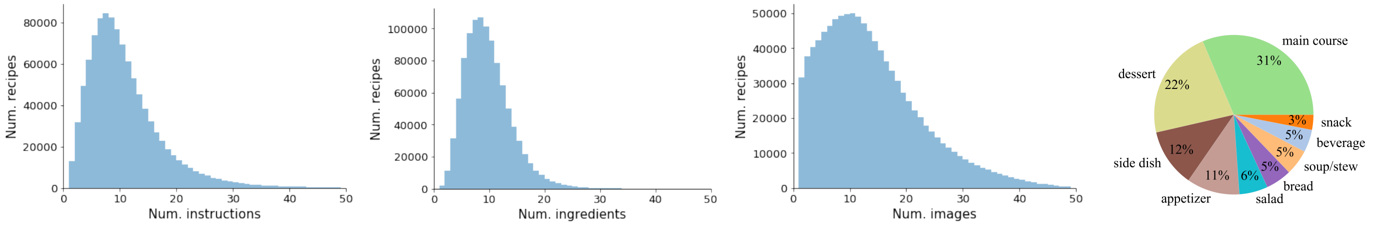


Fig. 5. **Dataset statistics.** Prevalence of course categories and number of instructions, ingredients and images per recipe in Recipe1M+.

3.1 Representation of Recipes

There are two major components of a recipe: its ingredients and cooking instructions. We develop a suitable representation for each of these components.

Ingredients. Each recipe contains a set of ingredient text as shown in Fig. 1. For each ingredient we learn an ingredient level word2vec [28] representation. In order to do so, the actual ingredient names are extracted from each ingredient text. For instance in “2 tbsp of olive oil” the *olive_oil* is extracted as the ingredient name and treated as a single word for word2vec computation. The initial ingredient name extraction task is solved by a bi-directional LSTM that performs logistic regression on each word in the ingredient text. Training is performed on a subset of our training set for which we have the annotation for actual ingredient names. Ingredient name extraction module works with 99.5% accuracy tested on a held-out set.

Cooking Instructions. Each recipe also has a list of cooking instructions. As the instructions are quite lengthy (averaging $\sim$208 words) a single LSTM is not well suited to their representation as gradients are diminished over the many time steps. Instead, we propose a two-stage LSTM model which is designed to encode a sequence of sequences. First, each instruction/sentence is represented as a *skip-instructions* vector, and then, an LSTM is trained over the sequence of these vectors to obtain the representation of all instructions. The resulting fixed-length representation is fed into to our joint embedding model (see instructions-encoder in Fig. 6).

Skip-instructions. Our cooking instruction representation, referred to as *skip-instructions*, is the product of a sequence-to-sequence model [29]. Specifically, we build upon the technique of skip-thoughts [30] which encodes a sentence and uses that encoding as context when decoding/predicting the previous and next sentences (see Fig. 7). Our modifications to this method include adding start- and end-of-recipe “instructions” and using an LSTM instead of a GRU. In either case, the representation of a single instruction is the final output of the encoder. As before, this is used as the instructions input to our embedding model.

3.2 Representation of Food Images

For the image representation we adopt two major state-of-the-art deep convolutional networks, namely VGG-16 [6] and Resnet-50 [7] models. In particular, the deep residual networks have a proven record of success on a variety of benchmarks [7]. Although [6] suggests training very deep networks with small convolutional filters, deep residual networks take it to another level using ubiquitous identity mappings that enable training of much deeper architectures (e.g., with 50, 101, or 152 layers) with better performance. We incorporate these models by removing the last softmax classification layer and connecting the rest to our joint embedding model as shown in the right side of Fig. 6.

4 Joint Neural Embedding

Building upon the previously described recipe and image representations, we now introduce our joint embedding method. The recipe model, displayed in Fig. 6, includes two encoders: one for ingredients and one for instructions, the combination of which are designed to learn a recipe level representation. The ingredients encoder combines the sequence of ingredient word vectors. Since the ingredient list is an unordered set, we choose to utilize a bidirectional LSTM model, which considers both forward and backward orderings. The instructions encoder is implemented as a forward LSTM model over skip-instructions vectors. The outputs of both encoders are concatenated and embedded into a recipe-image joint space. The image representation is simply projected into this space through a linear transformation. The goal is to learn transformations to make the embeddings for a given recipe-image pair “close.”

Formally, assume that we are given a set of the recipe-image pairs, (r_k, v_k) in which r_k is the k^{th} recipe and v_k is the associated image. Further, let $r_k = (\{s_k^t\}_{t=1}^{n_k}, \{g_k^t\}_{t=1}^{m_k})$, where $\{s_k^t\}_{t=1}^{n_k}$ is the sequence of n_k cooking instructions, $\{g_k^t\}_{t=1}^{m_k}$ is the sequence of m_k ingredient tokens. The objective is to maximize the cosine similarity between positive recipe-image pairs, and minimize it between all non-matching recipe-image pairs, up to a specified margin.

The ingredients encoder is implemented using a bi-directional LSTM: at each time step it takes two ingredient-word2vec representations of g_k^t and $g_k^{m_k-t+1}$, and eventually, it produces the fixed-length representation h_k^g for ingredients. The instructions encoder is implemented through a regular LSTM. At each time step it receives an instruction representation from the skip-instructions encoder, and finally it produces the fixed-length representation h_k^s. h_k^g and h_k^s are concatenated in order to obtain the recipe representation h_k^r. On the image side, the image encoder simply produces the fixed-length representation h_k^v. Then, the recipe and image representations are mapped into the joint embedding space as: $\phi^r = W^r h_k^r + b^r$ and $\phi^v = W^v h_k^v + b^v$, respectively. Note that W^r and W^v are embedding matrices which are also learned. Finally, the complete model is trained end-to-end with positive and negative recipe-image pairs (ϕ^r, ϕ^v) using the cosine similarity loss with margin defined as follows:

$$L_{cos}(\phi^r, \phi^v, y) = \begin{cases} 1 - \cos(\phi^r, \phi^v), & \text{if } y = 1 \\ \max(0, \cos(\phi^r, \phi^v) - \alpha), & \text{if } y = -1 \end{cases}$$

where $\cos(.)$ is the normalized cosine similarity and α is the margin.

5 Semantic Regularization

We incorporate additional regularization on our embedding through solving the same high-level classification problem in multiple

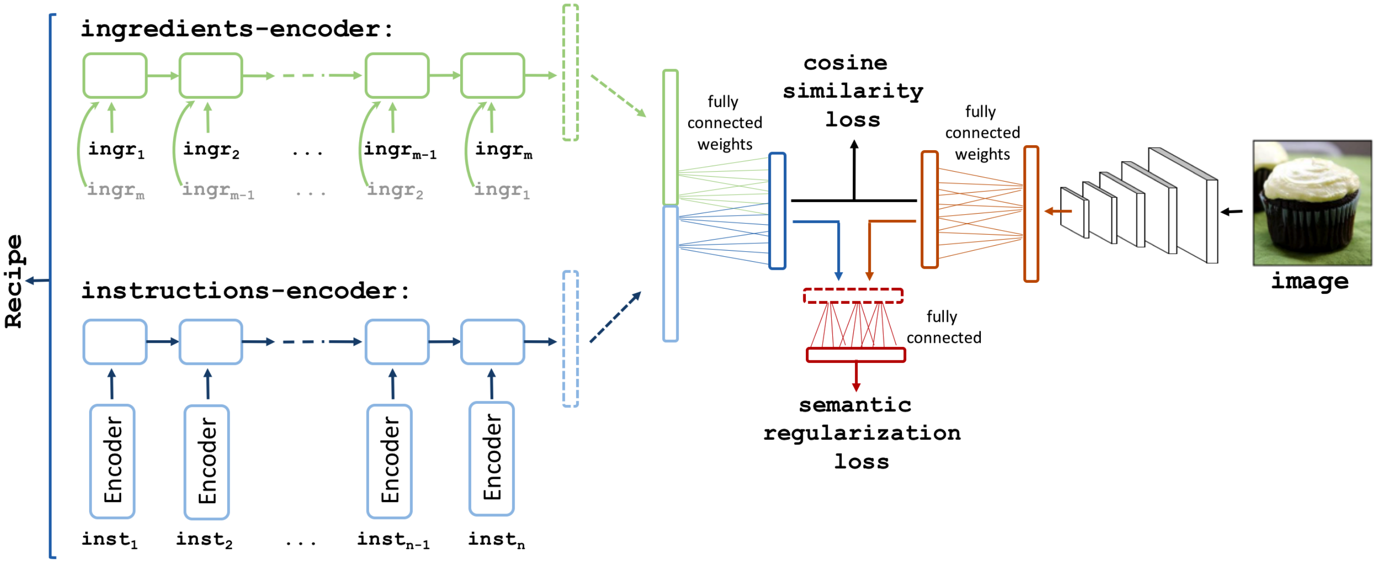
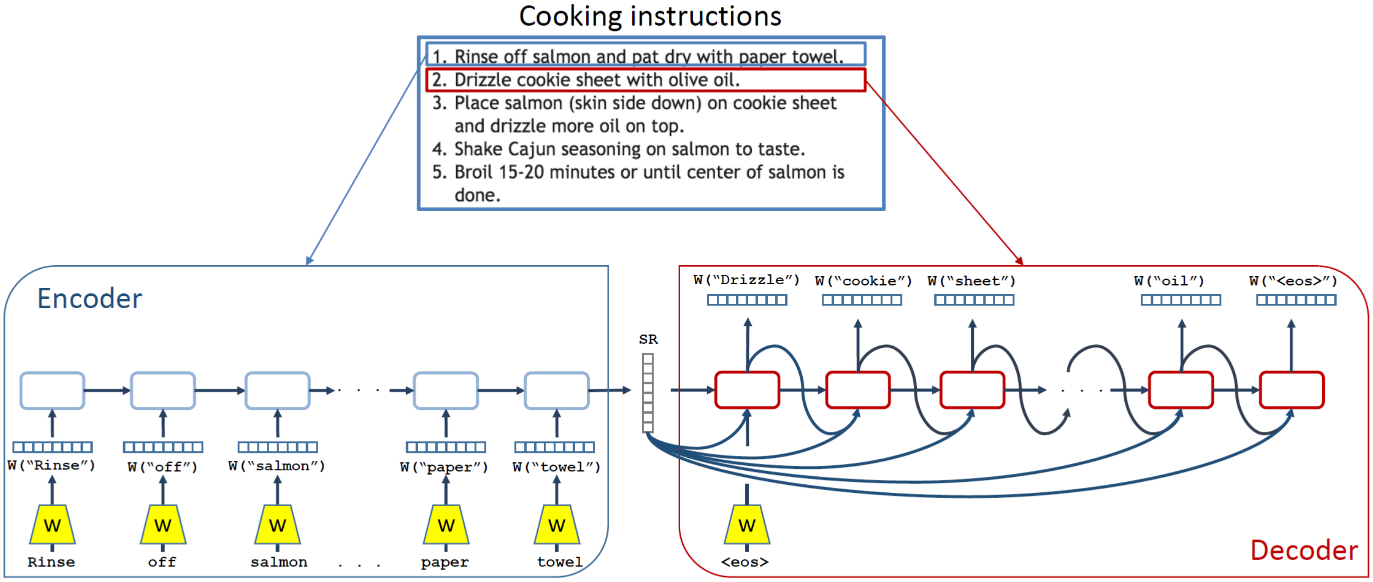


Fig. 6. **Joint neural embedding model with semantic regularization.** Our model learns a joint embedding space for food images and cooking recipes.



1. Rinse off salmon and pat dry with paper towel.
2. Drizzle cookie sheet with olive oil.
3. Place salmon (skin side down) on cookie sheet and drizzle more oil on top.
4. Shake Cajun seasoning on salmon to taste.
5. Broil 15-20 minutes or until center of salmon is done.

Fig. 7. **Skip-instructions model.** During training the encoder learns to predict the next instruction.

modalities with shared high-level weights. We refer to this method as semantic regularization. The key idea is that if high-level discriminative weights are shared, then both of the modalities (recipe and image embeddings) should utilize these weights in a similar way which brings another level of alignment based on discrimination. We optimize this objective together with our joint embedding loss. Essentially the model also learns to classify any image or recipe embedding into one of the food-related semantic categories. We limit the effect of semantic regularization as it is not the main problem that we aim to solve.

Semantic Categories. We start by assigning Food-101 categories to those recipes that contain them in their title. However, after this procedure we are only able to annotate 13% of our dataset, which we argue is not enough labeled data for a good regularization. Hence, we compose a larger set of semantic categories purely extracted from recipe titles. We first obtain the top 2,000 most frequent bigrams in recipe titles from our training set. We manually remove those that contain unwanted characters (e.g., *n'*, *!*, *?* or *&*) and those that do not have discriminative food properties (e.g., *best pizza*, *super easy* or *5 minutes*). We then assign each of the remaining bigrams as the semantic category to all recipes that include it in their title. By using bigrams and Food-101 categories together we obtain a total of 1,047 categories, which cover 50% of the dataset. *chicken salad*, *grilled vegetable*, *chocolate cake* and *fried fish* are some examples among the categories we collect using this procedure. All those recipes without a semantic category are assigned to an additional *background* class. Although there is some overlap in the generated categories, 73% of the recipes in our dataset (excluding those in the $background$ class) belong to a single category (i.e., only one of the generated classes appears in their title). For recipes where two or more categories appear in the title, the category with highest frequency rate in the dataset is chosen.

Classification. To incorporate semantic regularization to the joint embedding, we use a single fully connected layer. Given the embeddings ϕ^v and ϕ^r, class probabilities are obtained with $p_r = W^c \phi^r$ and $p_v = W^c \phi^v$ followed by a softmax activation. W^c is the matrix of learned weights, which are shared between image and recipe embeddings to promote semantic alignment between them. Formally, we express the semantic regularization loss as $L_{reg}(\phi^r, \phi^v, c_r, c_v)$ where c_r and c_v are the semantic category labels for recipe and image, respectively. Note that c_r and c_v are the same if (ϕ^r, ϕ^v) is a positive pair. Then, we can write the final objective as:

$$\begin{aligned} L(\phi^r, \phi^v, c_r, c_v, y) = \quad & L_{cos}(\phi^r, \phi^v, y) + \\ & \lambda L_{reg}(\phi^r, \phi^v, c_r, c_v) \end{aligned}$$

Optimization. We follow a two-stage optimization procedure while learning the model. If we update both the recipe encoding and image network at the same time, optimization becomes oscillatory and even divergent. Previous work on cross-modality training [31], [32] suggests training models for different modalities separately and fine tuning them jointly afterwards to allow alignment. Following this insight, we adopt a similar procedure when training our model. We first fix the weights of the image network, which are found from pre-training on the ImageNet object classification task, and learn the recipe encodings. This way the recipe network learns to align itself to the image representations and also learns semantic regularization parameters (W^c). Then we freeze the recipe encoding and semantic regularization weights, and learn the image network. This two-stage process is crucial for successful optimization of the objective function. After this initial alignment stage, we release all the weights to be learned. However, the results do not change much in this final, joint optimization. We take a step further from [12] in our extended study and change the validation procedure to use median rank (MedR) score as our performance measure, like in [19], while reimplementing our source code in PyTorch. This strategy appears to be more stable than using the validation loss.

Implementation Details. All the neural network models are implemented using Torch7[7] and PyTorch[8] frameworks. The margin α is selected as 0.1 in joint neural embedding models. The regularization hyper-parameter is set as $\lambda = 0.02$ in all our experiments. While optimizing the cosine loss, we pick a positive recipe-image pairs with 20% probability and a random negative recipe-image pair with 80% probability from the training set.

The models in Torch7 are trained on 4 NVIDIA Titan X with 12GB of memory for three days. The models in PyTorch are trained on 4 NVIDIA GTX 1080 with 8GB of memory for two and a half days (using a bigger batch size, i.e., 256 pairs instead of 150). When using Recipe1M+, the training in PyTorch tends to take over a week, using a batch size of 256. For efficiency purposes, we store the recipe text part of the dataset in LMDB[9] format and load the images on the fly using `DataLoader` function of the PyTorch library. This way our PyTorch code does not require as much RAM as our Torch7 code does. As a side note, between the two reference libraries, we did experience that PyTorch in general uses less GPU memory.

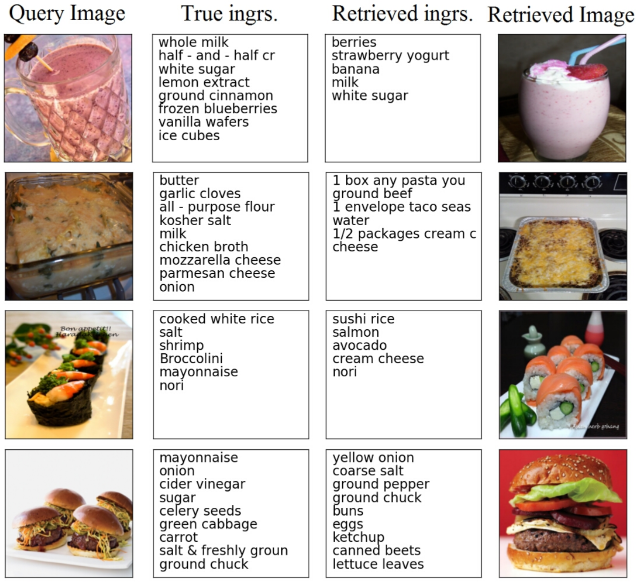


Fig. 8. **Im2recipe retrieval examples.** From left to right: (1) the query image, (2) its associated ingredient list, (3) the retrieved ingredients, and (4) the image associated to the retrieved recipe.

6 Experiments

We begin with the evaluation of our learned embeddings for the im2recipe retrieval task on the initial (i.e., recipe-website-only) version of our dataset (i.e., Recipe1M). Specifically, we study the effect of each component of our model and compare our final system against human performance for the im2recipe retrieval task. Then, using the best model architecture trained on the recipe-website-only version of the dataset, we compare its retrieval performance with the same one trained on the extended version of the dataset (i.e., Recipe1M+) to evaluate the benefit of data extension through an image search engine. We further evaluate the two models on Food-101 dataset to assess their generalization ability. Finally, we analyze the properties of our learned embeddings through unit visualizations and explore different vector arithmetics in the embedding space on both the initial (Recipe1M) and the extended (Recipe1M+) datasets.

6.1 Im2recipe Retrieval

We evaluate all the recipe representations for im2recipe retrieval. Given a food image, the task is to retrieve its recipe from a collection of test recipes. We also perform recipe2im retrieval using the same setting. All results are reported for the test set.

Comparison with the Baselines. Canonical Correlation Analysis (CCA) is one of the strongest statistical models for learning joint embeddings for different feature spaces when paired data are provided. We use CCA over many high-level recipe and image representations as our baseline. These CCA embeddings are learned using recipe-image pairs from the training data. In each recipe, the ingredients are represented with the mean word2vec across all its ingredients in the manner of [33]. The cooking instructions are represented with mean skip-thoughts vectors [30] across the cooking instructions. A recipe is then represented as concatenation of these two features. We also evaluate CCA over mean ingredient word2vec and skip-instructions features as another baseline. The

[7] http://torch.ch/
[8] https://pytorch.org/
[9] https://lmdb.readthedocs.io/en/release/

TABLE 3
Im2recipe retrieval comparisons on Recipe1M. Median ranks and recall rate at top K are reported for baselines and our method. Note that the joint neural embedding models consistently outperform all the baseline methods.

	im2recipe				recipe2im			
	medR	R@1	R@5	R@10	medR	R@1	R@5	R@10
random ranking	500	0.001	0.005	0.01	500	0.001	0.005	0.01
CCA w/ skip-thoughts + word2vec (GoogleNews) + image features	25.2	0.11	0.26	0.35	37.0	0.07	0.20	0.29
CCA w/ skip-instructions + ingredient word2vec + image features	15.7	0.14	0.32	0.43	24.8	0.09	0.24	0.35
joint emb. only	7.2	0.20	0.45	0.58	6.9	0.20	0.46	0.58
joint emb. + semantic	5.2	0.24	0.51	0.65	5.1	0.25	0.52	0.65
attention + SR. [18]	4.6	0.26	0.54	0.67	4.6	0.26	0.54	0.67
AdaMine [19]	1.0	0.40	0.69	0.77	1.0	0.40	0.68	0.79

image features utilized in the CCA baselines are the ResNet-50 features before the softmax layer. Although they are learned for visual object categorization tasks on ImageNet dataset, these features are widely adopted by the computer vision community, and they have been shown to generalize well to different visual recognition tasks [34].

For evaluation, given a test query image, we use cosine similarity in the common space for ranking the relevant recipes and perform im2recipe retrieval. The recipe2im retrieval setting is evaluated likewise. We adopt the test procedure from image2caption retrieval task [35], [36]. We report results on a subset of randomly selected 1,000 recipe-image pairs from the test set. We repeat the experiments 10 times and report the mean results. We report median rank (MedR), and recall rate at top K (R@K) for all the retrieval experiments. To clarify, R@5 in the im2recipe task represents the percentage of all the image queries where the corresponding recipe is retrieved in the top 5, hence higher is better. The quantitative results for im2recipe retrieval are shown in Table 3.

Our model outperforms the CCA baselines in all measures. As expected, CCA over ingredient word2vec and skip-instructions perform better than CCA over word2vec trained on GoogleNews [28] and skip-thoughts vectors that are learned over a large-scale book corpus [30]. In 65% of all evaluated queries, our method can retrieve the correct recipe given a food image. The semantic regularization notably improves the quality of our embedding for im2recipe task which is quantified with the medR drop from 7.2 to 5.2 in Table 3. The results for recipe2im task are also similar to those in the im2recipe retrieval setting.

Table 3 also presents results originally reported in [18] and [19] on Recipe1M. Attention-based modeling of [18] achieves slight performance increases whereas double-triplet learning scheme of [19] leads to larger performance gains in both retrieval settings.

Fig. 8 compares the ingredients from the original recipes (true recipes) with the retrieved recipes (coupled with their corresponding image) for different image queries. As can be observed in Fig. 8, our embeddings generalize well and allow overall satisfactory recipe retrieval results. However, at the ingredient level, one can find that in some cases our model retrieves recipes with missing ingredients. This usually occurs due to the lack of fine-grained features (e.g., confusion between *shrimps* and *salmon*) or simply because the ingredients are not visible in the query image (e.g., *blueberries* in a *smoothie* or *beef* in a *lasagna*).

Ablation Studies. We also analyze the effect of each component in our our model in several optimization stages. The results are reported in Table 4. Note that here we also report medR with $1K$, $5K$ and $10K$ random selections to show how the results scale in larger retrieval problems. As expected, visual features from the ResNet-50 model show a substantial improvement in retrieval performance when compared to VGG-16 features. Even with "fixed vision" networks the joint embedding achieved 7.9 medR using ResNet-50 architecture. Further "fine-tuning" of vision networks slightly improves the results. Although it becomes a lot harder to decrease the medR in small numbers, additional "semantic regularization" improves the medR in both cases.

Comparison with Human Performance. In order to better assess the quality of our embeddings we also evaluate the performance of humans on the im2recipe task. The experiments are performed through AMT. For quality purposes, we require each AMT worker to have at least 97% approval rate and have performed at least 500 tasks before our experiment. In a single evaluation batch, we first randomly choose 10 recipes and their corresponding images. We then ask an AMT worker to choose the correct recipe, out of the 10 provided recipes, for the given food image. This multiple choice selection task is performed 10 times for each food image in the batch. The accuracy of an evaluation batch is defined as the percentage of image queries correctly assigned to their corresponding recipe.

The evaluations are performed for three levels of difficulty. The batches (of 10 recipes) are randomly chosen from either all the test recipes (easy), recipes sharing the same course (e.g., soup, salad, or beverage; medium), or recipes sharing the name of the dish (e.g., salmon, pizza, or ravioli; hard). As expected–for our model as well as the AMT workers–the accuracies decrease as tasks become more specific. In both coarse and fine-grained tests, our method performs comparably to or better than the AMT workers. As hypothesized, semantic regularization further improves the results (see Table 5).

In the "all recipes" condition, 25 random evaluation batches (25×10 individual tasks in total) are selected from the entire test set. Joint embedding with semantic regularization performs the best with 3.2 percentage points improvement over average human accuracy. For the course-specific tests, 5 batches are randomly selected within each given meal course. Although, on average, our joint embedding's performance is slightly lower than the humans', with semantic regularization our joint embedding surpasses humans' performance by 6.8 percentage points. In dish-specific tests, five random batches are selected if they have the dish name (e.g., pizza) in their title. With slightly lower accuracies in general, dish-specific results also show similar behavior. Particularly for the "beverage" and "smoothie" results, human performance is better than our method, possibly because detailed analysis is needed to elicit the homogenized ingredients in drinks. Similar behavior is also observed for the "sushi" results where fine-grained features

TABLE 4
Ablation studies on Recipe1M. Effect of the different model components to the median rank, medR (the lower is better).

	Joint emb. methods	im2recipe			recipe2im		
		medR-1K	medR-5K	medR-10K	medR-1K	medR-5K	medR-10K
VGG-16	fixed vision	15.3	71.8	143.6	16.4	76.8	152.8
	finetuning (ft)	12.1	56.1	111.4	10.5	51.0	101.4
	ft + semantic reg.	8.2	36.4	72.4	7.3	33.4	64.9
ResNet-50	fixed vision	7.9	35.7	71.2	9.3	41.9	83.1
	finetuning (ft)	7.2	31.5	62.8	6.9	29.8	58.8
	ft + semantic reg.	5.2	21.2	41.9	5.1	20.2	39.2

TABLE 5
Comparison with human performance on im2recipe task on Recipe1M. The mean results are highlighted as bold for better visualization. Note that on average our method with semantic regularization performs better than average AMT worker.

	all recipes	course-specific recipes						dish-specific recipes								
		dessert	salad	bread	beverage	soup-stew	course-mean	pasta	pizza	steak	salmon	smoothie	hamburger	ravioli	sushi	dish-mean
human	**81.6 ± 8.9**	52.0	70.0	34.0	58.0	56.0	**54.0 ± 13.0**	54.0	48.0	58.0	52.0	48.0	46.0	54.0	58.0	**52.2 ± 04.6**
joint-emb. only	**83.6 ± 3.0**	76.0	68.0	38.0	24.0	62.0	**53.6 ± 21.8**	58.0	58.0	58.0	64.0	38.0	58.0	62.0	42.0	**54.8 ± 09.4**
joint-emb.+semantic	**84.8 ± 2.7**	74.0	82.0	56.0	30.0	62.0	**60.8 ± 20.0**	52.0	60.0	62.0	68.0	42.0	68.0	62.0	44.0	**57.2 ± 10.1**

TABLE 6
Comparison between models trained on Recipe1M vs. Recipe1M+. Median ranks and recall rate at top K are reported for both models. They have similar performance on the Recipe1M test set in terms of medR and R@K. However, when testing on the Recipe1M+ test set, the model trained on Recipe1M+ yields significantly better medR and better R@5 and R@10 scores. In this table, Recipe1M refers to the *intersection* dataset.

	Recipe1M test set				Recipe1M+ test set			
	im2recipe							
	medR	R@1	R@5	R@10	medR	R@1	R@5	R@10
Recipe1M training set	5.1	0.24	0.52	0.64	13.6	0.15	0.35	0.46
Recipe1M+ training set	5.7	0.21	0.49	0.62	8.6	0.17	0.42	0.54
	recipe2im							
	medR	R@1	R@5	R@10	medR	R@1	R@5	R@10
Recipe1M training set	4.8	0.27	0.54	0.65	11.9	0.17	0.38	0.48
Recipe1M+ training set	4.6	0.26	0.54	0.66	6.8	0.21	0.46	0.58

of the sushi roll's center are crucial to identify the correct sushi recipe.

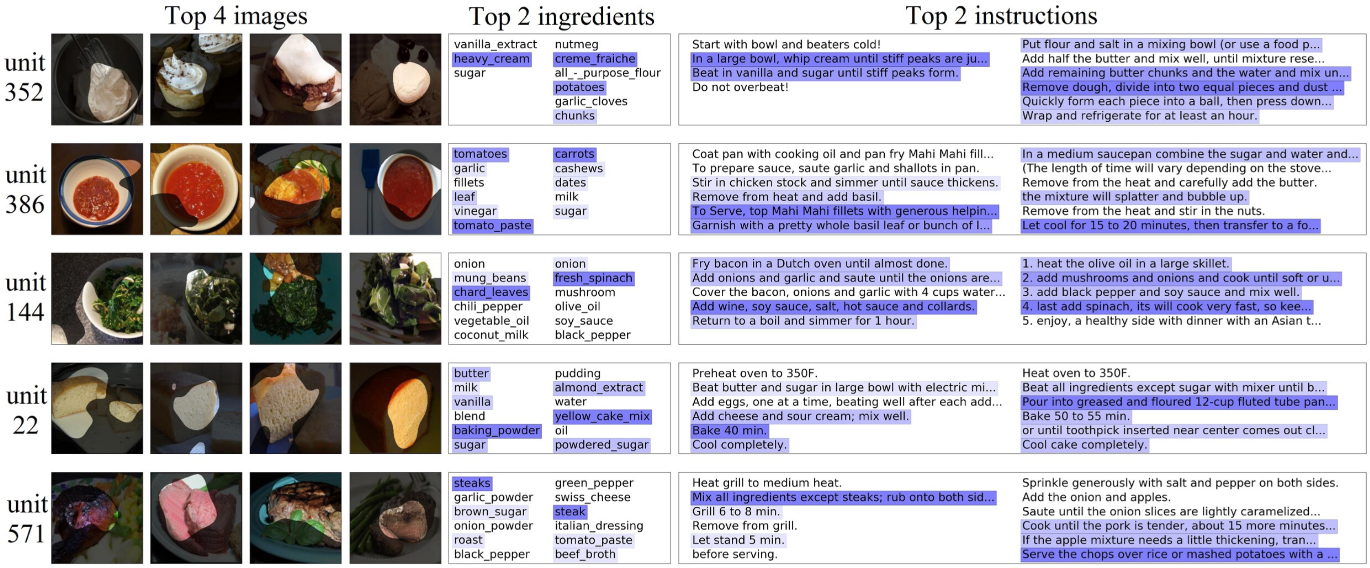
Recipe1M vs. Recipe1M+ Comparison. One of the main questions of the current study is how beneficial it is to incorporate images coming from a Web search engine into the initial collection of images obtained from recipe websites. One way to assess this is to compare im2recipe retrieval performance of a network architecture trained on Recipe1M with im2recipe retrieval performance of the same network architecture trained on Recipe1M+. In Table 6, we present im2recipe retrieval results achieved on both test sets. As can be seen, there is a clear benefit when we evaluate both models on the Recipe1M+ test set. The model trained on Recipe1M+ obtains a significantly better medR, 5 points lower in both retrieval tasks, and higher R@5 and R@10, in some cases up to a 10 percentage point increase. When looking into the Recipe1M test set, both models perform similarly. These results clearly demonstrate the benefit of using external search engines to extend the imagery content of Recipe1M. Note that the retrieval results on Tables 3 and 6 slightly differ due to the fact that we use a modified version of the dataset (see *intersection* dataset in Table 1) in the latter experiment. As we explained earlier in Section 2, this is done mainly to have a fair comparison of im2recipe retrieval results on both versions of the dataset.

TABLE 7
Im2recipe retrieval comparisons on Food-101 dataset. Median ranks and recall rate at top K are reported for both models. Note that the model trained on Recipe1M+ performs better than the model trained on Recipe1M. In this table, Recipe1M refers to the *intersection* dataset.

	im2recipe			
	medR	R@1	R@5	R@10
Recipe1M training set	17.35	16.13	33.68	42.53
Recipe1M+ training set	10.15	21.89	42.31	51.14
	recipe2im			
Recipe1M training set	4.75	26.19	54.52	67.50
Recipe1M+ training set	2.60	37.38	65.00	76.31

Model Generalization Ability Comparison. We experiment further to evaluate whether Recipe1M+ dataset improves the performance of our model on other food image datasets. For this purpose, we evaluate both our trained models on the popular Food-101 dataset [8]. The Food-101 dataset is a classification dataset containing 101 food categories and 1,000 images for each one of these 101 food categories, totaling up to 101,000 images.

Our method of evaluation involves randomly sampling an image



Start with bowl and beaters cold!
In a large bowl, whip cream until stiff peaks are ju...
Beat in vanilla and sugar until stiff peaks form.
Do not overbeat!

Put flour and salt in a mixing bowl (or use a food p...
Add half the butter and mix well, until mixture rese...
Add remaining butter chunks and the water and mix un...
Remove dough, divide into two equal pieces and dust ...
Quickly form each piece into a ball, then press down...
Wrap and refrigerate for at least an hour.

Coat pan with cooking oil and pan fry Mahi Mahi fill...
To prepare sauce, saute garlic and shallots in pan.
Stir in chicken stock and simmer until sauce thickens.
Remove from heat and add basil.
To Serve, top Mahi Mahi fillets with generous helpin...
Garnish with a pretty whole basil leaf or bunch of l...

In a medium saucepan combine the sugar and water and...
(The length of time will vary depending on the stove...
Remove from the heat and carefully add the butter.
the mixture will splatter and bubble up.
Remove from the heat and stir in the nuts.
Let cool for 15 to 20 minutes, then transfer to a fo...

Fry bacon in a Dutch oven until almost done.
Add onions and garlic and saute until the onions are...
Cover the bacon, onions and garlic with 4 cups water...
Add wine, soy sauce, salt, hot sauce and collards.
Return to a boil and simmer for 1 hour.

1. heat the olive oil in a large skillet.
2. add mushrooms and onions and cook until soft or u...
3. add black pepper and soy sauce and mix well.
4. last add spinach, its will cook very fast, so kee...
5. enjoy, a healthy side with dinner with an Asian t...

Preheat oven to 350F.
Beat butter and sugar in large bowl with electric mi...
Add eggs, one at a time, beating well after each add...
Add cheese and sour cream; mix well.
Bake 40 min.
Cool completely.

Heat oven to 350F.
Beat all ingredients except sugar with mixer until b...
Pour into greased and floured 12-cup fluted tube pan...
Bake 50 to 55 min.
or until toothpick inserted near center comes out cl...
Cool cake completely.

Heat grill to medium heat.
Mix all ingredients except steaks; rub onto both sid...
Grill 6 to 8 min.
Remove from grill.
Let stand 5 min.
before serving.

Sprinkle generously with salt and pepper on both sides.
Add the onion and apples.
Saute until the onion slices are lightly caramelized...
Cook until the pork is tender, about 15 more minutes...
If the apple mixture needs a little thickening, tran...
Serve the chops over rice or mashed potatoes with a ...

Fig. 9. **Localized unit activations.** We find that ingredient detectors emerge in different units in our embeddings, which are aligned across modalities (e.g., unit 352: “cream”, unit 22: “sponge cake” or unit 571: “steak”).

and a recipe corresponding to each of the Food-101 categories. The images are taken from the Food-101 dataset, while the recipes are taken from the test partition of the *intersection* dataset. Here, a recipe is considered to belong to a category only if the recipe title string matches with the Food-101 category. Here, we only sample images and recipes from those categories that correspond to at least N recipes among the test recipes that we sample from.

After sampling an image and a corresponding recipe for each category that is common enough, we evaluate our models on the retrieval task. In the im2recipe direction, we provide our model with the image and expect it to retrieve the corresponding recipe. In the recipe2im direction, we provide our model with the recipe and expect it to retrieve the corresponding image. We show the retrieval results of both models in Table 7. Note that the model trained on Recipe1M+ consistently outperforms the model trained on Recipe1M.

One possible explanation for Recipe1M+ dataset giving an advantage on the Food-101 task is that there might be an overlap between the images used to train the model on the Recipe1M+ and the Food-101 images. Further, it is possible that there might be images in Recipe1M+ training set that overlap with the Food-101 dataset that are not in the initial training set. This would give the model trained on Recipe1M+ an unfair advantage. We perform the following procedure to test whether this is true. First, we feed in all of the images in the Recipe1M+ training set and the Food-101 images into an 18 layer residual network that was pre-trained on ImageNet. The network outputs a prediction vector for each of these images. We next note that if an image in the extended training set has an exact copy in the Food-101 dataset, then both images must have the same prediction vector. When checking the prediction vectors of the images in Food-101 and the Recipe1M+ training set, we did not find any overlapping prediction vectors, meaning that the images between Food-101 and Recipe1M+ training set do not overlap.

6.2 Analysis of the Learned Embedding

To gain further insight into our neural embedding, we perform a series of qualitative analysis experiments. We explore whether any semantic concepts emerge in the neuron activations and whether the embedding space has certain arithmetic properties.

Neuron Visualizations. Through neural activation visualization, we investigate if any semantic concepts emerge in the neurons in our embedding vector despite not being explicitly trained for that purpose. We pick the top activating images, ingredient lists, and cooking instructions for a given neuron. Then we use the methodology introduced by Zhou et al. [37] to visualize image regions that contribute the most to the activation of specific units in our learned visual embeddings. We apply the same procedure on the recipe side to also obtain those ingredients and recipe instructions to which certain units react the most. Fig. 9 shows the results for the same unit in both the image and recipe embedding. We find that certain units display localized semantic alignment between the embeddings of the two modalities.

Semantic Vector Arithmetic. Different works in the literature [28], [38] have used simple arithmetic operations to demonstrate the capabilities of their learned representations. In the context of food recipes, one would expect that v(“chicken pizza”) $-$ v(“pizza”) $+$ v(“salad”) $=$ v(“chicken salad”), where v represents the map into the embedding space. We demonstrate that our learned embeddings have such properties by applying the previous equation template to the averaged vectors of recipes that contain the queried words in their title. We apply this procedure in the recipe and image embedding spaces and show results in Fig. 10 and Fig. 11, respectively. Our findings suggest that the learned embeddings have semantic properties that translate to simple geometric transformations in the learned space. Furthermore, the model trained on Recipe1M+ is better able to capture these semantic properties in the embedding space. The improvement is most seriously observable on the recipe arithmetic. Among the recipe analogy examples, notice that the result for the Recipe1M+ dataset for “chicken quesadilla” - “wrap” + “rice” returns a casserole dish, while for the Recipe1M dataset we have a quesadilla

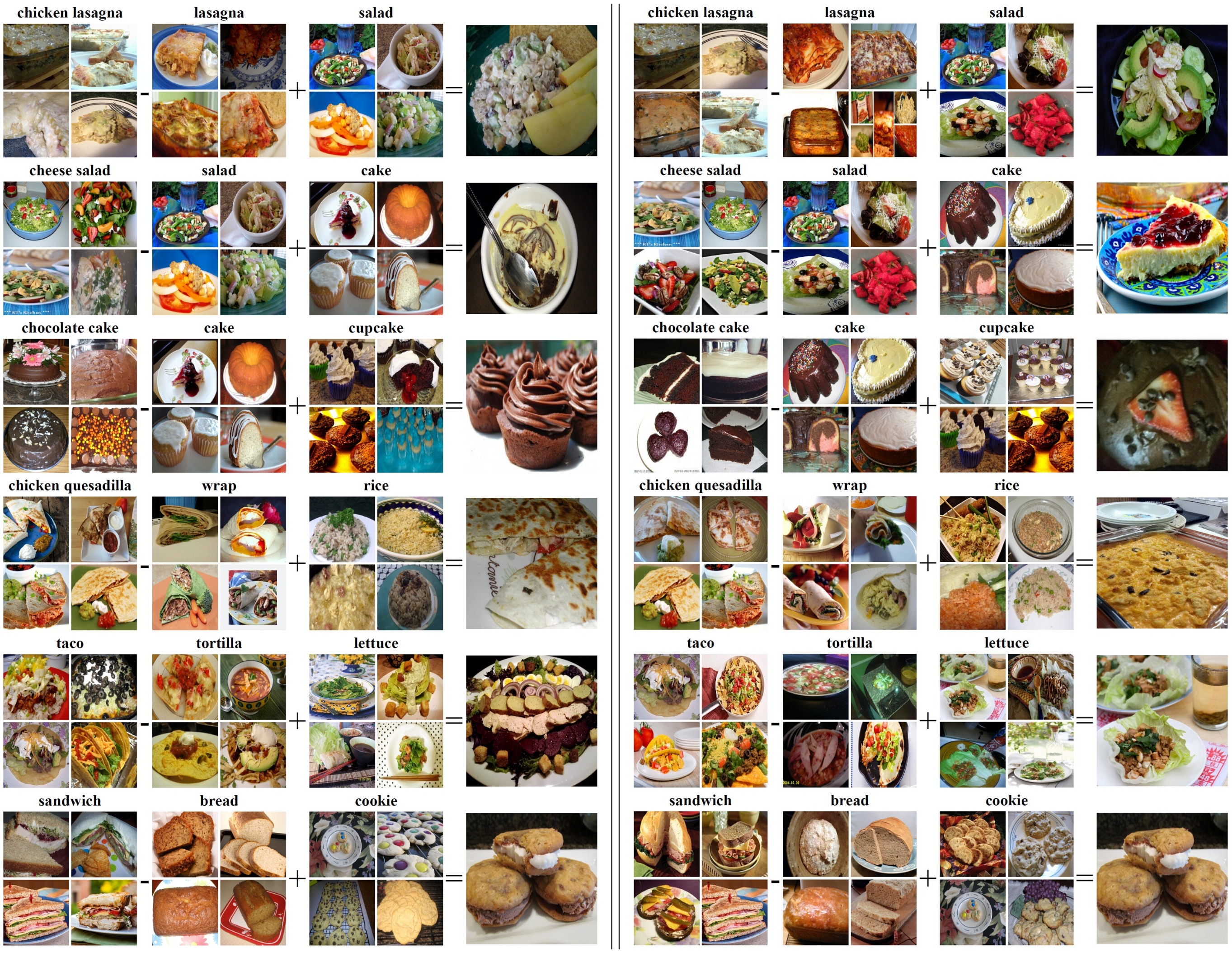


Fig. 10. **Analogy arithmetic results using recipe embeddings** on the Recipe1M test set. On the left hand side are arithmetic results using the model trained on Recipe1M. On the right hand side are the arithmetic results for the model trained on Recipe1M+. We represent the average vector of a query with the images from its 4 nearest neighbors. In the case of the arithmetic result, we show the nearest neighbor only.

dish. The casserole dish is much closer to matching the “chicken rice” result that we expect in this instance. Additionally, note how “taco” - “tortilla” + “lettuce” returns a salad for the Recipe1M model and a lettuce wrap for the Recipe1M+ model. Here, the former model is likely doing arithmetic over the ingredients in the dish - a taco without tortilla likely comprises of a salad, into which lettuce is added to give a salad-like dish. On the other hand, the Recipe1M+ model does arithmetic over higher level semantic concepts - it returns a lettuce wrap, which is the closest analogue to a taco which has the tortilla substituted out with lettuce. We can thus see how the Recipe1M+ model has a greater ability to capture semantic concepts in the recipe embedding space, and also performs somewhat better in general. If we examine the results of both models for the analogy task with image embeddings, then the Recipe1M+ model shows less of an improvement in general. However, we can still see differences between the two models. For instance, if we examine the “taco” - “tortilla” + “lettuce” analogy, then the Recipe1M model returns a result in which the lettuce is mixed in with other ingredients to form a salad. However, the Recipe1M+ model returns a result in which a salad is placed on top of a large piece of lettuce. This result is similar in a way to the lettuce wrap result, as the piece of lettuce is not just mixed in with the other ingredients, but acts as more of an object into which other ingredients are placed. All in all, the Recipe1M+ training set allows our model to better capture high level semantic concepts.

Fractional Arithmetic. Another type of arithmetic we examine is fractional arithmetic, in which our model interpolates across the vector representations of two concepts in the embedding space. Specifically, we examine the results for $x \times v$(“concept 1”) $+ (1 - x) \times v$(“concept 2”), where x varies from 0 to 1. We expect this to have interesting applications in spanning the space across two food concepts, such as pasta and salad, by adjusting the value of x to make the dish more “pasta-like” or “salad-like” for example. We apply this procedure in the recipe and image embedding spaces and show results in Fig. 12 and Fig. 13, respectively. With both fractional image arithmetic and fractional recipe arithmetic, we hope that adjusting the fractional coefficient will allow us to explore more fine-grained combinations of two concepts. However, the results are often not so fine-grained. For

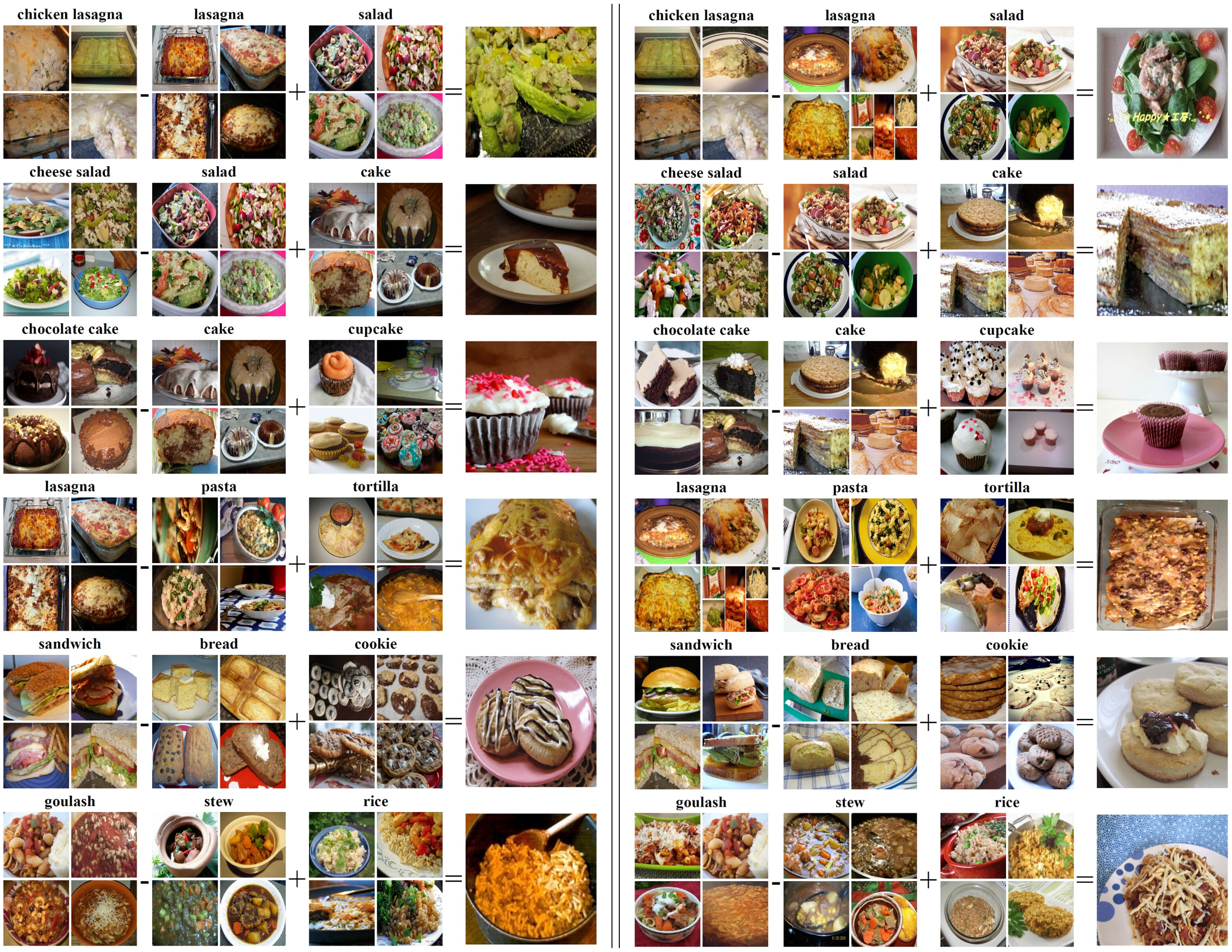


Fig. 11. **Analogy arithmetic results using image embeddings** on the Recipe1M test set. On the left hand side are arithmetic results using the model trained on Recipe1M. On the right hand side are the arithmetic results for the model trained on Recipe1M+. We represent the average vector of a query with the images from its four nearest neighbors. In the case of the arithmetic result, we show the nearest neighbor only.

instance, in the “burrito” and “sandwich” example for the model trained on the Recipe1M dataset for recipe fractional arithmetic, choosing a burrito coefficient of 0 does not yield different results from choosing the coefficient to be 0.5. Note that on the other hand, the model trained on the Recipe1M+ dataset is able to provide distinct results for each fractional coefficient value for this example. In general though, both models are able to effectively explore the gradient of recipes or food images between two different food concepts. For instance, note the models’ results for the “curry” and “soup” examples, in both the image and recipe modalities. The most “curry-like” image tends to have some broth, but is much chunkier than the images. As we increase the coefficient of “soup”, we see the food becoming less chunky and more broth-like. Such examples reflect the ability of our model to explore the space between food concepts in general.

The results of our fractional arithmetic experiments suggest that the recipe and image embeddings learned in our model are semantically aligned, which broaches the possibility of applications in recipe modification (e.g., ingredient replacement, calorie adjustment) or even cross-modal generation.

7 Conclusion

In this paper, we present Recipe1M+, the largest structured recipe dataset to date, the im2recipe problem, and neural embedding models with semantic regularization which achieve impressive results for the im2recipe task. The experiments conducted using AMT, together with the fact that on the Recipe1M test set we obtain the same test performance using Recipe1M+, show that the extended dataset is not much noisier. Moreover, the fact that this expansion strategy greatly helps on the Food 101 dataset demonstrates the value for generalizability. Additionally, we explored the properties of the resulting recipe and food representations by evaluating different vector arithmetics on the learned embeddings, which hinted at the possibility of applications such as recipe modification or even cross-modal recipe generation.

More generally, the methods presented here could be gainfully applied to other “recipes” like assembly instructions, tutorials, and industrial processes. Further, we hope that our contributions will support the creation of automated tools for food and recipe understanding and open doors for many less explored aspects of

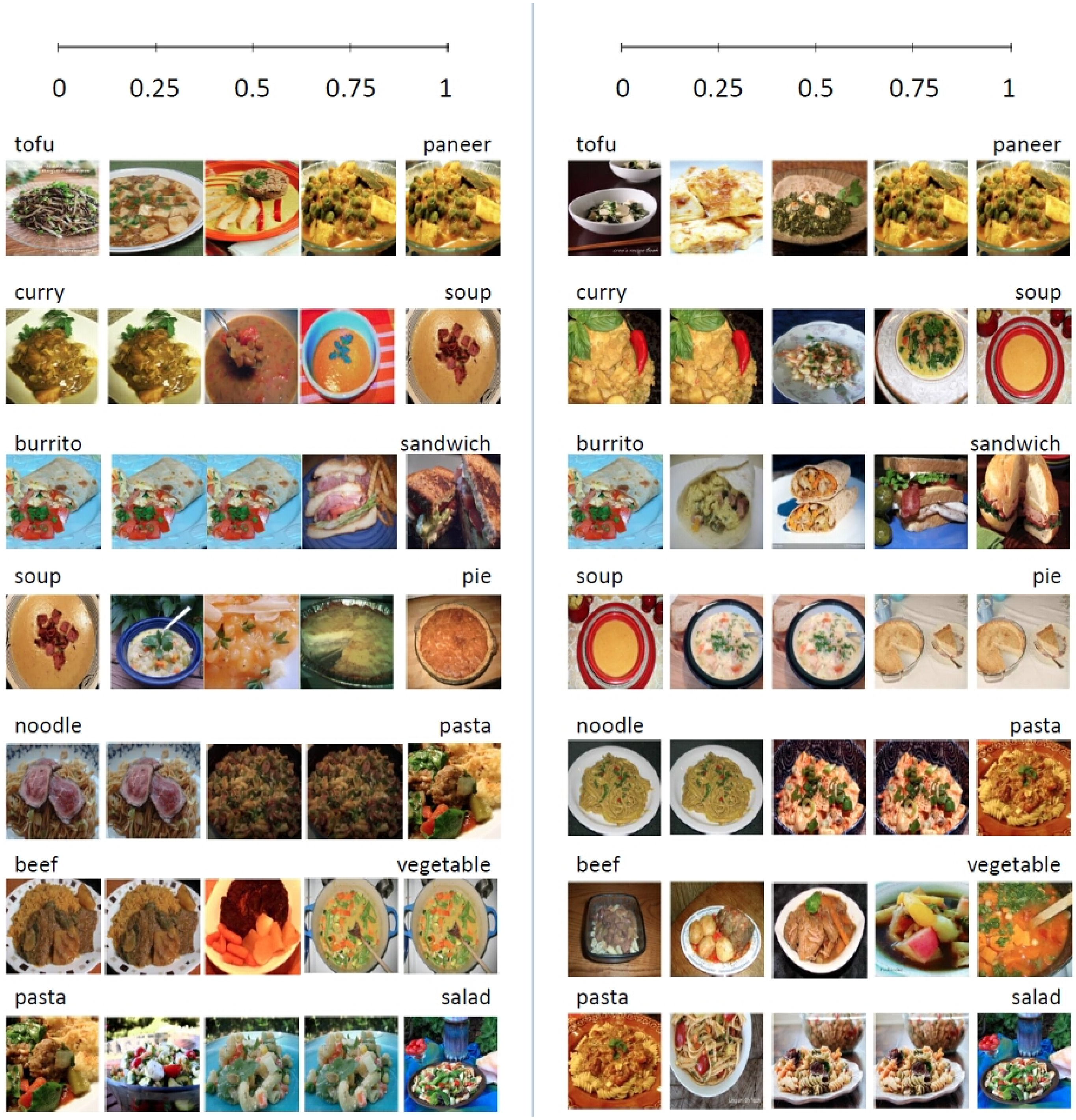


Fig. 12. **Fractional arithmetic results using recipe embeddings** on the Recipe1M test set. On the left hand side are arithmetic results using the model trained on Recipe1M. On the right hand side are the arithmetic results for the model trained on Recipe1M+. For each model, we fractionally interpolate across two example concepts (for instance, “salad” and “pasta”). We find the retrieved results for $x \times v$(“concept 1”) $+ (1-x) \times v$(“concept 2”), where x varies from 0 to 1.

learning such as compositional creativity and predicting visual outcomes of action sequences.

Acknowledgments

This work was supported by CSAIL-QCRI collaboration project.

References

[1] V. R. K. Garimella, A. Alfayad, and I. Weber, “Social media image analysis for public health,” in *CHI*, 2016, pp. 5543–5547.

[2] Y. Mejova, S. Abbar, and H. Haddadi, “Fetishizing food in digital age: #foodporn around the world,” in *ICWSM*, 2016, pp. 250–258.

[3] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, “Imagenet large scale visual

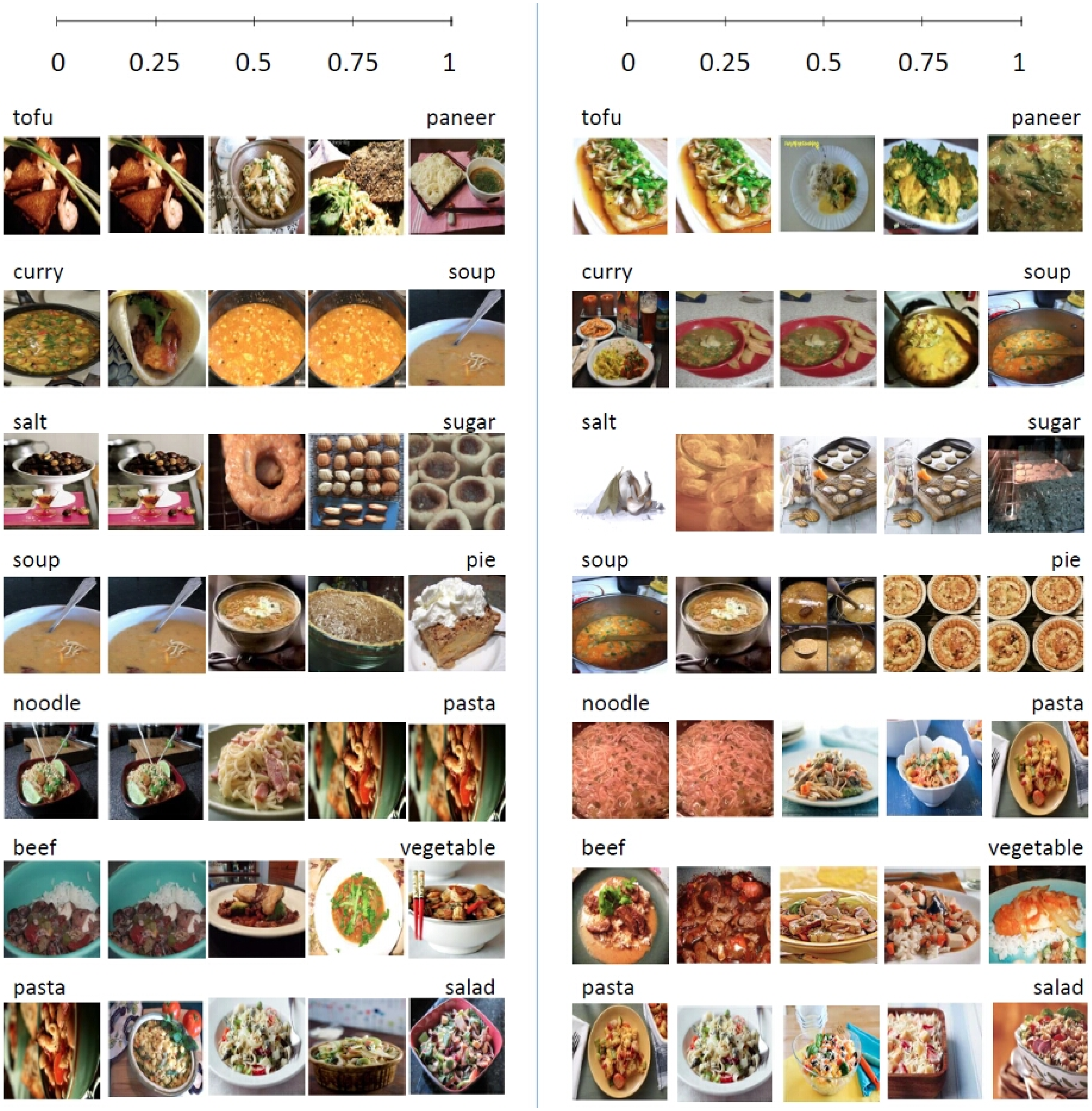


Fig. 13. **Fractional arithmetic results using image embeddings** on the Recipe1M test set. On the left hand side are arithmetic results using the model trained on Recipe1M. On the right hand side are the arithmetic results for the model trained on Recipe1M+. For each model, we fractionally interpolate across two example concepts (for instance, “salad” and “pasta”). We find the retrieved results for $x \times v$(“concept 1”) $+ (1 - x) \times v$(“concept 2”), where x varies from 0 to 1.

recognition challenge,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[4] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, “Learning deep features for scene recognition using places database,” in *Advances in neural information processing systems*, 2014, pp. 487–495.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *NIPS*, 2012.

[6] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[7] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *arXiv preprint arXiv:1512.03385*, 2015.

[8] L. Bossard, M. Guillaumin, and L. Van Gool, “Food-101–mining discriminative components with random forests,” in *European Conference on Computer Vision*. Springer, 2014, pp. 446–461.

[9] C. Liu, Y. Cao, Y. Luo, G. Chen, V. Vokkarane, and Y. Ma, “Deepfood: Deep learning-based food image recognition for computer-aided dietary assessment,” in *International Conference on Smart Homes and Health Telematics*. Springer, 2016, pp. 37–48.

[10] A. Myers, N. Johnston, V. Rathod, A. Korattikara, A. Gorban, N. Silberman, S. Guadarrama, G. Papandreou, J. Huang, and K. Murphy,

"Im2calories: Towards an automated mobile vision food diary," in *ICCV*, 2015, pp. 1233–1241.

[11] F. Ofli, Y. Aytar, I. Weber, R. Hammouri, and A. Torralba, "Is saki #delicious? the food perception gap on instagram and its relation to health," in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017.

[12] A. Salvador, N. Hynes, Y. Aytar, J. Marin, F. Ofli, I. Weber, and A. Torralba, "Learning cross-modal embeddings for cooking recipes and food images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, July 2017.

[13] L. Herranz, W. Min, and S. Jiang, "Food recognition and recipe analysis: integrating visual content, context and external knowledge," *CoRR*, vol. abs/1801.07239, 2018. [Online]. Available: http://arxiv.org/abs/1801.07239

[14] W. Min, S. Jiang, S. Wang, J. Sang, and S. Mei, "A delicious recipe analysis framework for exploring multi-modal recipes with various attributes," in *Proceedings of the 2017 ACM on Multimedia Conference*, ser. MM '17. New York, NY, USA: ACM, 2017, pp. 402–410. [Online]. Available: http://doi.acm.org/10.1145/3123266.3123272

[15] M. Chang, L. V. Guillain, H. Jung, V. M. Hare, J. Kim, and M. Agrawala, "Recipescape: An interactive tool for analyzing cooking instructions at scale," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, ser. CHI '18. New York, NY, USA: ACM, 2018, pp. 451:1–451:12. [Online]. Available: http://doi.acm.org/10.1145/3173574.3174025

[16] M. Engilberge, L. Chevallier, P. Pérez, and M. Cord, "Finding beans in burgers: Deep semantic-visual embedding with localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2018.

[17] J.-j. Chen, C.-W. Ngo, and T.-S. Chua, "Cross-modal recipe retrieval with rich food attributes," in *Proceedings of the 2017 ACM on Multimedia Conference*, ser. MM '17. New York, NY, USA: ACM, 2017, pp. 1771–1779. [Online]. Available: http://doi.acm.org/10.1145/3123266.3123428

[18] J.-J. Chen, C.-W. Ngo, F.-L. Feng, and T.-S. Chua, "Deep understanding of cooking procedure for cross-modal recipe retrieval," in *Proceedings of the 26th ACM International Conference on Multimedia*, ser. MM '18. New York, NY, USA: ACM, 2018, pp. 1020–1028. [Online]. Available: http://doi.acm.org/10.1145/3240508.3240627

[19] M. Carvalho, R. Cadène, D. Picard, L. Soulier, N. Thome, and M. Cord, "Cross-modal retrieval in the cooking context: Learning semantic text-image embeddings," in *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '18. New York, NY, USA: ACM, 2018.

[20] Y. Kawano and K. Yanai, "Foodcam: A real-time food recognition system on a smartphone," *Multimedia Tools and Applications*, vol. 74, no. 14, pp. 5263–5287, 2015.

[21] R. Xu, L. Herranz, S. Jiang, S. Wang, X. Song, and R. Jain, "Geolocalized modeling for dish recognition," *IEEE Trans. Multimedia*, vol. 17, no. 8, pp. 1187–1199, 2015.

[22] T. Kusmierczyk, C. Trattner, and K. Norvag, "Understanding and predicting online food recipe production patterns," in *HyperText*, 2016.

[23] X. Wang, D. Kumar, N. Thome, M. Cord, and F. Precioso, "Recipe recognition with large multimodal food dataset," in *ICME Workshops*, 2015, pp. 1–6.

[24] C.-w. N. Jing-jing Chen, "Deep-based ingredient recognition for cooking recipe retrival," *ACM Multimedia*, 2016.

[25] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 Million Image Database for Scene Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 6, pp. 1452–1464, Apr. 2018.

[26] US Department of Agriculture, Agricultural Research Service, Nutrient Data Laboratory, "Usda national nutrient database for standard reference, release 27," May 2015. [Online]. Available: http://www.ars.usda.gov/ba/bhnrc/ndl

[27] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[28] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013.

[29] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *NIPS*, 2014, pp. 3104–3112.

[30] R. Kiros, Y. Zhu, R. Salakhutdinov, R. Zemel, A. Torralba, R. Urtasun, and S. Fidler, "Skip-thought vectors," in *NIPS*, 2015, pp. 3294–3302.

[31] L. Castrejon, Y. Aytar, C. Vondrick, H. Pirsiavash, and A. Torralba, "Learning aligned cross-modal representations from weakly aligned data," in *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*. IEEE, 2016.

[32] Y. Aytar, L. Castrejon, C. Vondrick, H. Pirsiavash, and A. Torralba, "Cross-modal scene networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 10, pp. 2303–2314, 2018. [Online]. Available: https://doi.org/10.1109/TPAMI.2017.2753232

[33] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," *arXiv preprint arXiv:1405.4053*, 2014.

[34] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," *arXiv preprint arXiv:1310.1531*, 2013.

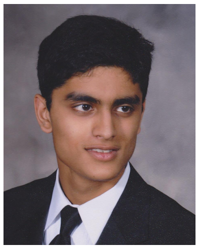
[35] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3128–3137.

[36] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3156–3164.

[37] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene cnns," *International Conference on Learning Representations*, 2015.

[38] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

Javier Marín received the B.Sc. degree in Mathematics at the Universitat de les Illes Balears in 2007. In June 2013 he received his Ph.D. in computer vision at the Universitat Autónoma de Barcelona. In 2017 he was a postdoctoral research associate at the Massachusetts Institute of Technology (MIT). Before that, he worked as an algorithm development engineer in the automotive sector, and as a researcher and project manager in both neuroscience and space fields. He currently combines working in the private sector as a senior data scientist at Satellogic Solutions with being a research affiliate at MIT. His research interests lie mainly in the area of computer vision and machine learning, focusing recently in cross-modal learning, object recognition and semantic segmentation.



Aritro Biswas received a Bachelor's degree in Computer Science at the Massachusetts Institute of Technology (MIT). He received his Master's degree in Computer Science at MIT. Recently, his research has focused on using computer vision for two applications: (i) understanding the content of food images and (ii) disaster recognition for images of humanitarian disasters.

Ferda Ofli (S'07–M'11–SM'18) received the B.Sc. degrees both in electrical and electronics engineering and computer engineering, and the Ph.D. degree in electrical engineering from Koc University, Istanbul, Turkey, in 2005 and 2010, respectively. From 2010 to 2014, he was a Postdoctoral Researcher at the University of California, Berkeley, CA, USA. He is currently a Scientist at the Qatar Computing Research Institute (QCRI), part of Hamad Bin Khalifa University (HBKU). His research interests cover computer vision, machine learning, and multimedia signal processing. He is an IEEE and ACM senior member with over 45 publications in refereed conferences and journals including CVPR, WACV, TMM, JBHI, and JVCI. He won the Elsevier JVCI best paper award in 2015, and IEEE SIU best student paper award in 2011. He also received the Graduate Studies Excellence Award in 2010 for outstanding academic achievement at Koc University.

Nicholas Hynes is a graduate student at the University of California Berkeley and research scientist at Oasis Labs. His research interests are generally in the domain of efficient machine learning on shared private data.

Amaia Salvador is a PhD candidate at Universitat Politcnica de Catalunya under the advisement of Professor Xavier Gir and Professor Ferran Marqus. She obtained her B.S. in Audiovisual Systems Engineering from Universitat Politcnica de Catalunya in 2013, after completing her thesis on interactive object segmentation at the ENSEEIHT Engineering School in Toulouse. She holds a M.S. in Computer Vision from Universitat Autnoma de Barcelona. She spent the summer of 2014 at the Insight Centre for Data Analytics in the Dublin City University, where she worked on her master thesis on visual instance retrieval. In 2015 and 2016 she was a visiting student at the National Institute of Informatics and the Massachusetts Institute of Technology, respectively. During the summer of 2018, she interned at Facebook AI Research in Montreal.

Yusuf Aytar is a Research Scientist at DeepMind since July 2017. He was a post-doctoral research associate at Massachusetts Institute of Technology (MIT) between 2014-2017. He received his D.Phil. degree from University of Oxford. As a Fulbright scholar, he obtained his M.Sc. degree from University of Central Florida (UCF), and his B.E. degree in Computer Engineering in Ege University. His research is mainly concentrated on computer vision, machine learning, and transfer learning.

Ingmar Weber is the Research Director for Social Computing at the Qatar Computing Research Institute (QCRI). As an undergraduate Ingmar studied mathematics at Cambridge University, before pursuing a PhD at the Max-Planck Institute for Computer Science. He subsequently held positions at the Ecole Polytechnique Federale de Lausanne (EPFL) and Yahoo Research Barcelona, as well as a visiting researcher position at Microsoft Research Cambridge. He is an ACM, IEEE and AAAI senior member.

Antonio Torralba received the degree in telecommunications engineering from Telecom BCN, Barcelona, Spain, in 1994 and the Ph.D. degree in signal, image, and speech processing from the Institut National Polytechnique de Grenoble, France, in 2000. From 2000 to 2005, he spent postdoctoral training at the Brain and Cognitive Science Department and the Computer Science and Artificial Intelligence Laboratory, MIT. He is now a Professor of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology (MIT). Prof. Torralba is an Associate Editor of the International Journal in Computer Vision, and has served as program chair for the Computer Vision and Pattern Recognition conference in 2015. He received the 2008 National Science Foundation (NSF) Career award, the best student paper award at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) in 2009, and the 2010 J. K. Aggarwal Prize from the International Association for Pattern Recognition (IAPR).