TasteRank: Personalized Image Search and Recommendation

Nicholas Asker
nga2120@columbia.edu

Abstract

How can we find relevant images in a database when queries are arbitrary user preferences such as "get protein from plants"? To compute relevance scores for each image, I investigate following a two-step framework: (1) first break down the target preference into descriptive features, then (2) perform recognition on the images over these features. The first step produces a list of descriptors for a given preference by calling on a language model, outputting, for example, [Legumes, Nuts & seeds, Leaved greens, Tofu & tempeh]. Step two leverages CLIP to test for the presence of the descriptors in zero-shot fashion. Then, a final score is computed for each image capturing the extent to which the descriptors were detected, and the score is treated as an approximation for relevance. For relevant-image identification, the proposed approach outperforms the baseline and comes built-in with a mechanism to explain its decisions. As an alternative to traditional tag-based retrieval schemes, the method has the potential to provide more sophisticated personalization.

1. Introduction & Related Work

How can we find relevant images in a database when queries are user preferences? This is a difficult task, partly because such preferences may express completely novel desires and vary widely along features such as specificity. This domain, however, is of critical importance: relevant-image retrieval is integral to the modern economy, from content-recommendation algorithms on social-media platforms to serving ads across the internet.

A simple approach is to use CLIP [5] to assign a score to each image: convert both the target preference text and image into an embedding (in shared, multimodal latent space), and compute their similarity. This score can be used to sort the images and return results to the user, ordered by relevance. But how good is this approach, really? And can we improve upon it?

This work explores these questions by comparing simple CLIP against a descriptor-based approach to recognition, as in Menon and Vondrick (2022) [4], specifically for the task of personalized image retrieval from a food database. The proposed approach, illustrated in Figure 1, consists of two steps:

1. Break down the target preference into descriptive features, or **descriptors**
2. Perform recognition on the images over the descriptors.

Images satisfying many of the descriptors end up with higher scores, while those lacking them receive lower scores. To generate the descriptors, I leverage a language model, and I use CLIP to determine the extent to which an image satisfies any particular descriptor, capitalizing on CLIP's zero-shot recognition capability [5].

I term the overall framework **TasteRank**[1] and find that a slight modification to the aforementioned method − combining it with the baseline as a **hybrid** approach − achieves the best results in experiments.

More broadly, this work extends Menon and Vondrick (2022) [4], applying *classification by description* to a new problem context but deriving its many benefits, including explainable decision-making and adaptability to recognize novel phenomena (to which CLIP was not directly exposed during training, such as proper nouns in more recent trends or the news). In addition, I draw inspiration from the nearest-neighbor retrieval scheme in Torralba *et al.* (2008) [6], especially in performing recognition at varying semantic levels of abstraction by leaning on an external source of linguistic knowledge (WordNet in their case, a language model here).

2. Methodology

2.1. Dataset

For the dataset, I used Food-101 [1]. The data consists of images of food dishes, each labeled with one of 101 possible classes (e.g., "beignets" or "spaghetti bolognese"). Write K to denote this set of classes. The whole dataset contains $101,000$ real-world photos sourced from

[1]The project code is available at https://github.com/N-G-Asker/TasteRank

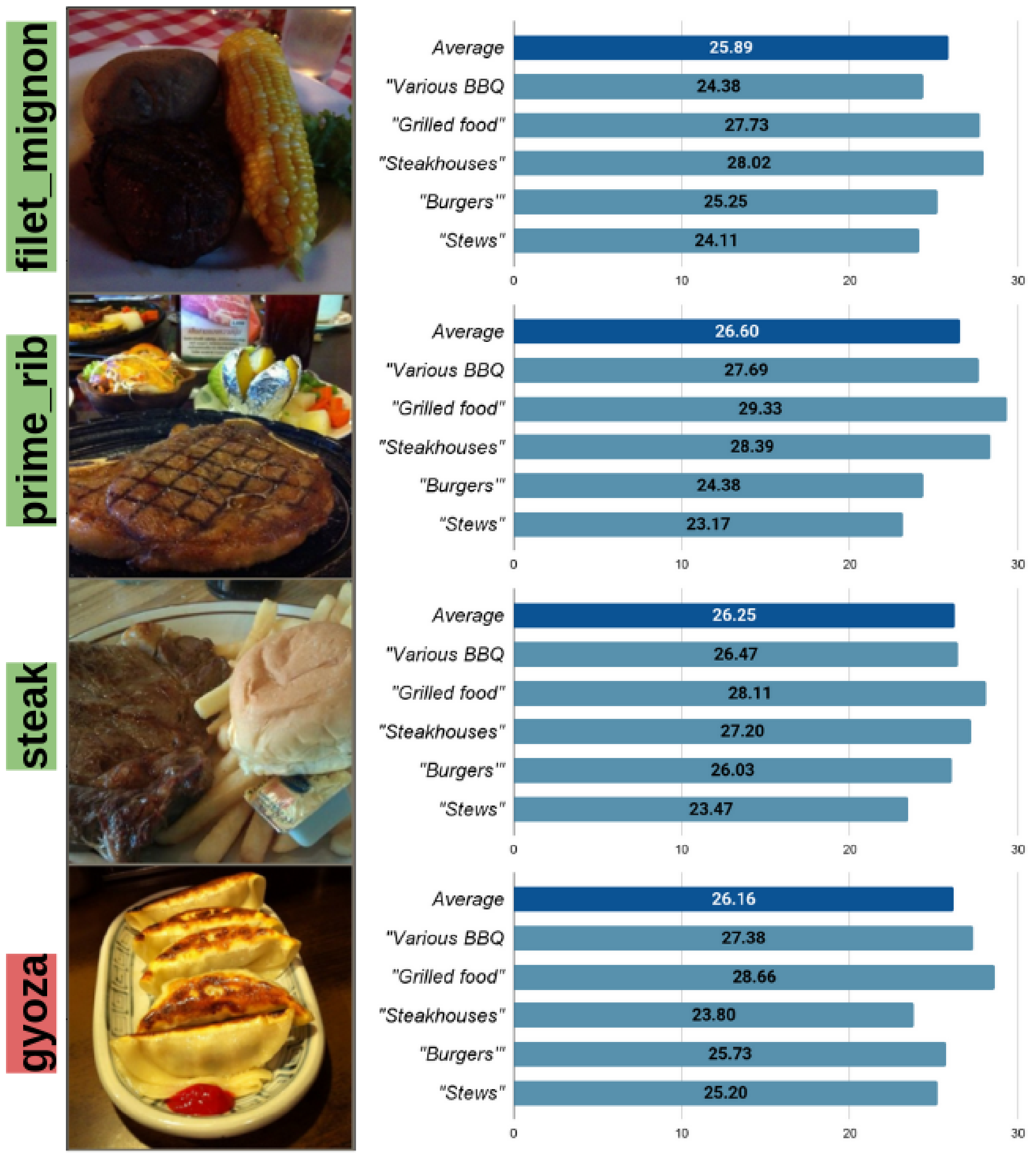


Figure 1. This particular test (test_id=193) tested the preference "eat lots of meat". Shown are the four images from the testset that my approach scored the highest, with their class label on the left, colored depending on whether it was considered to match the preference. For this test, the full set of matching classes was { baby_back_ribs, beef_tartare, chicken_wings, filet_mignon, hamburger, peking_duck, pork_chop, prime_rib, pulled_pork_sandwich, spaghetti_bolognese, spaghetti_carbonara, steak }. Note gyoza's absence. The scoring breakdown for each image is shown on the right – its score is the average over all five of the individual descriptor scores. A full test involves scoring all $5,050$ images in the testset. Here, the baseline's scores achieved a Pearson correlation with matching classes of 0.454, while my approach achieved 0.414 (both above average performances).

user uploads to a restaurant-focused social-media platform. To make the project possible under time and resource constraints, I only used a subset of the available data. In particular, I randomly sampled 20% of the test split, resulting in an experimentation dataset $\mathcal{D}$ of length $n = 5,050$ images, with 50 example dishes per class on average.

2.2. Method

Assume as given a target preference π and $D(\pi)$, the descriptors list for this π. I compute an estimated relevance score s for each image x in $\mathcal{D}$, by averaging over the CLIP similarity scores between x and each descriptor $d \in D(\pi)$. These are computed following the same additive decomposition technique as Menon and Vondrick (2022) [4]:

$$s(\pi, x) = \frac{1}{|D(\pi)|} \sum_{d \in D(\pi)} \phi(d, x)$$

where $\phi(d, x)$ is the cosine similarity score between d and x's respective CLIP embeddings.

The resulting values of s can then be used to sort, or rank, the images. If s is a good approximation for relevance to the end-user, this is highly useful for returning the best results. Experiments in this study attempt to assess how well s does at estimating relevance.

Section 2.3.3 addresses how each descriptors list $D(\pi)$ is established.

2.3. Experimental Setup

Setting up experiments involved three parts:

1. Generate Pool of Target Preferences
2. Determine Preference-Matching Classes
3. Generate Descriptors

Figure 2 illustrates the output of this experimental setup procedure, and each step is detailed in turn below.

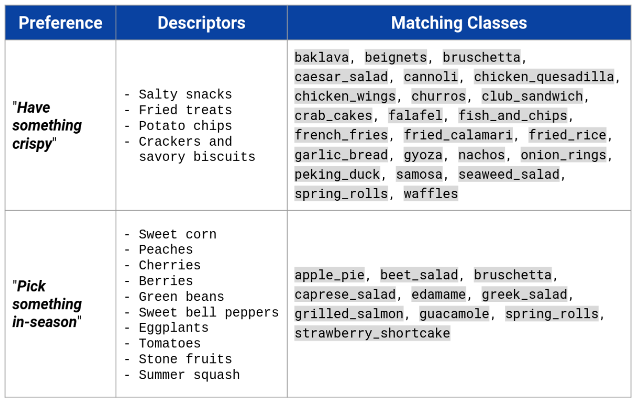
2.3.1 Generate Pool of Target Preferences

I generated a list of m target preferences $\Pi = (\pi_1, ..., \pi_m)$ to simulate real-world user preferences, or queries, related to food.[2] Preferences express desires to consume specific types of food, and as strings take the form

"{verb} + {specification}",

for example to "eat from the kids menu", "have Mediterranean food", "enjoy tart flavors", or "snack on a porch in summer". Generating this list from scratch was necessary for lack of an obvious source for arbitrary preferences of this form, related to food.

[2]One could use the term persona for a hypothetical user embodying a particular preference π_i.



Preference	Descriptors	Matching Classes
"Have something crispy"	- Salty snacks - Fried treats - Potato chips - Crackers and savory biscuits	baklava, beignets, bruschetta, caesar_salad, cannoli, chicken_quesadilla, chicken_wings, churros, club_sandwich, crab_cakes, falafel, fish_and_chips, french_fries, fried_calamari, fried_rice, garlic_bread, gyoza, nachos, onion_rings, peking_duck, samosa, seaweed_salad, spring_rolls, waffles
"Pick something in-season"	- Sweet corn - Peaches - Cherries - Berries - Green beans - Sweet bell peppers - Eggplants - Tomatoes - Stone fruits - Summer squash	apple_pie, beet_salad, bruschetta, caprese_salad, edamame, greek_salad, grilled_salmon, guacamole, spring_rolls, strawberry_shortcake

Figure 2. Two examples of LLM-generated food preferences are shown, together with (a) their corresponding LLM-generated descriptors and (b) their matching classes, the subset of Food-101 classes that an LLM considered applicable.

To produce plausible preferences at a sufficient scale, I took advantage of large language models (LLMs). Specifically, I used Mistral 7B v0.1 [3], an open, transformer-based [7] model with attention optimizations. After designing a prompt utilizing in-context learning [2], I generated with the model using multinomial sampling as the decoding strategy. See Appendix A.1 for the prompting details. I then manually reviewed the generations to ensure the overall list was reasonable.[3] This step resulted in a list Π of length $m = 264$.

Note that, in practical settings, target user preferences are known. For example, in search applications, a preference is furnished in the form of an end-user's query string; in the recommendation domain – in which unsolicited recommendations need to be served without the end-user explicitly providing a query – preferences would come from some user profile, browsing and search history, etc.

2.3.2 Determine Preference-Matching Classes

Creating synthetic labels for an existing dataset was necessary because, to my knowledge, there does not exist a publicly available dataset pairing arbitrary user preferences and images.

For each of the 264 preferences established above in 2.3.1, I automatically annotated each of the 101 food classes on applicability. That is, for every $\pi_i \in \Pi$, and for all $k \in K$, a boolean **match** value $M_{\pi_i}^{(k)}$ was assigned to

[3]My intention was to assess the quality of the list as a whole: I did not remove duplicates or preferences which, on their own, seemed inappropriate. Preferences that were so broad that all classes matched it, or so narrow such that no classes matched it, were filtered out in the next step.

capture whether the class is applicable for the given preference. For example, the preference to "get food that is sour" matched with only three out of the 101 categories, namely *ceviche*, *greek salad*, and *hot and sour soup*. All images belonging to one of these three classes received True values for $M_{\text{"get food that is sour"}}$, while images labeled with one of the other 98 categories received False. This step amounted to augmenting each sample in $\mathcal{D}$ with a (sparse) 264-dimensional vector depending on its class label.

To systematically achieve this labeling, I again leveraged LLMs and in-context learning, iteratively asking "Would {class name} normally match a preference to {preference}?". In particular, Mistral 7B v0.1 was prompted to decide applicability for every (k, π_i) pair. Adjudicating all cases had a Big-O time cost of $O(|K| * m)$. See Appendix A.2 for details and the prompt template.

For the purposes of the experiment, the language model's responses are treated as supplying the ground-truth matching classes. I manually reviewed its judgements to ensure that, overall, it was making reasonable decisions. While the resulting labels are imperfect, they plausibly approximate the ground-truth, and thereby provide a way forward to evaluate my framework against the baseline.[4]

2.3.3 Generate Descriptors

To efficiently obtain the descriptors for every preference in Π, I again utilized an LLM – Mistral 7B v0.1 once again. Designing the right prompt, to elicit the most useful descriptors, was challenging here. What type of descriptors should one solicit? One possibility would be to follow the approach in Menon and Vondrick (2022) [4] of asking explicitly for *visual* descriptors relevant to the target preference, e.g., querying for visual cues of Mediterranean food. But in initial experimentation (using the Food-101 trainset for validation), it did not seem to work well for this particular problem.

Ultimately, I abstained from deciding, choosing to word my request in more general terms. I prompted the LLM to respond as a bulleted list to the question, "*What kind of foods do I want if I want to {preference}?*", with several demonstrations prepended as additional context (see Appendix A.3 for details). This left it to the LLM to interpret and "decide" on a case-by-case basis, based on the preference at hand, what an apt response would be (implicitly deciding such things as the level of detail, how to make the list mutually exclusive and collectively exhaustive, etc.). For example, in Figure 2, the type of descriptors-list generated varies between the two examples: one is of an ingredient-based nature, while the other takes on a more categorical, overlapping-concept perspective.[5]

After generation, I confirmed that, on the whole, the responses were reasonable and consistent with my conception of descriptors.

In real-world settings such as search and recommendation systems, TasteRank descriptors could be generated on the fly by calling on an LLM. But past user activity in itself could serve as a source of descriptors, without involving an LLM. As a simplified example: if a user is known to like both bananas and ice cream, then one might hope that using these terms as descriptors will return images of foods like *banana split* or *banana yogurt* as the top scorers.

2.4. Performance Metrics

Baseline: I use CLIP to compute per-sample similarity scores ϕ. A given preference π (as text) and photo x are both embedded into the same space, and the inner product of their embeddings $\phi(\pi,\ x)$ computed. We then consider how well these values of ϕ approximate relevance.

Definitions: Given a π, denote $S = \{r_j\}_{j=1}^n$ as the list of estimated relevance scores output by an approach, one for each x_j in $\mathcal{D}$. For our approach, $r_j = s(\pi,\ x_j)$, whereas for the baseline, $r_j = \phi(\pi,\ x_j)$. Write $M = \{\mathbb{1}_\pi(x_j)\}_{j=1}^n$ for the list of "match" labels, a binary variable indicating each sample's applicability to π, based on whether its class-label belongs to the subset of classes deemed pertinent to π.

Metrics: For every test preference $\pi_i \in \Pi$, I needed to gauge how well the scores produced by either approach track with the ground-truth match labels. One straightforward way to do this was by measuring the Pearson correlation coefficient ρ between S and M.[6] I analyze these ρ values across all 264 tests, computing the average and variance to determine how well both approaches track relevance across the diverse set of preferences captured in Π.

3. Experimental Results

Table 1 summarizes the experimental results.[7]

[4]To reiterate, there is no clear way to otherwise collect gold labels for arbitrary, abstract preferences in this problem context. While one could manually go through the list of preferences and hand-select matching classes, doing so at scale would not be possible, and making decisions for certain subtle or subjective preferences like "eat umami flavors" or "have strong smelling food" would cause difficulty, so I took a more systematic approach of querying an LLM.

[5]I hypothesize that this technique worked better here, because it was more flexible to the wide range of preferences, which vary from the specific/concrete ("have something with mushrooms") to vague/abstract ("eat something you shouldn't have"). Dictating a priori that descriptors be of a particular nature ran the risk of producing sensible descriptors for some preferences but useless descriptors for others, given such diversity.

[6]In this case, equivalent to the point biserial correlation coefficient. See https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.pointbiserialr.html

[7]I report full experiment details and results at https://github.com/N-G-Asker/TasteRank. The repository contains a performance summary for each approach, as well as details on each individual

Approach	Average Correlation	σ
Baseline	0.223	0.181
Ours	0.219	0.167
Hybrid	**0.234**	0.168

Table 1. Over 264 test preferences, the baseline outperforms my approach, but together – the "hybrid" approach – they achieve the highest correlation. This suggests combining descriptors and the original query term as a potential path forward for relevant image retrieval.

On individual tests, my approach and the baseline tended to perform similarly, but not identically, and occasionally, quite differently. To confirm this, I calculated the Pearson and Spearman correlations between the two (that is, between my and the baseline's per-test correlations over the 264 tests), and found coefficients of 0.775 and 0.733 respectively. Further confirmation came from juxtaposing their performance directly on a subset of tests – the 10 preferences for which my approach performed the best, and the 10 worst (see Appendix B for details and visualization).

It was therefore natural to test a hybrid approach combining the baseline and the descriptors-based method. I simply appended the preference to the descriptors list as though it were another criterion, and executed the same experimental procedure. For example, for the first test shown in Figure 2, with π ="Have something Crispy", the descriptors list was augmented from [Salty snacks, . . . , Crackers and savory biscuits] to [Salty snacks, . . . , Crackers and savory biscuits, *Have something Crispy*].

The hybrid approach achieves slightly higher average correlation than the baseline alone, while reducing the standard deviation from the baseline. This suggests combining descriptors and the original query term as a promising direction for reliably fetching relevant images.

4. Discussion

Although TasteRank involves LLMs, it is distinct from purely linguistic approaches and ones that rely more heavily on LLMs. In particular, my approach relies on the power of CLIP's shared-embedding space to discern implicitly related visual and linguistic concepts. This makes it possible to discover and order relevant images, without relying on tags. Indeed, the database need not have any tagging whatsoever. In contrast, an LLM-heavier approach might rely on mapping each query to tags already present in the database. This would be interesting to see in future work.[8]

A key advantage of TasteRank is its inherent transparency. Clear explanations for why search results appeared first can be given by simply pointing to the individual descriptor scores. Taking the mistaken retrieval in Figure 1 as an example, one can understand why the recommendation system served up that particular photo of gyoza: at one point, the user queried for "food with lots of meat"; grilled food was strongly detected in the photo (that is, the image received a high individual descriptor score on grilled food); and the presence of grilled food is considered one of the most important features for matching this preference (it's one of the select few descriptors). This traceability is valuable to both end-users and the organizations designing these systems.

4.1. Limitations

Strength of Descriptor Generator: Reviewing the preferences for which our model performed the worst revealed a weak set of descriptors was usually to blame. For example, for test 201 "have a soup course", the descriptors are nonsensical: [Classic crepe, Fruit and nut tart, Egg-based casserole]. Access to a stronger generator, and better prompt design, would likely address this failure mode and boost TasteRank's performance.

Test Overlap: This limitation relates to the experimental design, not the approach. The preferences list contains some amount of redundancy, ranging from partial conceptual overlap to duplicate items, so some tests tested roughly the same preference. For example, of the ten tests on which our model did the best, nine had dessert as a common theme. This issue could be mitigated in future work by using a more powerful LLM to generate the preferences list (perhaps designing better prompts, with more demonstrations, to increase creativity and reduce repetition), and by better filtering the generated list.

Scalability and Runtime: This concern questions the approach's feasibility in practice, at scale. Often, image databases are orders of magnitude larger than the testset I used for experiments. Would it not be prohibitively expensive to go through every image in real-world deployments, computing similarity scores over all descriptors? This problem, however, is not unique to TasteRank, and there exist clever ways to reduce the number of images considered, from sampling methods to applying smart filtering upfront. That said, my approach might work best in practice as the final component of a search-results pipeline, to curate results from a pool whose size has already been reduced via some pre-filtering scheme(s). For example, after filtering the entire database for Mediterranean food, there is still the need

test (including preferences, descriptors, match-classes, and Pearson correlations).

[8] One idea for how this might work: If a user wants Mediterranean food, (1) query the LLM for concepts related to Mediterranean food, to serve as search terms, (2) query the database for images with (pre-exisiting) tags corresponding to the search terms, and finally, (3) use some algorithm to rank all the matches.

to find and rank order the top (personalized) results for the given user. This “last-mile” of information retrieval could benefit from TasteRank, instead of relying on finer-grained tags (capturing subtle, difficult-to-pin-down preferences) to perform final filtering.

5. Conclusion

This study proposes TasteRank, a descriptor-based recognition approach for personalized image retrieval. It consists of a simple pipeline: first break down a target preference into descriptors, and then, on each image in the database, perform recognition over the descriptors to produce relevance scores. Experimental results show TasteRank, instantiated as the hybrid approach, outperforms the baseline. Moreover, it is flexible, supporting arbitrary preferences, and its relevance scores are inherently explainable. Possible avenues for future work include leveraging more powerful LLMs for experiments, especially for descriptor generation, and exploring other LLM-assisted, tag-based retrieval methods.

References

[1] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014. 1

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. 3

[3] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. 3

[4] Sachit Menon and Carl Vondrick. Visual classification via description from large language models, 2022. 1, 3, 4

[5] 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, 2021. 1

[6] Antonio Torralba, Rob Fergus, and William T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008. 1

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. 3

A. Prompting

All prompts leverage in-context (few-shot) learning, prepending a couple demonstrations before the real question. I came up with all demonstrations myself, simply by brainstorming and trial and error.

A.1. Preference Generation

See Figure 3 for the 2-shot prompt I used.

A.2. Determining Preference-Matching Classes

See Figure 4 for the 5-shot prompt I used.

A.3. Descriptor Generation

See Figure 5 for the 4-shot prompt I used.

B. Top-10 vs Worst-10 Performance Analysis

Figure 6 illustrates the data. While the performance of the baseline and our approach are very similar for the tests on which our approach does the best, a large disparity can be seen in some of the individual worst-performing tests. For example, there was a large performance gap of 0.64 in test 201 (preference = “have a soup course”). This suggested that a combination of the two might work better, leading to the idea of the hybrid approach.

Preference Generation Prompt

```
Q: What are some standard food preferences someone might have?
A: There are several common food preferences a person could have. They may want to:
- be vegetarian
- avoid nuts
- keep kosher
- eat vegan
- avoid gluten
- be pescetarian
- keep away from dairy

Q: What are some creative food preferences someone might have?
A: There are several creative food preferences a person could have. They may want to:
- get mediterranean food
- eat tart flavors
- have something that smells strong
- eat umami flavors
- eat tomato-based dishes
- find summer dishes
- eat something stinky
- eat caribbean flavors
- have pirate-themed cuisine
- eat surprising flavors
- have exotic flavors
- be adventurous
- diet
- lose weight
- be health-conscious
- eat food that goes with kimchi
- eat something that smells like a fire
- try something floral
- order from the kids menu
- have acidic food
- have a smelly meal

Q: What are some interesting food preferences someone might have?
A: There are several interesting food preferences a person could have. They may want to:
-
```

Figure 3. The prompt used as context with Mistral 7B v0.1 to generate the list of feasible food preferences.

Prompt: Decide Preference-Matching Classes

```
Q: Would salad normally match a preference to eat healthy?
A: yes

Q: Would pork chops normally match a preference to keep kosher?
A: no

Q: Would bok choy normally match a preference to eat something green?
A: yes

Q: Would tacos normally match a preference to have something with corn?
A: yes

Q: Would penne alla vodka normally match a preference to be non-dairy?
A: no

Q: Would {class name} normally match a preference to {preference}?
A:
```

Figure 4. The prompt used as context with Mistral 7B v0.1 to decide whether each possible pair of ({class}, {preference}) was a match or not, that is, whether the class was applicable to the given preference.

Prompt: Generate Descriptors

```
Q: What kind of foods do I want if I want to **eat sugary food**? Answer as a very simple, short list.
A: You may want the following types of foods:
- Desserts
- Candies and chocolates
- Sugary drinks
- Pastries & baked goods

Q: What kind of foods do I want if I want to **be kosher**? Answer as a very simple, short list.
A: You may want the following types of foods:
- Kosher meat
- Permitted fish
- Fresh produce
- Certified processed foods

Q: What kind of foods do I want if I want to **have noodles**? Answer as a very simple, short list.
A: You may want the following types of foods:
- Asian noodles
- Italian pastas
- Soupy noodles
- Quick cup noodles

Q: What kind of foods do I want if I want to **make something on a porch in the summer**? Answer as a very simple,
   short list.
A: You may want the following types of foods:
- Grilled fare
- Cool salads
- Summery drinks
- Easy finger foods
- Frozen treats

Q: What kind of foods do I want if I want to **{preference}**? Answer as a very simple, short list.
A: You may want the following types of foods:
-
```

Figure 5. The prompt used as context with Mistral 7B v0.1 to get $D(\pi_i)$, the set of descriptors for the given preference.

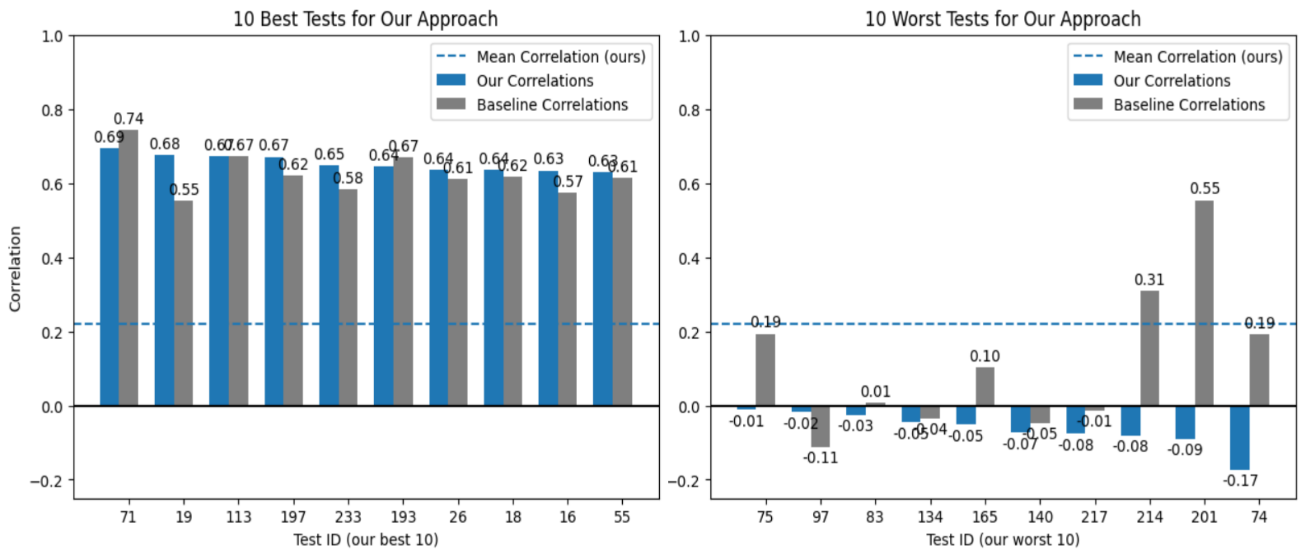


Figure 6. Juxtaposing the baseline vs. our performance directly on a subset of tests − the 10 preferences for which my approach performed the best, and the 10 worst.