SCALABLE PARAMETRIC ENCODING OF MULTIPLE MODALITIES

David Banh
Data Science
AskExplain
St Lucia, Brisbane, Australia
david.b@askexplain.com

Alan Huang
School of Mathematics and Physics
University of Queensland
St Lucia, Brisbane, Australia
alan.huang@uq.edu.au

ABSTRACT

A flexible model is introduced which shares ideas with the Autoencoder, Canonical Correlation Analysis, Singular Value Decomposition, and Procrustes Analysis. It is proposed to find relevant maps to transform multiple datasets of various types from one modality to another. Here, the Generative Encoder is used to transform spatial gene expression from breast tissue, to the images of histology tissue measured with Spatial Transcriptomics. The model is directly interpretable given all parameters are linked to the data space. It is scalable on Big Data, training reasonably on several thousand RGB images of 100 by 100 pixels in under an hour, which equates to 30,000 pixel features per sample image.

***K*eywords** Data Transformation, Multi-modal data, Scalable, Big Data, Spatial Transcriptomics, Autoencoder, Singular Value Decomposition, Canonical Correlation Analysis, Procrustes Analysis

1 Introduction

Data transform functions aim to map a set of points from one space to another space more useful for further analysis. Transformations are of immense utility, from the kernel trick in Support Vector Machines using a Gaussian transformation [1], to the non-linear composition of functions in neural networks. [2].

Ideas from linear algebra can also have functions acting as transformations. One involving Matrix Projection [3]: used for projecting one matrix onto another second vector or matrix such as in regression, and another being the traditional Procrustes Analysis [4]: used for matching shapes represented by point clouds via matrix rotations and scaling.

State of the art methods that evaluate both the sample and feature spaces appear recently. In 2021, Zhou and Troyanskaya formulate the GraphDR method which projects to a latent space via linearly interpretable parameters that operate on both the samples and features [5]. In 2019, Tarashansky et al with Self-Assembling Manifolds (SAM) use a nearest-neighbour projection method that operates on the samples and features to generate a manifold on the points for further dimensionality reduction and visualisation. [6].

These methods show that the sample space holds vital information that describes nearest-neighbours in a local space, with recent developments also showing the importance of mutual data neighbourhoods [7]. Combining both the data space of samples and features using the ideas of Mutual Nearest-Neighbours with Canonical Correlation Analysis respectively, has led to recent developments in multi-modal integration in single cell analysis [8].

The method developed and proposed in this investigation is similar to the quasi-linear method GraphDR, where model design is chosen such that parameters operate on both the sample and feature spaces. The parameters learn functions which encode both the sample and feature spaces of any dataset, fitting any multiple number of datasets into a shared encoded subspace. From only the learned parameters, the dataset can be decoded to be fully expressed as a generative model. The mapping from the dataset to a lower dimensional space via encoding and mapped back to the original dataset via decoding gives it the style of a Generative Encoder.

2 Method

The aim of Generative Encoding is to integrate data from several datasets, Y, X and A, into an aligned and reduced dimensional space d via a transformation. An example of Generative Encoding that is implemented here:

$$\alpha_{Y_{k,i}} Y_{i,l} \beta_{Y_{l,c}} = \alpha_{X_{k,i}} X_{i,j} \beta_{X_{j,c}} = \alpha_{A_{k,i}} A_{i,d} \beta_{A_{d,c}}$$

Here, k and c are the dimensions of the reduced dimensional encoding. The sample size i is shared between the datasets Y, X, and A. However, each dataset contains differing feature sizes l, j, d. For example, this could be of a single cell experiment where each dataset contains the profile of gene expression, protein and chromatin accessibility, respectively for Y, X and A.

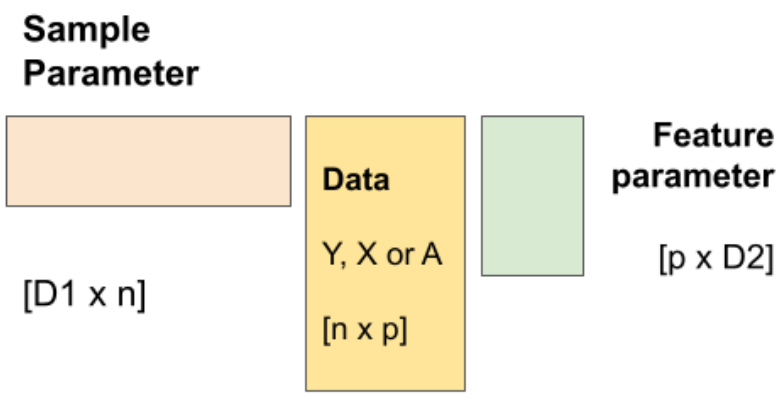

Figure 1: Encoding the dataset using two sets of parameters - an encoding of the sample space and an encoding of the feature space. Multiple sets of data can be aligned to the same space, provided there is a proportional relationship between the two encoding spaces, for example, an equality

The α parameter transforms the samples into the same reduced dimensional plane along the sample space. Furthermore, the β parameter transforms the features across the datasets into the same space. The matrix αX are the **encoded features** where the sample space has been encoded leaving only the features in the original space. The matrix $X\beta$ are the **encoded samples** where the feature space has been encoded leaving only the samples in the original space.

2.1 Further details of the Generative Encoder

Inspection of the parameters and how they are linked can be taken a further step. By expressing Y, X, and A as a function of the parameters, it is possible to extract the core common element shared between all datasets called the code. For example, let the code be Z of reduced dimension k by c.

For Y the decoded estimate is $\hat{Y}$,

$$\hat{Y} = \alpha^T_{Y_{i,k}} Z_{k,c} \beta^T_{Y_{c,l}}$$

Likewise, X can be recovered by the decoded estimate $\hat{X}$

$$\hat{X} = \alpha^T_{X_{i,k}} Z_{k,c} \beta^T_{X_{c,j}}$$

Finally, A can be recovered by the decoded estimate $\hat{A}$

$$\hat{A} = \alpha^T_{A_{i,k}} Z_{k,c} \beta^T_{A_{c,d}}$$

These expressions using the code Z can recover the original datasets, with the key important concept that the latent space shared amongst the datasets are given as Z.

Furthermore, it can be limiting to consider the datasets as matrices - they can be of any size, including three dimensional objects or tensors.

2.2 Properties of the Generative Encoder

Functions and matrix properties : The encoding can be considered a learned set of functions which operate on the data modalities. Let α and β represent $g(.)$ and $f(.)$ respectively, such that going from left to right:

$$\alpha X \beta = f(g(X))$$

Note it is important to learn the optimisation computationally such that

$$f(g(X)) = \alpha X \beta = g(f(X))$$

Which is the property of associativity.

Samples can be encoded : Sample sizes can be reduced and thus save computational resources when running prediction or nearest-neighbour algorithms. For example, by first reducing the large sample size (several million data points) to a smaller encoded set (several thousand), an algorithm can be run on the smaller set and then decoded back into the original space.

Data can be recovered and imputed : Given all data points can be fully expressed by the model, then iterating over updates of the model while simultaneously updating any missing data points can recover the structure of the data and impute missing points. This is similar to the concept of a bootstrap, yet for prediction.

Transformations can be transferred : If of similar sample ID the sample parameter α can be transferred, if of similar feature label the feature parameter β can be transferred. For this reason, a transform learned to reduced the dimensions from a set of genes can be trained on one dataset, and transfered to the second dataset, provided the gene list is identical.

Parameters can be joined : If multiple datasets share the same sample ID, the sample parameter can be shared. Otherwise, if multiple datasets have overlapping feature labels, the feature parameter can be shared. This provides an extra level of interpretability.

2.3 Coordinate descent updates for learning parameters

The coordinate descent update to estimate the parameters, iterates through multiple steps outlined by Algorithm 1 until convergence. The full step loops over all datasets, and is run in a while loop within gcode.

Algorithm 1 Generative Encoding via Generalised Canonical Procrustes

Input: D_L Dataset for each modality L
Output: α_L sample parameters, β_L feature parameters, Z code

procedure COORDINATE DESCENT UPDATE(D_L)
  for L in D_L **do** ▷ For each dataset
    $\alpha_L = D_L^T (Z\beta_L^T)^T ((Z\beta_L^T)(Z\beta_L^T)^T)^{-1}$ ▷ Update sample parameters
    $\beta_L = ((\alpha_L^T Z)^T (\alpha_L^T Z))^{-1} (\alpha_L^T Z)^T D_L$ ▷ Update feature parameters
    $Z = (\alpha_L^T \alpha_L)^{-1} \alpha_L^T D_L \beta_L^T (\beta_L \beta_L^T)^{-1}$ ▷ Update Code

2.4 Why learn in a subspace?

Learning in a reduced subspace enables more computational efficiency as the parameters are learned in the smaller reduced dimension. Furthermore, when the dimensions of both sample and features are encoded down into a reduced dimension, it is possible to learn an algorithm on the encoded data, and then project the learned information back up into the original space via a decoding - with results comparable to when the algorithm is run on the full samples.

3 Numerical Results

The results contains two parts - a run on single cell data, and, an integration of histology images with RNA-seq data:

3.1 Clustering and Trajectory of RNA from single cells of adult mouse hippocampus

Generative encoding summarises information by encoding the sample space into a reduced dimension (in addition to encoding the feature space). Doing so reduces the samples to a smaller dimension, capturing and then representing only the most important structures in the data. This reduces variability and concentrates points to identifiable clusters in visualisation dimension reduction tools such as UMAP or t-SNE.

Here single cell data from an adult mouse hippocampus has had transcriptome measured at a nuclei level [9]. It is expected for both visualisations: 1 - with, and 2 - without gcode; to represent the anatomical structure of the adult mouse hippocampus.

In Figure 2, cell types are more visible - which is expected given gcode encodes the sample points to refine structure. Clusters of points exhibit the finer structures of cell types: clustering well with the upper anatomical cell labels, and still exhibiting neighbouring proximity to similar cell types in the lower level anatomical annotations.

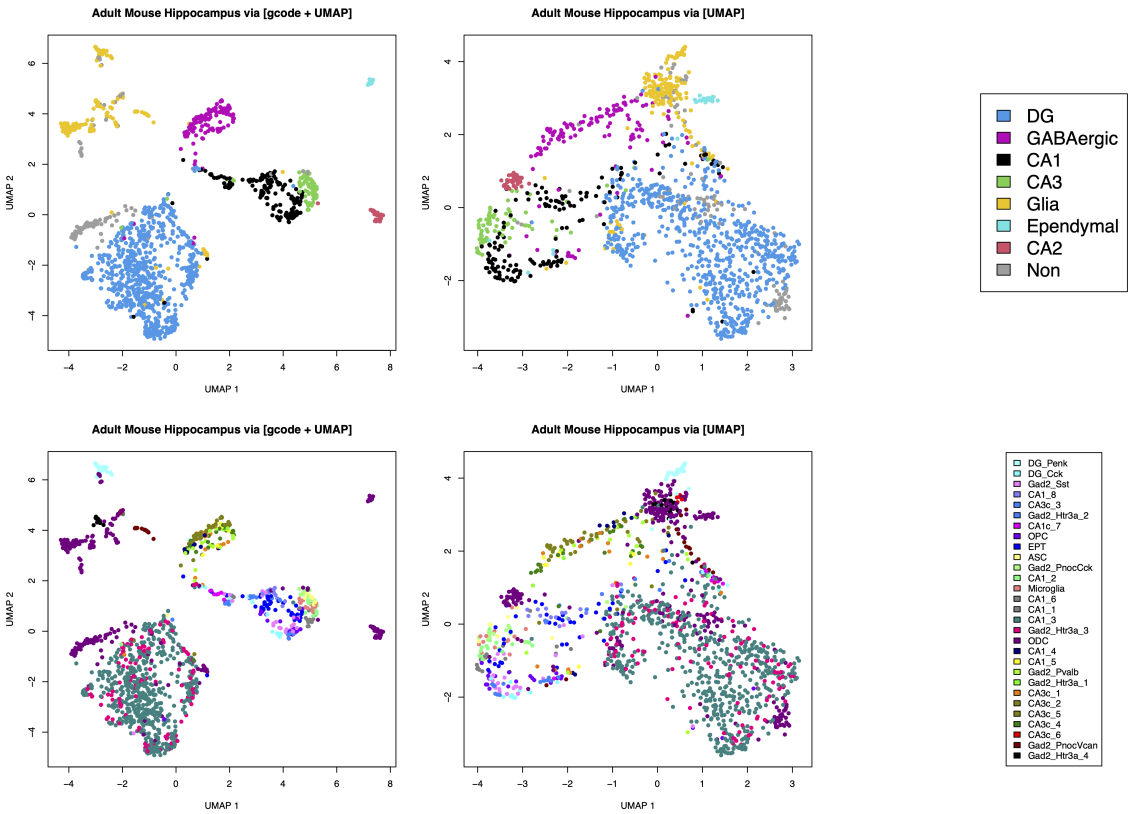

Figure 2: UMAP plots comparing effects of gcode (left) and without (right). Cell type labels are closer neighbours and more visible as clusters for both upper and lower cell annotations.

3.2 Transforming histology to gene expression

Tumourous Breast tissue was taken from He et al, where histological samples were measured with gene expression using Spatial Transcriptomics [10]. Generative Encoding was used to find a transformation between the spatial gene expression and the tissue histology of the Spatial Transcriptomics dataset.

In Figure 4, histology is transformed to gene expression. This idea of finding gene expression values from histology images using spatial cell structure and morphology from Spatial Transcriptomic data is also shared with the work in stNet and XFuse [10] [11].

To begin, let E represent the spatial transcript data, and, H to indicate the histology pixel matrix information. Now, given the Generative Encoder relation:

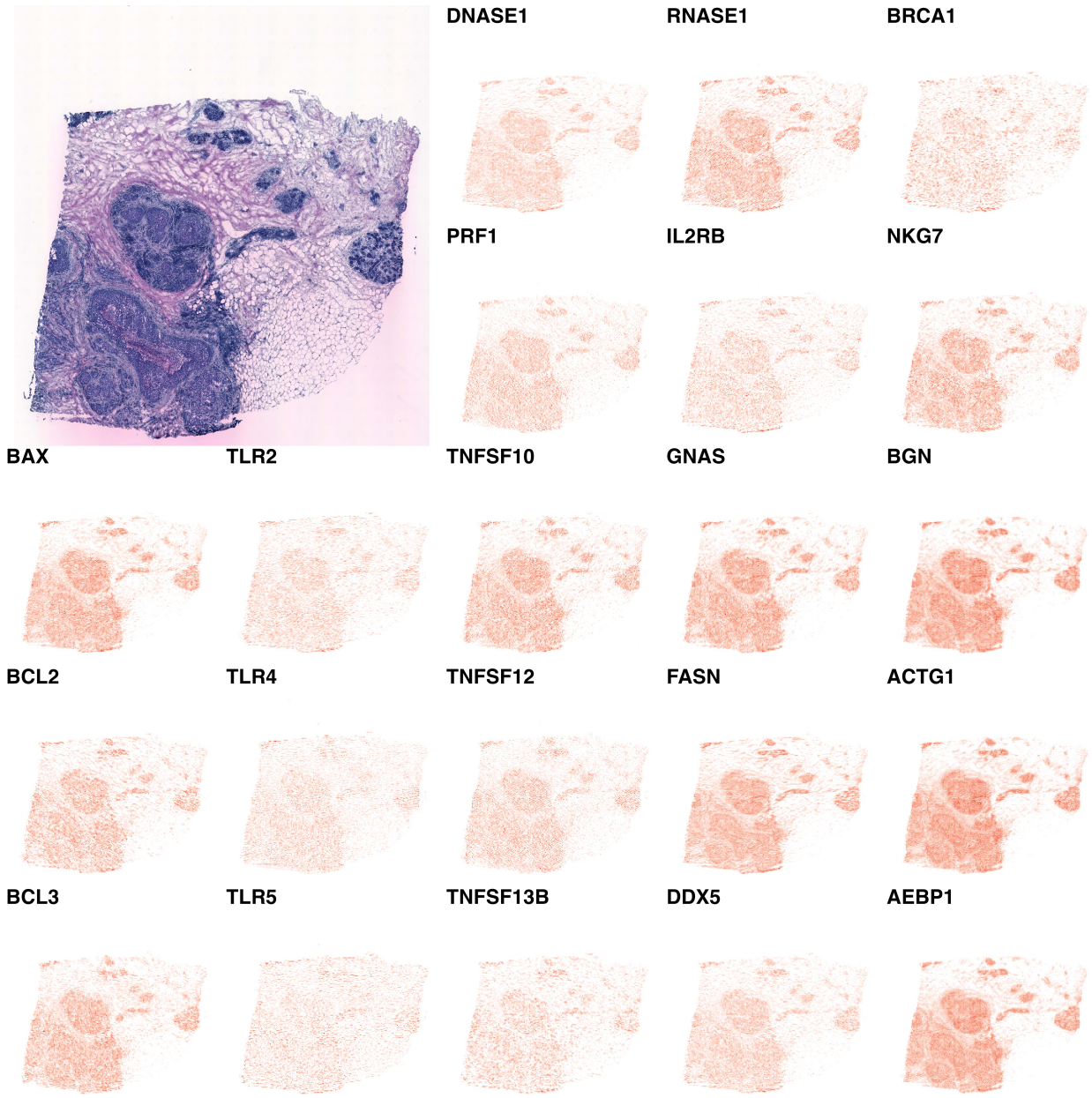

Figure 3: Tumourigenic breast tissue with histology measured using Spatial Transcriptomics accompanied by expression heatmaps of genes. Importantly, genes have been transformed to a standard normal such that relative expression level between genes are at different scales.

$$\alpha_H H \beta_H = \alpha_E E \beta_E$$

It can be considered that the sample space (represented by α is identical between spots of histology pixels, such that it no longer contains relevant information and can be ignored:

$$H\beta_H \propto E\beta_E$$

From here, using this relation it is possible to write H as the full expression:

$$H = \alpha_H^T Z \beta_H^T$$

Putting the full expression into the relation for H, and the equivalent full expression for E gives:

$$\alpha_H^T Z \beta_H^T \beta_H \propto \alpha_E^T Z \beta_E^T \beta_E$$

Following, the pixel and gene effects are removed as $\beta^T \beta$ is orthogonal when the model is initialised via truncated SVD (irlba), and the model is designed to maintain the orthogonality of the parameters:

$$\alpha_H^T Z \propto \alpha_E^T Z$$

Finally, $E = \alpha_E^T Z \beta_E$ can be regenerated on the right hand side by multiplying with β_E:

$$\hat{E} = \alpha_H^T Z \beta_E \propto \alpha_E^T Z \beta_E = E$$

This gives the relation and the eventual transformation of the histology H to the expected expression of the histology $\hat{E}$, which is directly proportional to E. Through $\hat{E} = \alpha_H^T Z \beta_E$, the samples of $\hat{E}$ are described by α_H, the histology sample parameter.

The large histology image slide is tiled into a space interval of 100 pixels, with each tile of 100 x 100 RGB pixels. These are then transformed to gene expression vectors, whereby heatmaps can be generated. The heatmap is plotted such that the expression is scaled first by a z transform along the genes to a standard normal distribution:

$$E_z = (E - \mu)/\sigma \sim N(0, 1)$$

Where E indicates gene expression data and E_z indicates the scaled gene expression.

Following on, a relative transform t by the minimum and maximum such that the lowest expression value is closest to -1 and the highest expression value is closest to +1, although, not necessarily equal to -1 or +1. By first transforming to a standard normal, it allows different genes to be compared relative to each other according to their distributions:

$$t(E_z) = E_z/min(E_z) \quad E_z < 0$$
$$t(E_z) = E_z/max(E_z) \quad E_z > 0$$

Notice in Figure 4 that the detailed patterns exhibited by gene heatmaps have a diverse, variable, and highly detailed structure. When comparing genes that are expressed in extracellular matrix - BGN, ACTG1 or AEBP1, notice the patterns are very smooth near dense regions of cells. The BCL and TLR gene families show a rough patterning, with small striped lines going mostly in a single direction. Other genes representing Natural Killer cells, PRF1, IL2RB1, and NKG7 are sparser and reveal spotted regions of activity with other regions of inactivity.

3.3 In-silico perturbation of histology tissue with gene expression mappings

Histology images can be transformed to gene expression vectors for each image, followed by a perturbation to alter the gene expression level relative to each gene by a factor of the gene expression standard deviation representing the density required for approximately 25% to cover a two-tailed standard normal distribution.

A transformation similar to the histology to gene expression projection is used from the previous section.

$$f(H) \propto \hat{E}$$

For any given gene s in a gene set G, the perturbation K_{G^s} is calculated via ϕ^{-1} which is the inverse cumulative distribution function of a normal distribution with probability $p = 25\%$.

$$K_{G^s} = \phi^{-1}(\mu = 0, \sigma = 1; p = 0.25)$$

The final perturbation on genes G^s is a addition or subtraction by $K_{G^s}\sigma_{\hat{E}_{G^s}}$, where σ is the standard deviation,

$$\bar{E} = \hat{E}_{G^s} \pm K_{G^s}\sigma_{\hat{E}_{G^s}}$$

Thereafter, the perturbed expression $\bar{E}$ is transformed back into a histology image. This step follows the same processes in the first experiment when converting histology to gene expression although with the inverse of the transform.

$$H = f^{-1}(f(H)) \propto f^{-1}(\hat{E})$$

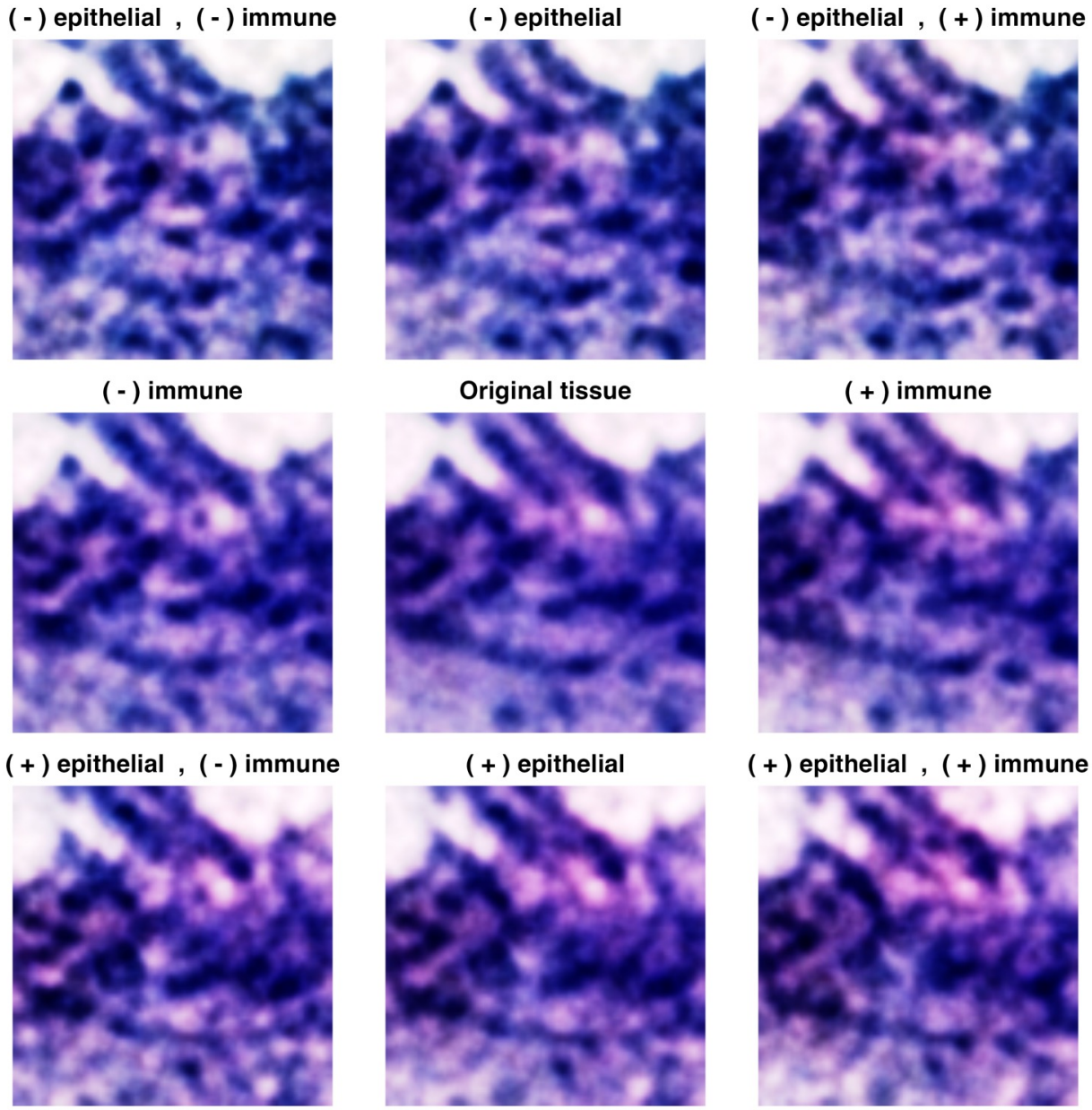

Figure 4: A single image of a H&E stained histology tissue capturing cells in dark blue globules, and extracellular matrix in shades of purple. The center plot shows the original histology image. The vertical and horizontal axis shows the histology image altered by perturbing the expected gene expression of the image (histology transformed to gene expression vector, perturbed, and then transformed back to histology image). From top to bottom shows increasing expression of epithelial gene markers, whereas left to right shows increasing expression of immune gene markers.

Notice in Figure 4, with increasing epithelial gene expression, the tissue is increasingly marked for a pink eosin stain. Epithelial and immune marker genes are shown in the Supplementary.

4 Related Work

4.1 Similarities to Singular Value Decomposition

When the model is expressed as $\alpha Y \beta$ it is similar to a Procrustes Analysis. However, the detailed model outlined in further detail (Section 2.1), $Y = \alpha^T Z \beta^T$ shows an expression depicting Singular Value Decomposition.

4.2 Similarities to Procrustes

In order to align two sets of shapes or points within a dataset, the method centers and scales the points within each set, and then takes a reference set upon which the other set is rotated and fit to. The expression for the centering and rescaling with rotation is : $Y_{i,l} = k_{k,i} X_{i,j} Q_{j,c}$ [4].

4.3 Similarities to Canonical Correlation Analysis

The aim is to find a set of component vectors for each of the reference and the non-reference dataset which maximise the correlation. The generative encoder borrows ideas, by finding sets of parameters for each dataset that maximises the similarity of the structures of the encoded datasets.

4.4 Similarities to Neural Networks and Autoencoders

Deep Learning models are optimisation methods that use a non-linear composition of weights to minimise a loss function [2]. The style of using many weights that scales with both features and samples to analyse images is capable with both Deep Learning and Generative Encoding.

5 Conclusion

Encoding datasets into a shared subspace enables the simultaneous analysis of all datasets. The analysis is done through learned data transformations that are parameters which enable a functional mapping between modalities.

The properties of the Generative Encoder are shown in the summarising ability to concentrate signal and reduce spurious noise. The data transformation from Generative Encoding is evaluated by mapping between histology and gene expression. An example of the mapping is the transform from images of a histology tissue specimen into a gene expression heatmap at a detailed level. Another example is the perturbation of a given histology image, by perturbing the expected gene expression of any given tissue image.

The study shows the capacity of Generative Encoding to be an interpretable analysis tool for biologists and in other disciplines where there is a need to make use of many datasets of different modalities.

6 Code and Data

The GitHub version of the gcode package: `https://github.com/AskExplain/gcode/`

The GitHub scripts used to output figures and analyses: `https://github.com/AskExplain/gcode_analysis`

More figures on the in-silico perturbation and transforamtion of gene expression to histology: `https://doi.org/10.5281/zenodo.5851702`

The GitHub verison of R for gcproc (previous version of gcode): `https://github.com/AskExplain/gcproc/`

A preliminary GitHub version of Python for gcproc (previous version of gcode): `https://github.com/thisismygitrepo/gcprocpy`

A dashboard detailing ideas on the role of Generative Encoding with Spatial Transcriptomics: `https://board.askexplain.com/genecode`

7 Acknowledgements

David Banh would like to acknowledge:

1 - Ryan Deslandes (University of Queensland, Australia) for assistance with software development of the genecode dashboard: `https://board.askexplain.com/genecode`. This idea was presented

in the NanoString Hackathon earning third place and bonus prize for best images (first and second place went to the teams from Dr Omer Bayraktar's Lab from the Wellcome Sanger Institute, one of which presented cell2location).

2 - Cameron Gordon and Olivia Ou (University of Adelaide, Australia and University of Queensland, Australia) for consistent support and advice throughout the project.

3 - Alex Alsaffar (University of Queensland, Australia) for relevant discussions on the model of Generalised Canonical Procrustes and Generative Encoding as well as work on a preliminary Python version of gcproc (a prior version of gcode): `https://github.com/thisismygitrepo/gcprocpy`.

4 - Dr Quan Nguyen (University of Queensland, Australia) for a brief discussion on the model of corevec (a prior version of gcproc, which is an earlier version of gcode) regarding imputation and alignment in early 2021 https://github.com/AskExplain/corevec

8 Supplementary

Epithelial and Immune gene sets that are altered in the in-silico perturbation of histology images are:

Gene List	
Epithelial	ACTA2, AGRN, ANXA1, ANXA11, BMP1, BMP2K, BMP8B, BMPR1A, BMPR2, BRCA1, CAV1, CD44, COL10A1, COL12A1, COL14A1, COL15A1, COL16A1, COL18A1, COL1A1, COL1A2, COL27A1, COL3A1, COL4A1, COL4A2, COL4A5, COL5A1, COL5A2, COL5A3, COL6A1, COL6A2, COL6A3, COL7A1, COL8A1, COL8A2, COL9A3, COLEC12, COLQ, EFEMP1, EFEMP2, FBLN1, FBLN2, FBLN5, FBLN7, FBN1, FKBP10, FKBP11, FKBP14, FKBP15, FKBP1A, FKBP1C, FKBP2, FKBP3, FKBP4, FKBP5, FKBP7, FKBP8, FKBP9, FKBPL, FN1, HSP90AA1, HSP90AB1, HSP90B1, HSPA12B, HSPA13, HSPA14, HSPA1B, HSPA2, HSPA4, HSPA4L, HSPA5, HSPA6, HSPA8, HSPA9, HSPB1, HSPB11, HSPB7, HSPB8, HSPBAP1, HSPBP1, HSPD1, HSPE1, HSPG2, HSPH1, IGF1, IGF1R, IGF2, IGF2BP1, IGF2BP2, IGF2R, IGFBP2, IGFBP3, IGFBP4, IGFBP5, IGFBP6, IGFBP7, IGFLR1, ITGB2, LAMA2, LAMA4, LAMA5, LAMB1, LAMB2, LAMB3, LAMC1, LAMC2, LAMP1, LAMP2, LAMP5, LAMTOR1, LAMTOR2, LAMTOR3, LTB, LTB4R, LTBP1, LTBP2, LTBP3, LTBP4, LTBR, MGP, RSPH10B, RSPH10B2, RSPH3, RSPH9, RSPRY1, SMOC2, TGFB1, TGFB1I1, TGFB2, TGFB3, TGFBI, TGFBR1, TGFBR2, TGFBR3, TGFBRAP1, VCAM1, VCAN, VIM, WNT2, WNT2B, WNT9A, ZP3
Immune	CCL11, CCL2, CCL22, CCL3, CCL3L3, CCL4, CCL4L2, CCL5, CCR1, CCR10, CCR7, CD109, CD14, CD151, CD163, CD164, CD180, CD1D, CD2, CD209, CD22, CD248, CD27, CD276, CD2AP, CD2BP2, CD300A, CD302, CD320, CD33, CD34, CD36, CD37, CD3D, CD3E, CD4, CD40, CD44, CD46, CD47, CD48, CD5, CD52, CD53, CD55, CD58, CD59, CD6, CD63, CD68, CD69, CD72, CD74, CD79A, CD79B, CD81, CD82, CD83, CD84, CD86, CD8A, CD9, CD93, CD99, CD99L2, DUSP1, DUSP10, DUSP11, DUSP12, DUSP14, DUSP16, DUSP18, DUSP2, DUSP22, DUSP23, DUSP28, DUSP3, DUSP4, DUSP5, DUSP6, DUSP7, DUSP8, FCGBP, FCGR1A, FCGR1B, FCGR2A, FCGR2B, FCGR2C, FCGR3A, FCGR3B, FCGRT, HLA-A, HLA-B, HLA-C, HLA-DMA, HLA-DMB, HLA-DOA, HLA-DPA1, HLA-DPB1, HLA-DQA1, HLA-DQA2, HLA-DQB1, HLA-DQB2, HLA-DRA, HLA-DRB1, HLA-DRB5, HLA-E, HLA-F, HLA-G, IL10RA, IL10RB, IL11RA, IL12RB1, IL13RA1, IL15RA, IL16, IL17RA, IL17RB, IL17RC, IL17RE, IL18BP, IL1R1, IL1RN, IL21R, IL27RA, IL2RB, IL32, IL3RA, IL4I1, IL4R, IL6R, IL6ST, IL7R, ILF2, ILF3, ILK, ILKAP, ILVBL, JUN, JUNB, JUND, NKAP, NKD2, NKG7, NKIRAS1, NKIRAS2, NKRF, NKTR, NKX6-2

References

[1] Thomas Hofmann, Bernhard Schölkopf, and Alexander J Smola. Kernel methods in machine learning. *The annals of statistics*, 36(3):1171–1220, 2008.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[3] Cinzia Viroli. On matrix-variate regression analysis. *Journal of Multivariate Analysis*, 111:296–309, 2012.

[4] Colin Goodall. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(2):285–321.

[5] Jian Zhou and Olga G. Troyanskaya. An analytical framework for interpretable and generalizable single-cell data analysis. *Nature Methods*, 18(11):1317–1321, Nov 2021.

[6] Alexander J. Tarashansky, Yuan Xue, Pengyang Li, Stephen R. Quake, and Bo Wang. Self-assembling manifolds in single-cell rna sequencing data. *eLife*, 8:e48994, Sep 2019.

[7] Laleh Haghverdi, Aaron T. L. Lun, Michael D. Morgan, and John C. Marioni. Batch effects in single-cell rna-sequencing data are corrected by matching mutual nearest neighbors. *Nature Biotechnology*, 36(5):421–427, May 2018.

[8] Tim Stuart, Andrew Butler, Paul Hoffman, Christoph Hafemeister, Efthymia Papalexi, 3rd Mauck, William M, Yuhan Hao, Marlon Stoeckius, Peter Smibert, and Rahul Satija. Comprehensive integration of single-cell data. *Cell*, 177(7):1888–1902.e21, Jun 2019.

[9] Habib Naomi, Li Yinqing, Heidenreich Matthias, Swiech Lukasz, Avraham-Davidi Inbal, Trombetta John J., Hession Cynthia, Zhang Feng, and Regev Aviv. Div-seq: Single-nucleus rna-seq reveals dynamics of rare adult newborn neurons. *Science*, 353(6302):925–928, Aug 2016.

[10] Bryan He, Ludvig Bergenstråhle, Linnea Stenbeck, Abubakar Abid, Alma Andersson, Åke Borg, Jonas Maaskola, Joakim Lundeberg, and James Zou. Integrating spatial gene expression and breast tumour morphology via deep learning. *Nature Biomedical Engineering*, 4(8):827–834, Aug 2020.

[11] Ludvig Bergenstråhle, Bryan He, Joseph Bergenstråhle, Xesús Abalo, Reza Mirzazadeh, Kim Thrane, Andrew L. Ji, Alma Andersson, Ludvig Larsson, Nathalie Stakenborg, Guy Boeckxstaens, Paul Khavari, James Zou, Joakim Lundeberg, and Jonas Maaskola. Super-resolved spatial transcriptomics by deep data fusion. *Nature Biotechnology*, Nov 2021.