Vision Transformer-Based Autoencoder for dimensionality reduction of single-cell RNA sequences

Johan Lindqvist
Department of Computer Science
University of Washington
Seattle, WA 98195
lindq2@cse.uw.edu

Abstract

The project explores the creation of a vision transformer based architecture for single cell RNA analysis and downsampling. The results show that the implemented model (scViT) exhitibts poor performance compared to more commonly used methods such as scVI and outlines some further experiments which may be performed in order to improve the model performance.

1 Introduction

1.1 Single cell RNA sequencing

Over the past decade single cell RNA sequencing [8] technologies have greatly improved the ability for creating detailed measurements of the occurrences of various genes on the single cell level. These developments has lead to a great rise in the amount of publicly available data, however the vast scale of the datasets presents some issues for computational downstream analysis tasks.

Several methods have been developed to reduce the dimensionality of the scRNA data in order to simplify analysis. Frequently used methods include both statistics based tools, such as Principal Component Analysis (PCA)[14], and models leveraging machine learning methodologies, for example DCA [5] and scVI [6].

1.2 Statistical dimensionality reduction

PCA [14] is a commonly used linear dimensionality reduction technique where the attibutes of an input matrix are calculated in order to find which components are most critical for the variance of the data.

For downstream analysis tasks an issue faced by PCA-based pre-processing is that it is calculated across the entire dataset in a single computation which places high demands on both memory and compute [17]. For larger datasets this method can become unfeasible to compute and therefore there is a need for alternate methods. Another issue is that as the values are calculated for one specific dataset it can be difficult to transfer that information to other datasets which limits the usefulness of downstream analysis models as they would have to be re-calibrated for the new principal components.

Machine learning-based approaches, on the contrary, can be "trained" by receiving subsets of the dataset. By encoding the downsampling method through learned parameters, these approaches can be applied to any subset of the data without needing to load the entire dataset into memory simultaneously.

This trait enables the ability that large models can be trained on massive datasets, to create what has been termed Foundation Models [3]. These type of models have proven wildly successful in the fields of computer vision and natural language processing which has lead to an growing interest in creating similar models to process scRNA data. Some examples being scBERT [22] and scFoundation [7]. Developing a architecture that is able to store large amounts of learned information is the key to developling a strong foundation model and is a relevant task for future studies.

1.3 Autoencoders

For the downsampling task one variant of neural networks, Autoencoders [1], have been successful for effectively reducing the size of datasets. These types of models consist of two main components, an encoder and a decoder. The encoder reduces the dimensionality of the data by incrementally reducing the size of each layer before reaching a sufficiently small latent space. The objective of the latent space is to provide a learned representation of the complete dataset. The decoder receives this latent space as an input and from the latent space recreates the input data. Using this network structure the model is forced to learn an efficient representation of the data. Additionally variance the latent space can be a distribution, which allows for variance of the data and which may be used to generate synthetic samples. It is based on this paradigm that both scVI and DCA are implemented.

1.4 Vision transformers

A key innovation in terms of model architecture which has found great success in recent years is the transformer. Originally described in the landmark paper Attention is all you need [18], transformers were designed to handle sequences of text and use the attention mechanism to allow for elements over the entire sequence to influence each other. This algorithm has proven remarkably successful in finding correlations between the different elements in the sequences in a computationally efficient manner.

Vision Transformers (ViT) is an adaptation of the transformer architecture used for computer vision[4]. These models are able to leverage the strengths of the transformers with the aim of aiding the model to extract important correlations across the entire image. This has been seen as a weakness with the previously standard implementation of computer vision network based on convolutional layers. Convolutional layers are generally more limited in what area of the image they may operate because of the design of the kernel size and are therefore more suited for tasks were locality is most important for predictive accuracy.

As transformers naturally operate on sequences through the attention mechanism, the image is converted into a sequence by dividing it up into patches. Each patch consists of a subsection of the image and through the patching process as well as linear operations the image representation can be reduced in size to force it to summarize the valuable information in the image.

This model architecture has been very successful in recent years, especially when combined with convolutional layers, as it is able to both learn close range and global information across images. Relevant for the study of single cell data it furthermore shows that transformers can achieve good results even when the data does not naturally form a sequence.

2 Model architecture

The scViT model is constructed as an autoencoder, which can be divided into three components: An initial patching module, an encoder and a decoder. The data is fed through these layers sequentially with the latent space being constructed following the encoder component. Prior to input the genes needs to be sorted to establish a standardized order in which the genes are placed. This process is necessary when working with different datasets which may not contain the same genes in the same order, requiring a process to align each gene in the matrix with the matching weights of the model. The output of the network is a zero-inflated negative binomial distribution.

The model was built upon the Pytorch [15] framework for the ease of implementation and effective implementations of the necessary learning algorithms. The model further uses modules from Pyro [**a**]nd Torchvision for more specialized components compatible with Pytorch. The data managment was handled through the use of scanpy [20] and anndata [19], both of which are mature frameworks for the reading and management of the h5ad files used for storing single-cell transcriptomic data.

2.1 Patching component

The purpose of this module is to split the input gene data into patches by using a series of convolutional layers and was inspired by the paper Early Convolutions Help Transformers See Better[21]. As vision transformers suffer more significantly in terms of performance when applied with suboptimal hyperparameters the aim of this module is to utilize an convolutional stem to improve the robustness of the model. As compute limitations restrict the scope of the hyperparameter tuning this component should limit the potential damage of naively chosen hyperparameters and improve learning stability.

2.2 Encoder

The encoder of the model is the component which reduces the size dimensionality of each element. As input it receives a series of patches from the patching component. The patches are then downsampled using a linear operation applied per patch such that each patch has a unique set of weights associated with them. These form a "patch-wise linear" layer. This is different to the more commonly used method of applying a single set of learned weights to all of the patches which is generally used for vision transformer architectures. The reasoning behind this change is that compared to image patches there is no inherent similarity in terms of data for all patches. While images patches all consists of pixels which depict a specific subsection of an image following with correlations of locality, the patches created by the patching component do not necessarily share the same degree of similarity between patches. Weights learnable by patch allow for a more adjustable network which should more accurately be able to find relevant correlations within specific to each patch.

The change does create far more learnable parameters which comes at a cost of increased compute and memory requirements.

Following the downsampling the SELU [11] activation function is applied to each of the values in the patches.

The downsampled patches are then put through a multihead-attention [18]. This layer allows the different patches to attend to information present in the other patches by calculating the self-attention over the sequence of patches.

After a series of patchwise-linear and encoding layers has sufficiently reduced the size of the data the patches are flattened to return to a one dimensional representation. This representation is then used as an input for fully-connected linear layers performing the final reduction in scope for the latent space.

In between each layer there is also a dropout [9] and a batch normalization [10] layer to aid the stability during learning.

The structure of the encoder allows for a balanced decrease in dimensionality for each layer, reducing the influence of the initial encoding layer by creating a fairly high dimensional output and more smoothly decreasing each layers dimensionality. The attention portion allows the otherwise disconnected patches patches to find relevant information from other patches.

2.3 Decoder

The decoder follows a similar structure as the encoder with the latent space being input into fully-connected linear layer, followed by the same structure of patch-wise linear and multi-head attention as seen in the encoder. The main distinction for the decoder is the final layer is larger than the input layer to accommodate for the necessary parameters of the ZINB distribution.

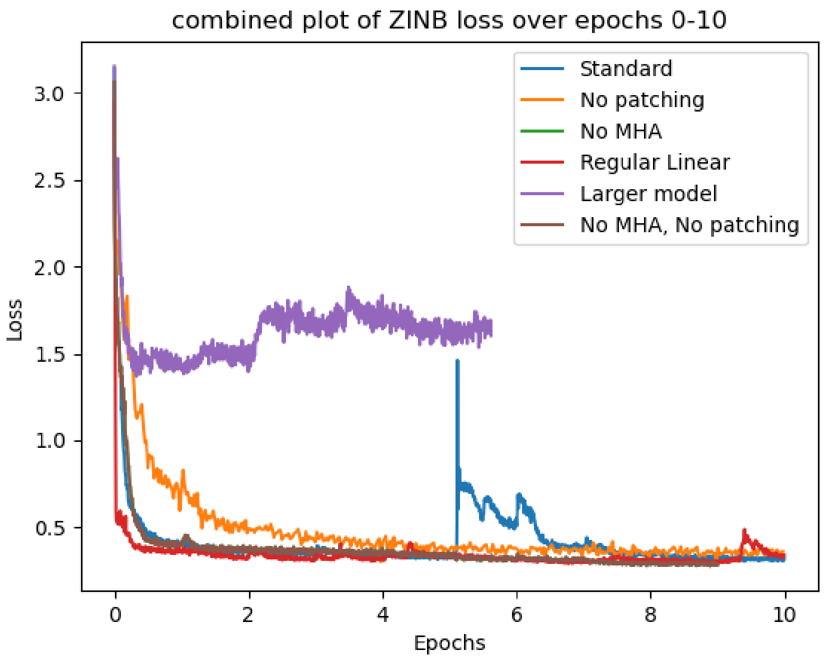

Figure 1: Batch loss over training with various model structures changed

3 Results

3.1 Model training

The model was trained in several experiments using subsets of the Human Brain Cell Atlas [16]. The complete dataset consists of 3 million cells with each cell being mapped for 59480 cells. The high dimensionality and large amount of available data as well as the varied tissues represented makes it a suitable dataset for training large neural network models.

The optimizer used was AdamW [12] with a cosine annealing [13] learning rate scheduler.

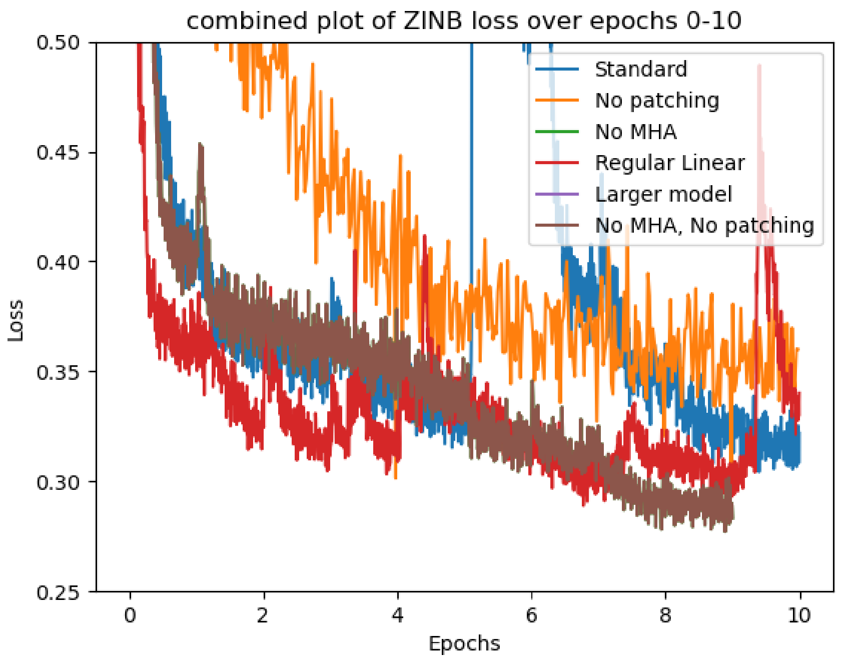

Figure 2: Zoomed in version of loss over training

The model does function and is able to learn relatively stably. There is no indication that the model is diverging. However the loss saturates quickly though it continues to improve slowly even over latter epochs. Verifying the loss values against a validation dataset shows that the model does not appear to overfit possibly pointing to the issue of the loss saturation rather being an underfitting problem. Underfitting occurs when a model [2] is unable to accurately capture the true correlations that are necessary in order to provide good predictions for the training dataset. Attempting to solve this issue by increasing the number of parameters in the model does not solve this problem as the loss values increase rather than decreasing following this change.

More worringly attempts to study the impact of the various components of the encoder and the patching layers shows that removing them has little impact on the overall training of the model. The loss curves for all six of the model types trained for achieve similiar loss values in a relatively similiar number of epochs. Removing the multi-head attention and the patching layers even seem to slightly improve the learning of the model. This implies that the overall structure of the network may be unsuitable for the task as it does not appear to be able to utilize these layers properly.

3.2 Downstream analysis and reference implementations

Table 1: Sample table title

	Part			
Model	No. of Training Cells	Training Time	Reconstruction Loss	Cell Type Accuracy
scVI	42,116	4 minutes	13,470.76	91.75%
scViT	42,116	1 hour	20,541.98	26.93%
scViT	168,464	4 hours	20,892.72	17.76%

To evaluate the performance of the model in comparison to other auto encoder based models, two comparison experiments were run. As a reference implementation scVI [6] was used. All models were trained over 10 epochs using subsets of the Human Brain Atlas dataset [16]. The first two models are both trained on the same number of cells to establish baseline between scVI and scViT under identical circumstances, while the third test examines the impact of increasing the size of the training data.

The results clearly show that while the scViT model is able to learn some information the loss is still significantly worse than the standard scVI implementation. Furthermore the scVI model trains far quicker scViT than the requiring only 4 minutes compared to one hour. Increasing the size of the training dataset does not improve performance for the scViT model.

For the downstream analysis comparison a simple multilayer perceptron was paired with downsampling using either scVI and scViT for classifying cell types. The two encoding networks used a latent space of size 64. This test aims to study how well the model is able to summarize the data into a latent space and how that latent space can be applied to other biologically relevant tasks. The results clearly show that scVI greatly outperforms the scViT network.

4 Conclusions

4.1 Conclusions drawn from results

In the current form scViT is able to learn and does technically function, however the performance compared to more conventional architecture such as scVI is poor. On the tasks surveyed as a part of this project scViT fails to match the performance of scVI while still being significantly more computationally expensive to run. A concern during training of the models was the potential that the high-sparsity of the dataset may lead to a model finding a local minima by only predicting a count of zeros for each gene irregardless of the input data. However observing the results shows that this does not appear to be what is actually occurring as the model is able to predict positive mean values. The usage of the ZINB distribution Another possible reason why this would appear is that the rate of zero-inflation is very high though this also doesn't appear to be the case as the rate is on average only about 10 percent.

The results for the losses during training appear to show that the architecture in its current state is flawed and does not perform as expected. The fact that components which theoretically should have an impact on the performance can be removed without significant impact on training implies that they do not perform as expected.

The problems with model structure may also account for the failure of the latent space to properly encode enough information to be usable for downstream analysis tasks. As these results are significantly worse than the scVI and even continue to get worse when trained over more data implies that the model might not even be learning biologically relevant details but rather optimizing it task through some form of copying.

Further studies would need to be performed in order to establish this definitely as computational limitations restricted the number of tests that could be run. Perhaps adapting explainable AI methods, such as DeepSHAP, could help clarify why the layers do not perform as hoped.

The high number of parameters and different layers used in the model further complicate the study of the model performance as there are many different possible components which could cause the issues that are occurring during the training process and it greatly increases the difficulty of the hyperparameter tuning. The fact that each layer has several different values which can be tuned makes it infeasible to perform exhaustive an hyperparameter search.

Overall the model is not practically usable and for future research a different direction should most likely be taken. Smaller incremental improvements to previously successful models would perhaps be a more productive approach and an important lesson which could be drawn from this project is that one should be cautious with adding additional components to neural networks.

References

[1] Dor Bank, Noam Koenigstein, and Raja Giryes. *Autoencoders*. 2021. arXiv: `2003.05991 [cs.LG]`. URL: `https://arxiv.org/abs/2003.05991`.

[2] Daniel Bashir et al. *An Information-Theoretic Perspective on Overfitting and Underfitting*. 2020. arXiv: `2010.06076 [cs.LG]`. URL: `https://arxiv.org/abs/2010.06076`.

[3] Rishi Bommasani et al. *On the Opportunities and Risks of Foundation Models*. 2022. arXiv: `2108.07258 [cs.LG]`. URL: `https://arxiv.org/abs/2108.07258`.

[4] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: `2010.11929 [cs.CV]`. URL: `https://arxiv.org/abs/2010.11929`.

[5] Gökcen Eraslan et al. "Single-cell RNA-seq denoising using a deep count autoencoder". In: *Nature Communications* 10 (2019), p. 390. DOI: `10.1038/s41467-018-07931-2`. URL: `https://doi.org/10.1038/s41467-018-07931-2`.

[6] Adam Gayoso et al. "scvi-tools: a library for deep probabilistic analysis of single-cell omics data". In: *bioRxiv* (2021). DOI: `10.1101/2021.04.28.441833`. URL: `https://doi.org/10.1101/2021.04.28.441833`.

[7] Mengting Hao et al. "Large-scale foundation model on single-cell transcriptomics". In: *Nature Methods* 21 (2024), pp. 1481–1491. DOI: `10.1038/s41592-024-02305-7`. URL: `https://doi.org/10.1038/s41592-024-02305-7`.

[8] Asif Haque et al. "A practical guide to single-cell RNA-sequencing for biomedical research and clinical applications". In: *Genome Medicine* 9 (2017), p. 75. DOI: `10.1186/s13073-017-0467-4`. URL: `https://doi.org/10.1186/s13073-017-0467-4`.

[9] Geoffrey E. Hinton et al. *Improving neural networks by preventing co-adaptation of feature detectors*. 2012. arXiv: `1207.0580 [cs.NE]`. URL: `https://arxiv.org/abs/1207.0580`.

[10] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. arXiv: `1502.03167 [cs.LG]`. URL: `https://arxiv.org/abs/1502.03167`.

[11] Günter Klambauer et al. *Self-Normalizing Neural Networks*. 2017. arXiv: `1706.02515 [cs.LG]`. URL: `https://arxiv.org/abs/1706.02515`.

[12] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: `1711.05101 [cs.LG]`. URL: `https://arxiv.org/abs/1711.05101`.

[13] Ilya Loshchilov and Frank Hutter. *SGDR: Stochastic Gradient Descent with Warm Restarts*. 2017. arXiv: `1608.03983 [cs.LG]`. URL: `https://arxiv.org/abs/1608.03983`.

[14] Andrzej Maćkiewicz and Waldemar Ratajczak. "Principal components analysis (PCA)". In: *Computers & Geosciences* 19.3 (1993), pp. 303–342. ISSN: 0098-3004. DOI: `10.1016/0098-3004(93)90090-R`. URL: `https://www.sciencedirect.com/science/article/pii/009830049390090R`.

[15] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

[16] Kimberly Siletti and et al. "Transcriptomic diversity of cell types across the adult human brain". In: *Science* 382 (2023), eadd7046. DOI: `10.1126/science.add7046`. URL: `https://doi.org/10.1126/science.add7046`.

[17] Keisuke Tsuyuzaki et al. "Benchmarking principal component analysis for large-scale single-cell RNA-sequencing". In: *Genome Biology* 21 (2020), p. 9. DOI: `10.1186/s13059-019-1900-3`. URL: `https://doi.org/10.1186/s13059-019-1900-3`.

[18] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: `1706.03762 [cs.CL]`. URL: `https://arxiv.org/abs/1706.03762`.

[19] Isaac Virshup et al. "anndata: Annotated data". In: *Journal of Open Source Software* (Sept. 2024). DOI: `10.21105/joss.04371`. URL: `https://doi.org/10.21105/joss.04371`.

[20] Fabian A. Wolf, Philipp Angerer, and Fabian J. Theis. "SCANPY: large-scale single-cell gene expression data analysis". In: *Genome Biology* 19 (2018), p. 15. DOI: `10.1186/s13059-017-1382-0`. URL: `https://doi.org/10.1186/s13059-017-1382-0`.

[21] Tete Xiao et al. *Early Convolutions Help Transformers See Better*. 2021. arXiv: `2106.14881 [cs.CV]`. URL: `https://arxiv.org/abs/2106.14881`.

[22] Fan Yang et al. "scBERT as a large-scale pretrained deep language model for cell type annotation of single-cell RNA-seq data". In: *Nature Machine Intelligence* 4 (2022), pp. 852–866. DOI: `10.1038/s42256-022-00534-z`. URL: `https://doi.org/10.1038/s42256-022-00534-z`.