LUNCH AT THE EIGENSALAD BAR: LINEAR APPROACHES TO DIMENSIONALITY REDUCTION FOR IMAGE PROCESSING

TECHNICAL REPORT

Zachary Novack
Carnegie Mellon University
Pittsburgh, PA 15217
znovack@andrew.cmu.edu

ABSTRACT

Current pseudo-academic satirical thought on the boundaries of food categories are often overly restrictive or defined exactly and exhaustively (i.e. the set of all sandwiches is everything explicitly said to be a sandwich), and consistently lack empirical evidence of their claims. This paper seeks to act as a survey of techniques in which to experimentally validate the *Hyper-Salad Space Standard Projection*, a low dimensional basis for the space in which all salads and saladoids exist in. Using Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Locality Preserving Projections (LPP), Independent Component Analysis (ICA), and Non-Negative Matrix Factorizations (NMF), we are able to recover the estimated basis for each low dimensional representation of our food image dataset. Empirical results in ICA and PCA showed that there may be some evidence of the arrangement entropy axis of HS3P, and thus further research with stronger dimensionality reduction techniques are needed in order to investigate this further.

1 Introduction

What is a salad? While the answer may seem trivial, a surprising amount of satirical and scientific thought has been invested in formally defining the boundaries of this set. Many of these theories often draw some arbitrary line of exclusion explicity, thus restricting the set of all salads to something that is cognitively tractable but scientifically uninteresting, or necessarily define the set such that the definition works for *most, but not all* salad-like foods. [8] offers a wide-encompassing approach that successfully incorporates all traditionally held salads but is defined using limited constraints, and posits the *Hyper-Salad Space Standard Projection* (HS3P) as a way to visualize the set in a low-dimensional subspace. Qualitatively validating the HS3P presents an interesting image processing task; we thus approach it through the use of linear dimensionality reduction techniques. Thus, this report presents a survey of different linear matrix factorization algorithms, their strengths and weaknesses, computational bottlenecks, and how they perform on decomposing food image data into interpretable bases.

2 Linear Dimensionality Reduction

As the name implies, linear dimensionality reduction techniques can be characterized as algorithms that project a set of data in high-dimensional space into some lower-dimensional space *linearly*, as opposed to nonlinear dimensionality reduction techniques such as t-SNE, Laplacian Eigenmaps, and ISOMAP. Specifically, for our high-dimensional data $X \in \mathbb{R}^{n \times d}$ (i.e. n samples in d-dimensional space), we find:

$$XP = Z \in \mathbb{R}^{n \times k} \tag{1}$$

Where Z is our low-dimensional representation of the initial data ($k << d$) and $P \in \mathbb{R}^{d \times k}$ is the projection matrix that defines each algorithm. Notably, this means that every data point (in our case, every food image) in X can be represented as a linear combination of the columns of P. In the case of Eigenfaces [12], P represents the top k eigenvectors of the correlation matrix $X^\top X$, and when used on image data, represents the k "eigenfaces" that act as a visualization of the axes of largest variance in image data. This approach of using the columns of P to visualize how X is linearly separated into components (and thus approximate HS3P) motivates the rest of the paper.

2.1 Principal Component Analysis (PCA)

2.1.1 Traditional PCA

Principal Component Analysis (PCA) is an exceedingly popular technique for linear dimensionality reduction. Here, the algorithm attempts to minimize the reconstruction error between the high-dimensional data and their low-dimensional representations (i.e. the Euclidean distance between the original data and the projected data) through the following [11]:

1. Center X (that is, $X = X - \boldsymbol{\mu}$, where $\boldsymbol{\mu}_j$ is the mean of the jth column of X
2. Compute $C_x = \frac{1}{n} X^\top X$
3. Eigendecomposition: $C_x = U \Lambda U^\top$ where $\Lambda = diag(\lambda_1, \dots, \lambda_d)$ and U_j (jth column of U) is the eigenvector corresponding to the λ_j
4. $P = k$ eigenvectors corresponding to the top k eigenvalues
5. $Z = XP$

Thus, the columns of P are the orthonormal directions of maximum variance (one can actually contruct PCA as a maximization problem of variance rather than minimizing reconstruction error) and represent (in our food image use-case) the *EigenSalads* that form a basis of our space.

While this approach generally works for data in which d is small, the classical PCA algorithm generally fails for high-dimensional data. This is due to both the conditioning of the linear system for PCA and its computation complexity. In classical PCA, the condition number $\kappa(C_x)$ is proportional to $\kappa(X)^2$, and thus PCA for ill-conditioned matrices may compute $\tilde{P}$ that is considerably far off from what P would be in exact arithmetic [9]. Additionally, the computational costs of computing the covariance matrix and the eigendecomposition are $O(nd^2)$ and $O(d^3)$, so for high dimensional data such as images (where even an 100×100 pixel image is in $\mathbb{R}^{10000}$) these steps become huge computational bottlenecks. The storage costs for the covariance matrix is also $O(d^2)$, which also becomes an issue for high-dimensional data. These issues thus motivate the most-common numerical implementation of PCA: PCA by SVD.

2.1.2 SVD PCA

SVD PCA takes advantage of the relationship between the eigenvalues of a matrix and its singular values. Let $X' = \frac{1}{\sqrt{n}}(X - \boldsymbol{\mu})$ (i.e. X' is the scaled high-dimensional data with zero mean), and note that trivially $C_x = X'^\top X'$. Instead of directly taking the eigendecomposition of C_x, we instead perform SVD on X' such that $X' = U \Sigma V^\top$, where U and V are orthogonal matrices and Σ is a diagonal matrix [10]. Through this, we get the following equality:

$$\begin{aligned} C_x &= X'^\top X' \\ &= (U\Sigma V^\top)^\top U \Sigma V^\top \\ &= V \Sigma^\top U^\top U \Sigma V^\top \\ &= V \Sigma^2 V^\top \end{aligned} \tag{2}$$

Thus, this offers an alternative eigendecomposition of the covariance matrix without ever explicitly calculating it, as the columns of V contain the eigenvectors of C_x and Σ^2 contains the respective eigenvalues. While this process does not directly deal with the $O(d^3)$ computational bottleneck from before (as normal SVD is also $O(d^3)$), SVD PCA is considerably more numerically stable than traditional PCA, since by not computing the covariance matrix, the error in the system is only proportional to $\kappa(X)$. In practice, many approaches to SVD PCA take advantage of the fact that we are only interested in the k largest right singular vectors, thus making the calculation of the rest of the decomposition irrelevant, and perform Truncated SVD:

$$\tilde{X} \approx U_k \Sigma_k V_k^\top \tag{3}$$

Where U_k and V_k are only the top k left and right singular vectors respectively and Σ_k is only the top k singular values. By using a randomized approached to compute this approximation of the full SVD [3], the truncated SVD algorithm is able to achieve a time complexity of $O(ndk)$, making it superior to classical SVD in every bottleneck.

2.2 Linear Discriminant Analysis (LDA)

Since, in the case of food image processing, we are dealing with class-labeled data, one may wonder if finding the low-dimensional representation that maximizes variance (as we do in PCA) is as important as maximizing *class discrimination*. Fisher's Linear Discriminant (also known as Linear Discriminant Analysis or LDA) is another linear dimensionality reduction technique that specifically aims to generate a low-dimensional representation of the data for the aid of classification (and is thus, a supervised method as opposed to an unsupervised method like PCA) [1]. Let $\Sigma_c, \Sigma_b \in \mathbb{R}^{d\times d}$ be the within-class and between class scatter matrices, such that:

$$\begin{aligned}\Sigma_c &= \sum_{i=1}^{c} \sum_{x_j \in X_i} (x_j - \tilde{\boldsymbol{\mu}}_i)(x_j - \tilde{\boldsymbol{\mu}}_i)^\top \\ \Sigma_b &= \sum_{i=1}^{c} (\tilde{\boldsymbol{\mu}}_i - \boldsymbol{\mu}_X)(\tilde{\boldsymbol{\mu}}_i - \boldsymbol{\mu}_X)^\top\end{aligned} \tag{4}$$

Where c is the number of classes, $\boldsymbol{\mu}_X$ is the mean vector of each feature in the whole dataset and each $\tilde{\boldsymbol{\mu}}_i$ is the mean vector of each feature for the ith class. Then the columns of the optimal projection matrix $P \in \mathbb{R}^{d\times k}$ to maximize class separation are given by the k eigenvectors corresponding to the k largest eigenvalues of the following generalized eigenvalue problem:

$$\Sigma_b v = \lambda \Sigma_c v \tag{5}$$

(One can also think of $P = \text{argmax}_{P^*} \frac{P^{*\top}\Sigma_b P^*}{P^{*\top}\Sigma_c P^*}$, which for $k = 1$ represents maximizing the generalized Rayleigh Quotient of the linear system). The main issue that naturally arises in image processing is that $n << d$ in most cases, and since the the rank of Σ_c is at most $n - c$, Σ_c is always singular. We work around this issue by first applying SVD PCA to reduce the dimensionality of the dataset to $n - c$ (thus producing the projection matrix $P_{pca} \in \mathbb{R}^{d\times n-c}$) and then applying LDA on the intermediate dimension representation (producing $P_{lda} \in \mathbb{R}^{n-c\times k}$), and thus P is given by:

$$P = P_{lda}P_{pca} \tag{6}$$

This is the most commonly used algorithm for the *Fisherface* method, and here we refer to the component columns of P as *FisherSalads*. In comparison to PCA, LDA can often perform much better on dimensionality reduction tasks when the classes are not separated along directions of high variance (in fact, PCA can actually make class distinctions *worse*), though in its direct approach the within and between class scatter matrices share the same issues that classical PCA has in terms of instability, and the direct eigendecomposition to solve Eq. 5 is still $O(d^3)$.

2.3 Independent Component Analysis (ICA)

Whereas PCA attempts to find the components that are the orthogonal directions of maximum variance, Independent Component Analysis (or ICA) attempts to find the directions that are the most *independent* of one-another. ICA originated as a solution to the blind-source separation problem in signal processing, and for the sake of this paper we specifically refer to the FastICA algorithm [6].

FastICA seeks to offer a fast, parallelizable algorithm for computing k independent components from a set a data. We provide the pseudo-code of the algorithm in Algo.1.

Algorithm 1: FastICA

```
for i = 1 ..., k do
    initialize $\boldsymbol{w}_i$
    while not converged do
        $\boldsymbol{w}_i^* = \frac{1}{n}\sum_{j=1}^{n} x_j g(\boldsymbol{w}_i^\top x_j) - \frac{1}{n}\sum_{j=1}^{n} g'(\boldsymbol{w}_i^\top x_j)\boldsymbol{w}_i$
        $\boldsymbol{w}_i^+ = \boldsymbol{w}_i^* / ||\boldsymbol{w}_i^*||$
        $\boldsymbol{w}_i = \boldsymbol{w}_i^+ - \sum_{l=1}^{i-1} \boldsymbol{w}_i^{+\top}\boldsymbol{w}_l\boldsymbol{w}_l$
        $\boldsymbol{w}_i = \boldsymbol{w}_i / ||\boldsymbol{w}_i||$
    end
end
```

Where g is some differentiable, nonquadratic function. Thus, we estimate each independent component successively and at each iteration decorrelate the intermediate iterate with all the previously calculated independent components.

Note that though the k independent components (for the sake of our study refered to as *ICAnSalads*) are computed in succession and thus have an inherent bias for earlier computed components, there is no actual intrinsic ordering/importance to the set of independent components as their is in PCA or LDA (since both use ranked eigenvectors). This algorithm converges an order of magnitude faster than traditional ICA algorithms, and is remarkably stable depending on the choice of the function g. Since ICA is not limited to estimating at most second-order interactions (as PCA is with the covariance matrix), ICA projections can possibly be much more rich in information than techniques like PCA and LDA. The conditioning of this method can be improved through the use of preprocessing steps such as centering and whitening (that is, transforming the initial data so that each feature is decorrelated from the rest and each has unit variance).

2.4 Locality Preserving Projections (LPP)

Many nonlinear dimensionality reduction techniques wish to prioritize local relationships between data points over global interactions, yielding such methods as Local Linear Embedding or Laplacian Eigenmaps, which are both grounded in Spectral Graph Theory. Locality Preserving Projections (LPP) [5] works from this Spectral clustering framework but is notably linear in its approach and is thus fully defined in the low-dimensional subspace in which it projects data onto.

Let $W \in \mathbb{R}^{n \times n}$ be the adjacency matrix representation of the graph of the input data G such that:

$$W_{ij} = \begin{cases} e^{\frac{\|x_i - x_j\|^2}{h}} & \text{if } x_i \text{ and } x_j \text{ are neighbors} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Where h is the width of the heat kernel and the neighbor constraint determined by running a t-nearest neighbor (t-NN) algorithm on the dataset (that is, x_i and x_j are neighbors if x_j is one of the t closest data points in terms of the euclidean norm). Then $D \in \mathbb{R}^{n \times n}$ is a diagonal matrix such that $D_{ii} = \sum_j W_{ji}$ and thus $L = D - W$ is the graph Laplacian matrix. Hence, our projection matrix P is given by the k eigenvectors that correspond to the top k eigenvalues of the following generalized eigenvalue problem:

$$X^\top L X v = \lambda X^\top D X v \tag{8}$$

If $X^\top D X$ is nonsingular, then the LPP objective may be reduced to a regular eigenvalue problem, and by performing an eigendecomposition on $X^\top D X = U S U^\top$, can be made into the following form:

$$A\boldsymbol{y} = \lambda \boldsymbol{y} \tag{9}$$

Where $A = S^{-1/2} U^\top X^\top L X U S^{-1/2}$ and $\boldsymbol{y} = S^{1/2} U^\top$. This derivations avoids the explicit computation of $(X^\top D X)^{-1} X^\top L X$, which may be numerically unstable [4]. Note that in practice for image processing, $X^\top D X$ is often singular (in much the same way that the within class scatter matrix is singular in LDA), and thus SVD PCA can be used to reduce the dimensionality such that $X^\top D X$ is nonsingular before applying LPP. This yields k components that aim to maintain local relationships that we refer to as *LaplaceSalads*. It is important to note that though LPP offers a locality preserving effect that is absent in most other linear dimensionality reduction algorithms, LPP suffers from being more sensitive to its external hyperparameters in terms of accuracy.

2.5 Non-Negative Matrix Factorization (NMF)

One significant benefit of working with image data is that all information in X is strictly non-negative, as values are either intensity values in $[0, 1]$ or alpha/rgb integer values in $[0, 255]$. This allows the technique of Non-Negative Matrix Factorization (NMF) to be used in order to decompose our data matrix into nonnegative components. Specifically, NMF seeks to solve the following optimization problem [7]:

$$\text{argmin}_{W,H} \|X - WH\|_F^2 \quad \text{s.t.} \quad W, H \geq 0 \tag{10}$$

Thus, Eq.10 is minimized when $X \approx WH$, where $W \in \mathbb{R}^{n \times k}$ and $H \in \mathbb{R}^{k \times d}$, and hence W contains our k non-negative components that we refer to as *PosiSalads*. The above objective does not have a closed-form solution, and thus we must resort to iterative optimization techniques (like Gradient Descent) in order to solve the objective. Currently, Coordinate Descent (which solves 1-dimensional optimization problems in each coordinate direction iteratively) is one of the most popular approaches for solving NMF systems, and much research has been invested to improving the stability and computational costs for this method.

3 Salad Theory

[8] posits a wide-encompassing definition of salads that is both simple and avoids any unnecessary ambiguities or accidental exclusions:

Theorem 1 *A salad is any edible meal that contains at least two ingredients.*

Foods that don't fit this criteria are categorized as *low-entropy hyper-salads*. Given this definition, the axes of the HS3P are defined as the following:

1. Soupiness: a real-valued number in $[0, 1]$ that measures the proportion of exterior liquid in the food (i.e. a soupiness coefficient of 0 represents no exterior liquid and 1 represents only liquid)
2. Ingredient Entropy: $i_e = \log_2(\text{# of Ingredients}) \times (\text{# of Unique Ingredients})$. This represents a function of the food's ingredients that is both monotonically increasing with the addition of more ingredients and rewards a variety of ingredients over multiple of the same ingredient.
3. Arrangement Entropy: $a_e = \text{Axes of Ingredient Translation} + \text{Axes of Ingredient Rotation}$. Here Axes of Ingredient Translation represents the average number of axes in 3-D space that ingredients can be translated within the convex hull of the food without a noticeable change, and Axes of Ingredient Rotation is defined similarly for rotations. For example, a Caesar salad would have $a_e = 3 + 3 = 6$, since any ingredient can be both translated and rotated without changing the salad all that much, but a Hamburger would have $a_e = 1.5 + 1 = 2.5$, since ingredients can only be translated to an extent (otherwise they would fall out of the bun) and can only be rotated around the vertical axis.

Now that we have explicitly defined the HS3P as a theoretical framework, we can move into our experimental approximation of HS3P using the aforementioned linear dimensionality reduction techniques.

4 Experiment

4.1 Data

We draw our sample of food images from the Food-101 Dataset [2], which represents 10100 photos of food in 101 different classes (see Fig.1). In order to make the experiment more computationally tractable, we subsample the data and use 1000 food images from 10 different classes (thus, 100 images per class). The images were greyscaled and downsampled into 200×200 pixel images for ease of computation as well.

4.2 Methods

As previously stated, we used the linear dimensionality reduction techniques described earlier according to the following specifications:

- *EigenSalads*: SVD PCA with $k = 8$ components
- *FisherSalads*: SVD PCA with $k = 990$ components, followed by LDA
- *ICAnSalads*: FastICA with $k = 8$ components and $g(x) = -e^{-x^2/2}$
- *LaplaceSalads*: SVD PCA with $k = 1000$ components, followed by LPP with $t = 40$ and $h = 0.5$
- *PosiSalads*: NMF with $k = 8$ components using Coordinate Gradient as its solver

4.3 Results

We observe the top 8 components for each method in figures 2 - 6. We can qualitatively divide our results into those which were able to uncover low-dimensional positional representations of the data (PCA, ICA, and NMF) and those that were not able to produce meaningful basis images (LDA and LPP). Possible justifications for the poor performance

Figure 1: Example Image from Food-101 Dataset

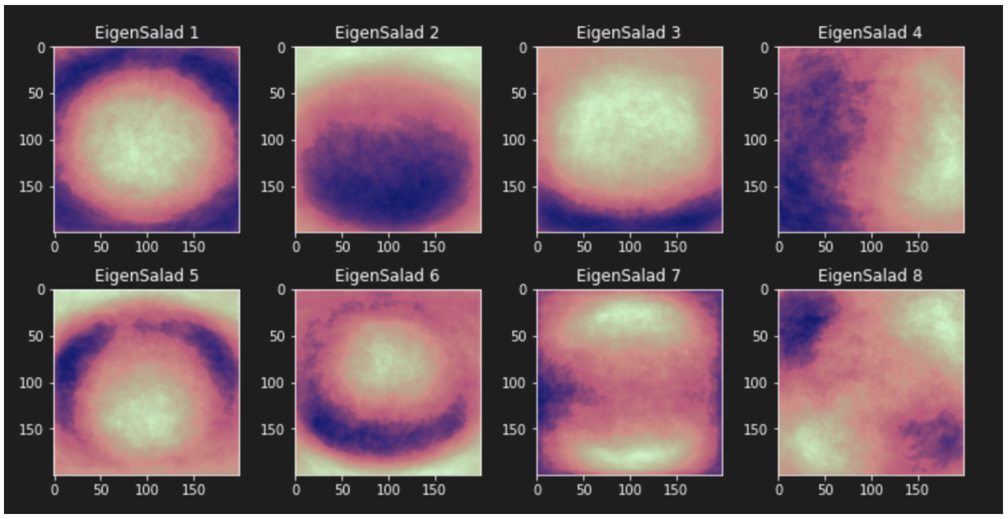

Figure 2: Top 8 EigenSalads (PCA)

of LDA and LPP may stem issues inherent in the dataset itself: the photos lack any sort of consistent lighting or camera angle, and thus there is incredibly high variation at the within-class level (thereby making it hard for LDA to find any discriminating representation of the data), and there is no consistent patterns to be found at the local level (hence explaining the lack of meaningful components in the LPP decomposition). Additionally, both of these methods used SVD PCA as a preprocessing step in order to account for singularity issues in their linear systems, but such a reduction may have gotten rid of a considerable amount of useful information in the image set.

PCA, ICA, and NMF were all able to uncover basis images that seem to coincide with positional data within the images. While NMF basis images are most likely picking up on the same variation in lighting and camera angle that led to poor performance in LDA and LPP (thus making the NMF images more of basis for the images themselves rather than the foods directly, the PCA and ICA bases seem to pick up on some form of finer-grained information. Particularly with EigenSalads 7 and 8 (see figure 2) and ICAnSalads 2, 3, and 5 (see figure 4), these basis images seem to possibly approximate the rotational components that make up part of the arrangement entropy axis of HS3P. Notably absent were any approximations of the soupiness or ingredient entropy axes of HS3P (which might've looked like basis images that were noisier and just represented variation in the image); this may be due to working with greyscaled images, which while traditionally held to improve performance of image processing techniques may have gotten rid of color-based variation in the photos, or due to an actual lack of evidence of either axis of HS3P being empirically valid.

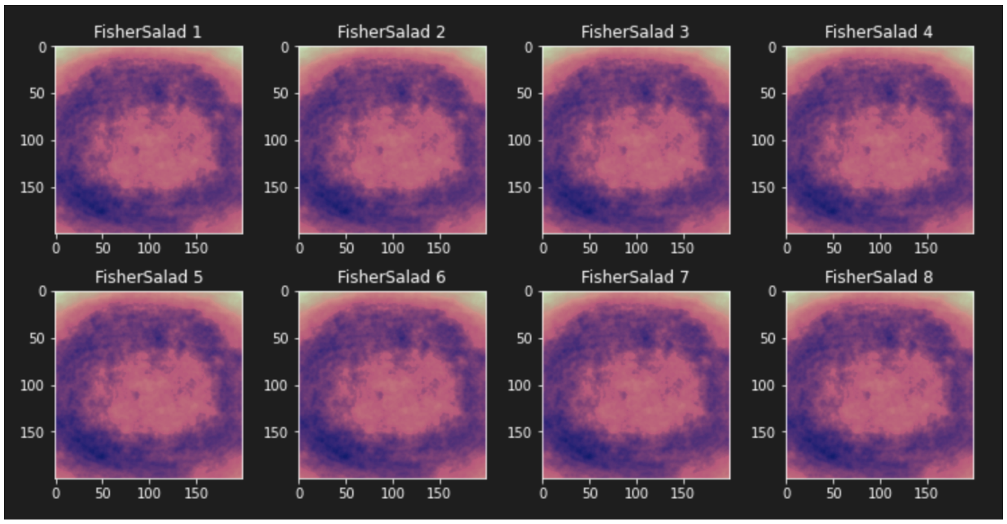

Figure 3: Top 8 FisherSalads (LDA)

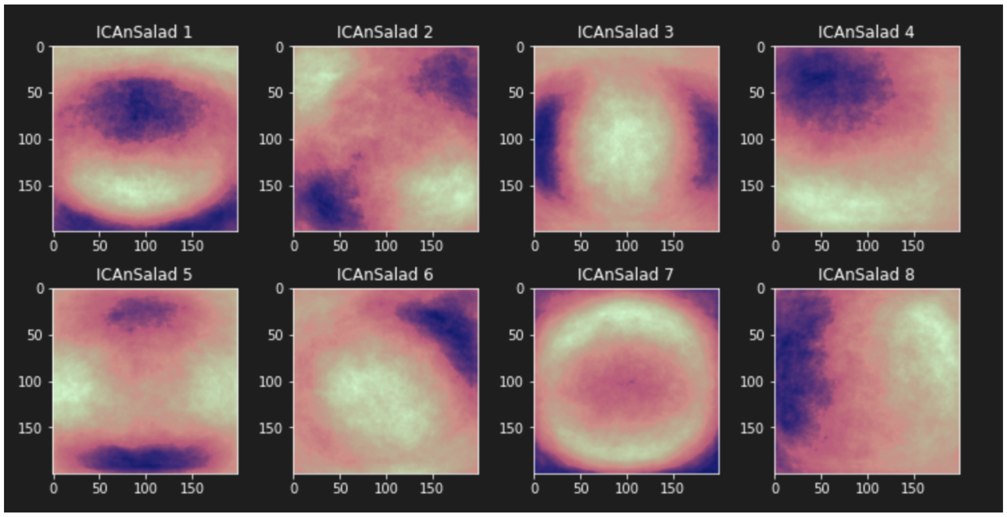

Figure 4: Top 8 ICAnSalads (ICA)

5 Conclusion

Overall, this paper has presented a high-level survey of linear dimensionality reduction techniques that provide qualitatively interpretable, low-dimensional bases for our high-dimensional image data. Empirical validation of HS3P showed some promise, but issues in the dataset added considerable noise to the learning process, which is an inherent limitation of the present study. Further research should be done to pursue a more standardized dataset and to use more sophisticated dimensionality reduction techniques, such as deep Variational Autoencoders (VAEs), that loosen the linearity assumption in order to uncover significantly more fine-grained patterns in the input data.

References

[1] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1997.

[2] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101 – mining discriminative components with random forests. 2014.

[3] N. Halko, P.-G. Martinsson, and J. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *arXiv preprint arXiv:0909.4061v2*, 2009.

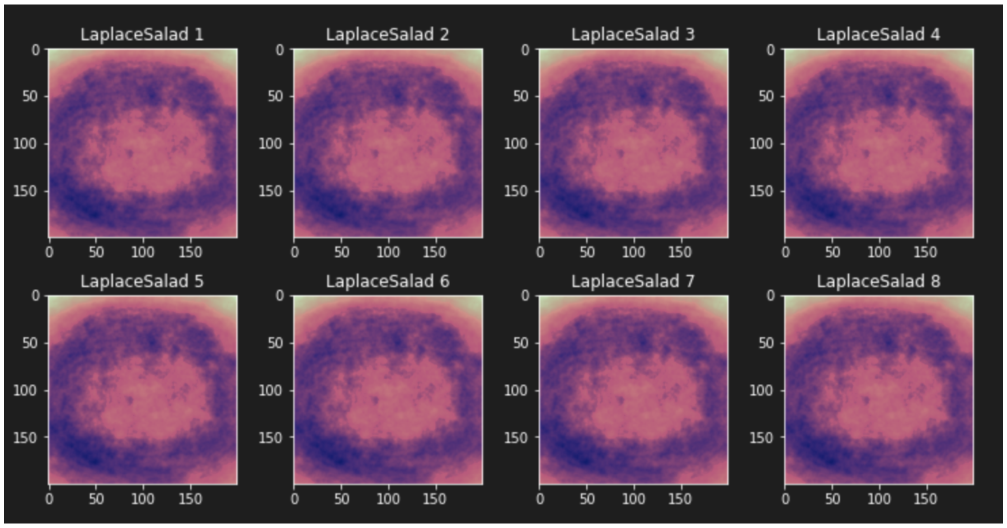

Figure 5: Top 8 LaplaceSalads (LPP)

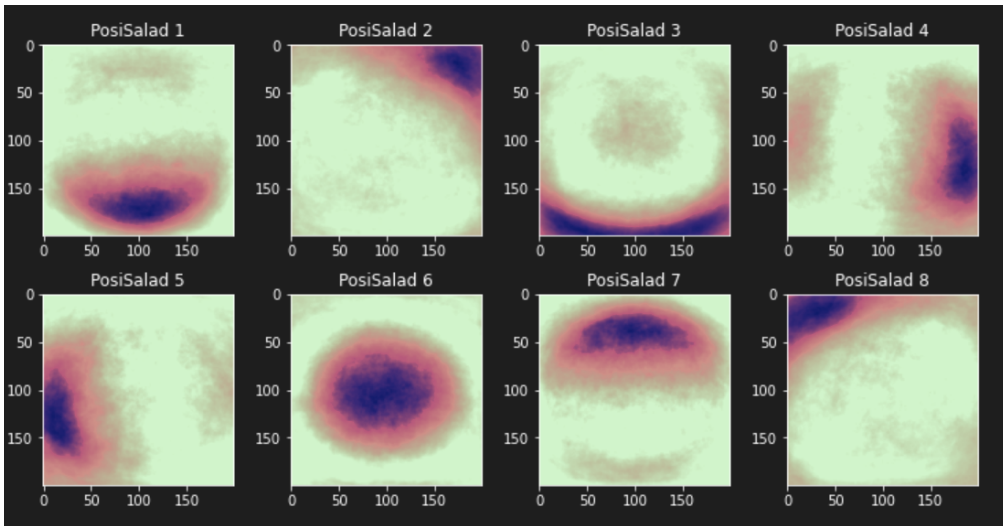

Figure 6: Top 8 PosiSalads (NMF)

[4] X. He. *Locality Preserving Projections*. PhD thesis, University of Chicago, 2005.

[5] X. He and P. Niyogi. Locality preserving projections. 2003.

[6] A. Hyvärinen and E. Oja. Independent component analysis: Algorithms and applications. 2000.

[7] D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. 2000.

[8] G. Malmquist. Salad theory, 2020.

[9] F. Porter and I. Narsky. *Statistical Analysis Techniques in Particle Physics*. 2013.

[10] J. Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100v1*, 2014.

[11] A. Talwalkar and V. Smith. Machine learning with large datasets lecture notes, 2021.

[12] M. Turk and A. Pentland. Face recognition using eigenfaces. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, page 586–591. IEEE, 1991.