GENERATIVE MODELLING FOR PHENOTYPIC PROFILING

Andrea Valletta (S180768), Julia Cathrine Youngman (S123540) and Simon Daniel Eiriksson (S180722)

https://github.com/simoneiriksson/DeepLearning_02456_final

ABSTRACT

This article seeks to investigate different generative models for phenotype profiling of cells. The models suggested learn representations of single cell images in the context of phenotype profiling which is an open scientific problem with applications in disease diagnosis and drug discovery. Variational Autoencoder (VAEs), Sparse Variational Autoencoders (SparseVAEs) and Generative Adversarial Networks (GANs) were the models suggested in this work. All suggested models were developed from scratch with inspiration from previous state of the art generative models. Among the suggested models, Sparse VAEs and GAN model types had the highest classification accuracies of 90.4% compared to standard VAE models having classification accuracies of 83.7-88.5%.

1. INTRODUCTION

Phenotype profiling is the task of identifying and categorizing phenotypes of cells based on their physical characteristics. This is a fundamental problem in biological and medical sciences and has applications within disease diagnosis and drug discovery. Phenotype classification can provide insights into the underlying genetic and environmental factors that contribute to how a cell develop into a particular phenotype. For example, many diseases are associated with specific phenotype abnormalities. Identifying the presence of certain physical features in a patient's cells can potentially assist in diagnosis and for predicting their risk of developing a given disease.

In the work presented in this paper, a pre-processed version of the BBBC dataset is used for phenotype classification [1] . The dataset consists of 480,000 fluorescent microscopy images of single breast cancer cells. Phenotype classification is obtained by firstly constructing a sparse Variational Autoencoder (sparse VAE) and a Generative Adversarial Network (GAN) model and afterwards a K nearest neighbor classifier.

2. BACKGROUND

VAEs and GANs are generative models consisting of two neural networks each. VAEs can be divided into an encoder neural network (NN) and a decoder NN, and are often used for semi-supervised learning and classification tasks. GANs consist of a generator and a discriminator. GANs are often used for tasks such as image generation and data augmentation. Following Lafarge et al. [2] we want to investigate if adding a GAN image discriminator improves the classification performance of our VAE model.

2.1. Autoencoders

An autoencoder is a type of neural network that is trained to map an input from a larger input space, into a smaller latent space, and back into the original input space. From a technical point of view, the autoencoder code can be described as splitting the network into two parts: an encoder function $\mathbf{z} = f(\mathbf{x})$ that takes the input and shrinks it to the latent space $\mathbf{z}$, and a decoder function that uses the learned latent space to produce an output, $\mathbf{r} = g(\mathbf{z})$. For constraint, the output should be as close as possible to the input, which is why it is called "input reconstruction".

2.2. Variational autoencoders

VAEs use learned approximate inference by imposing a constraint on the distribution (the prior) of the latent space variables $\mathbf{z}$. The VAE generates an output sample from the model by drawing a sample $\mathbf{z}$ from the learned latent distribution $p_{model}(\mathbf{z})$, which is then passed through the decoding network, $g(\mathbf{z})$ so that finally $\mathbf{x}$ is sampled from the distribution $p_{model}(\mathbf{x}|\mathbf{z})$. The training of VAEs is usually evaluated by three metrics: The model specific Mean Square Error loss, the Kullback-Leibler (KL) divergence and the evidence lower bound (ELBO). The KL-divergence measures the difference between the approximate posterior $q(\mathbf{z}|\mathbf{x})$ and the prior distribution $p(\mathbf{z})$. The ELBO is a scalar value measuring the fit of the model to the data. If the chosen loss function is the log-likelihood of the observed input, the ELBO function can be written as:

$$\mathcal{L}(\mathbf{x}) = E\left[\log\left(p\left(\mathbf{x}|\mathbf{z}\right)\right)\right] - \mathcal{D}_{KL}\left(p\left(\mathbf{z}|\mathbf{x}\right)|p\left(\mathbf{z}\right)\right) \quad (1)$$

Optimizing the ELBO is a trade-off between two terms: the first term measures the quality of the reconstruction while the second term enforces the approximate posterior $q(\mathbf{z}|\mathbf{x})$ to

Supervised by Jes Frellsen.

match the prior distribution $p(\mathbf{z})$. The KL-divergence is calculated as

$$\mathcal{D}_{KL}\left(p\left(\mathbf{z}|\mathbf{x}\right)|p\left(\mathbf{z}\right)\right) = \log q_\phi(\mathbf{z}|\mathbf{x}) + \log p_\theta(\mathbf{z}) \quad (2)$$

where $p_\theta(\mathbf{z})$ is the prior distribution for the latent variables, and $q_\phi(\mathbf{z}|\mathbf{x})$ denotes the posterior of the latent representation, given the observation $\mathbf{x}$. Most commonly, the prior distribution of the latent variables is Gaussian where the posterior takes the form

$$q_\phi(\mathbf{z}|\mathbf{x}) := \mathcal{N}(\mathbf{z} \mid \mu_\phi(\mathbf{x}), \sigma_\phi(\mathbf{x})),$$

where $\mu_\phi(\mathbf{x})$ and $\sigma_\phi(\mathbf{x})$ are outputs from the neural network.

Using the log-probabilities is however only possible if it is possible to write out the probability density function for the distributions in closed form which is not always possible. Therefore, it is chosen to implement the VAE using a closed form expression for the KL-divergence which shares the minimum values. In the standard case the KL-divergence can according to Kingma et al. [3] be rewritten as:

$$\mathcal{D}_{KL}\left(p\left(\mathbf{z}|\mathbf{x}\right)|p\left(\mathbf{z}\right)\right) = \frac{1}{2}\sum_{j=1}^{J} 1+\log(\sigma_j^2(x_i))-\mu_j^2(x_i)-\sigma_j^2(x_i) \quad (3)$$

Similarly, the log-likelihood term $\log(p(\mathbf{x}|\mathbf{z}))$ has the same minima as the MSE-loss meaning that in practice the MSE-loss can be used instead.

2.3. Variational Sparse coding

Autoencoders produce latent representations, without any restrictions to their distribution. The Gaussian VAE forces a normal prior distribution unto the latent representations making the distribution of the latent representation fill out the latent space approximately normally distributed. However, this distribution of latent representations are disperse in the latent space which is not necessarily ideal for classification tasks and human interpretability. Sparse VAEs address this problem by merging ideas from VAEs and sparse coding. A sparse VAE with modelled sparsity in the latent space is introduced by using a Spike and Slab prior distribution rather than the Gaussian distribution. The Spike and Slab prior distribution is defined by two nested random variables; a selector variable drawn from a Bernoulli-distribution with propability α, and a continuous variable that is Gaussian distributed. The selector variable then chooses if the output should be zero or taken from the Gaussian distribution. This means that with probability α, the value for the given dimension will be normally distributed, and with probability $(1-\alpha)$ the value will be zero. This causes the network output to only occupy a smaller subset of the dimensions of the latent space.

Thus, the prior distribution takes the form:

$$p(\mathbf{z}) = \prod_{j=1}^{J} \alpha\mathcal{N}(z_j; 0, 1) + (1-\alpha)\delta(z_j) \quad (4)$$

while the posterior distribution is

$$q_\phi(\mathbf{z}|x_i) = \prod_{j=1}^{J} \gamma_{i,j}\mathcal{N}(z_j(x_i); \mu_j(x_i), \sigma_j^2(x_i)) \quad (5)$$

$$+ (1-\gamma_j(x_i))\delta(z_j(x_i)) \quad (6)$$

The first term is the slab part and the second the spike [4]. Here, the $\mu(x_i), \sigma(x_i)$ and $\gamma(x_i)$ variables are output form the neural network.

The resulting KL-divergence consists of two terms now: the first is the negative KL-divergence of the Slab variables multiplied by the probability of $z_j(x_i)$ being non-zero; the second term is the negative KL divergence between the distributions of the Spike variables.

$$\begin{aligned} &D_{KL}\left(q_\phi\left(z|x_i\right)||p\left(z\right)\right) = \\ &-\sum_{j}^{J}\Bigg[\gamma_j(x_i)\frac{1}{2}\left(1+\log\sigma_j^2(x_i)-\mu_j^2(x_i)-\sigma_j^2(x_i)\right) \\ &-(1-\gamma_j(x_i))\log\left(\frac{1-\alpha}{1-\gamma_j(x_i)}\right)-\gamma_j(x_i)\log\left(\frac{\alpha}{\gamma_j(x_i)}\right)\Bigg] \end{aligned}$$

The Spike variables takes values of either 1 or 0 with defined possibilities of α and $(1-\alpha)$, respectively. $\mu_j(x_i)$, $\sigma_j(x_i)$ and $\gamma_j(x_i)$ are outputs of a neural network with input x_i.

2.4. Generative Adversarial Networks

The final class of deep learning model used in this work are Generative Adversarial Networks (GANs). Similarly to VAEs, GANs consists of two neural networks: a generator network designed to generate new, synthetic data similar to a given training dataset and a discriminator network trained to distinguish between synthetics data and the real training data. The generator and discriminator networks are trained simultaneously, with the generator trying to produce synthetic data that the discriminator cannot distinguish from real data, and the discriminator trying to correctly classify the synthetic data as fake. Training of a GAN ultimately seeks to find a balance between the generators ability to generate trustworthy synthetic data and the discriminators ability to correctly distinguish between synthetic and real data. [5] p. 690 and [6]

In the setup of this work, the VAE takes the role of the generator, where the GAN discriminator is trained to decide if the reconstructed cell image is a true or a fake image cf.

3. MATERIAL AND METHODS

This section seeks to describe how we have combined the methods in the neural network architectures for the generative models described in the previous section in the construction of four different new architectures. All code is built from

scratch, using various articles and repositories as inspiration. The sources are referred when relevant.

3.1. CytoVAE

The first model is an implementation of the CytoVAE as described by Lafarge et al.[2] This model is a simple VAE built on two convolutional neural networks. It is attempted to implement the exact same architecture as [2, 7] where the encoder consists of 4 layers and each layer has a convolutional layer, followed by downsampling with MaxPool2, then a LeakyReLU activation function and lastly batch normalization. The output of the encoder is a single pixel "image" with 256 channels for both μ and σ. From this the final latent representation is drawn from a $\mathcal{N}(\mu, \sigma^2)$ distribution. The decoder is similarly constructed with four layers consisting of a convolutional layer, batchnormalization, activation function and than finally a layer that upsamples the image size. The loss function that the network seeks to minimize is given in equation 1. The process of sampling from a distribution parameterised by a model is not differentiable. This is solved by applying the reparametrization trick treating a random sample as a noise term. In case of a Gaussian prior distribution, the noise is treated like a standard normal distribution, making the noise term independent from the model. The reparametrization thereby allows a gradient path through a non-stochastic node enabling backpropagation.

The reparametrization trick is adopted from [3], using the following pseudocode:

$\varepsilon \sim \mathcal{N}(0, I)$
$z = \mu + \sigma \cdot \varepsilon$

This allows the autograd compute graph to differentiate through the added noise.

3.2. SparseVAE

The second model introduced is the SparseVAE model based on the CytoVAE model but rather than a two-dimensional output of the encoder the SparseVAE model produces a three dimensional output: γ, μ and σ functioning as the parameters for the distribution of the latent representation as described in (5). In this case the reparamerization trick is slightly more complicated. The method is adopted from [8] by using the following pseudocode:

$\epsilon = \mathcal{N}(0, I)$
$\eta = \mathcal{N}(0, I)$
$selector = \text{sigmoid}(\gamma + \eta - 1)$
$z = selector \cdot (\mu + \epsilon \cdot \sigma)$

Where the $selector$ is ~ 1 if the variable is a slab and ~ 0 if the variable is a spike. Sparsity forces the model to use only a subset of the total dimensions of the latent space when representing each image. The choice of dimensions to use is determined on an image-to-image basis and should in theory lead to similar images being represented in separate subsets of the dimensions of the latent space. This should result in more distinguishable image clusters since the clusters populate different dimensions of the latent space. On the other hand it would be expected that the sparsity of the latent space will result in higher reconstruction errors. The loss function in this case is the same as for the CytoVAE, however, the KL-divergence is different.

3.3. CytoVAEGAN

The third model introduced is inspired by the CytoVAE+ model described by [2]. In this paper we have named our own implementation of this model CytoVAEGAN. This model adds a discriminator which aims to decide whether an image is a reconstruction from the VAE or is an original cell image. To do so the discriminator has an architecture similar to the encoder in the CytoVAE except that after the final layer there is a dense layer with a one-dimensional output followed by a sigmoid activation function. While the standard GAN setup has a generator which is trained to minimize the discriminators ability to distinguish between real and fake data, the CytoVAE+ takes a different approach. Here the VAE is trained partly on the reconstruction loss as in the CytoVAE but it is also trained to minimize the euclidean distance between the representations of the original and the reconstructed image inside of the discriminator. If the euclidean distance between the discriminators inner representations of the fake and the real image in layer n of the discriminator is denoted $\mathcal{L}_n^D$, then the VAE is trained to minimize the sum loss $\mathcal{L}^{totalvae} = \mathcal{L}^{vae} + \mathcal{L}_1^D + \mathcal{L}_2^D + \mathcal{L}_3^D$ as seen in fig. 1. The discriminator is trained to minimize the cross-entropy between its guess and the true case of each image. Attention should be given to the fact that the autograd compute tree of the two loss functions are entangled. This can be resolved by first computing the $\mathcal{L}^{totalvae}$ for a batch using both the VAE and the discriminator and then optimizing the VAE. Hereafter, the images used in the batch are explicitly detached, the discriminator loss is recalculated and the discriminator is optimized.

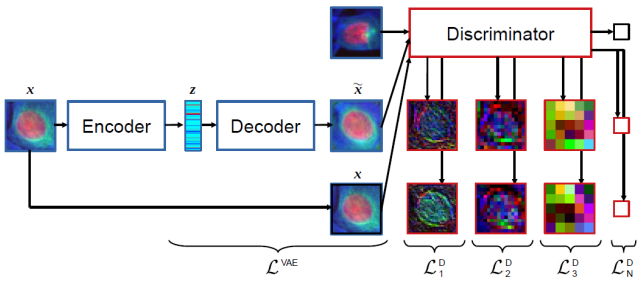

Fig. 1: Sketch of VAE/GAN from [2]

For the CytoVAEGAN model the loss between the inner representations in the discriminator $\mathcal{L}_n^D$ will force the VAE

network to pay more attention to the texture of the reconstructed images than the CytoVAE model. This should lead to image reconstruction with improved texture quality.

3.4. SparseVAEGAN

The SparseVAEGAN model is a combination of the SparseVAE model and the CytoVAEGAN, where the SparseVAE is used as the image reconstructor and the discriminator from the CytoVAEGAN.

3.5. The BBBC021 dataset

The generative models were trained and evaluated using a preprocessed version of the BBBC021 (Broad Biobank Benchmark Collection) dataset. The dataset consists of 480,000 fluorescent microscopy images of single breast cancer cells. The data originates from a high-throughput experiment performed in multi-well plates. Each plate has 96 wells and in each well a sample of cells has been treated with a compound at a specific concentration. There is a total of 104 different treatment compounds and 8 different compound concentrations. After treatment the cells were stained using fluorescent markers for DNA, F-actin and B-tubulin and visualised by fluorescent microscopy. The single-cell images are treated as RGB images mapping DNA→R, F-actin→G, b-Tubulin→B. The metadata about the images contains information about the well, compound, compound concentration and compound mode of action (moa) of every cell. In total there is 13 different moas, meaning that multiple treatment compounds can have the same moa [1].

4. EXPERIMENTS

Four types of machine learning models (CytoVAE, CytoVAEGAN, SparseVAE, SparseVAEGAN) were trained with varying hyperparameters, as seen in table 1. Accuracies of each model is listed along with the results using similar models produced by [2]. The models built for this work have classification accuracies ranging from 86.7% to 90.4% while [2] report 90.6% as their highest accuracy. A few models were trained increasing the number of epochs from 50 to 100 with no significant changes in accuracy. Models implemented here were built using the Pytorch framework rather than Tensorflow like in the original work by [2].

The results in this work compares well to the classification accuracies reported by [2]: our CytoVAE with $\beta = 0$ (acc. 88.5%), compares well with their similar model CytoAE (acc. 87.6%), while our CytoVAE with $\beta = 1$ (acc. 83.7%) compares well with their model (CytoVAE with acc. 83.5%) and lastly our CytoVAEGAN with $\beta = 0$ (acc. 90.4%) compares well with their CytoVAE+ (acc. 90.6%) model. In addition to the implementation of the original models in [2], sparse VAEs were built and implemented in this work. The sparse VAE models were trained using $\alpha \in \{0.05, 0.1, 0.2\}$, with the highest accuracy received using $\alpha = 0.1$. For the Cyto models, models with $\beta = 0$ have a higher performance than $\beta = 1$.

In the interest of avoiding redundant information, the remaining of this section will show results only for the SparseVAE model with $\alpha = 0.05$ and $\beta = 1$ trained with 50 epochs.

Table 1: Summary of the Generative models developed and trained for this work with varying α, β, Epochs along with the resulting Mean Square Error loss and accuracy. Results are compared with [2].

Model	α	β	Epochs	MSE	Acc. [%]
CytoVAE	-	0	50	30.1	88.5
CytoVAE	-	0	100	29.4	88.5
CytoVAE	-	1	50	93.7	83.7
CytoVAEGAN	-	0	50	83.1	90.4
CytoVAEGAN	-	1	50	80.1	82.7
SparseVAE	0.05	1	50	59.7	89.4
SparseVAE	0.1	1	50	62.2	90.4
SparseVAE	0.2	1	50	72.6	88.5
SparseVAEGAN	0.05	1	50	74.8	86.5
SparseVAEGAN	0.1	1	50	66.1	88.5
SparseVAEGAN	0.2	1	50	71.5	87.5
Lafarge (2019)	α	β	Epochs	MSE	Acc. [%]
CytoAE	-	0	-	-	87.6
CytoVAE	-	1	-	-	83.5
CytoVAE+	-	1	-	-	90.6

4.1. Learning Graphs and Reconstructions

The ELBO is used for training the SparseVAE. It combines the Mean Squared Error (MSE) loss between the reconstructions and the original input and the KL divergence between the approximate posterior and the prior distribution. The MSE loss encourages the SparseVAE model to reconstruct the input accurately. The KL divergence encourages the approximate posterior to be close to the prior. By maximizing the ELBO, the SparseVAE is able to learn to reconstruct the input accurately while also learning a compact latent representation of the data. As can be seen from figure 4, the image reconstruction improves when extending with the GAN network (second from above), as it emphasizes texture in the reconstructed image, compared to the CytoVAE (first row).

4.2. Interpolating between Phenotypes

Given two different images of cells, a target and control cell, interpolations between the two images were made by creating evenly distributed samples interpolating between the la-

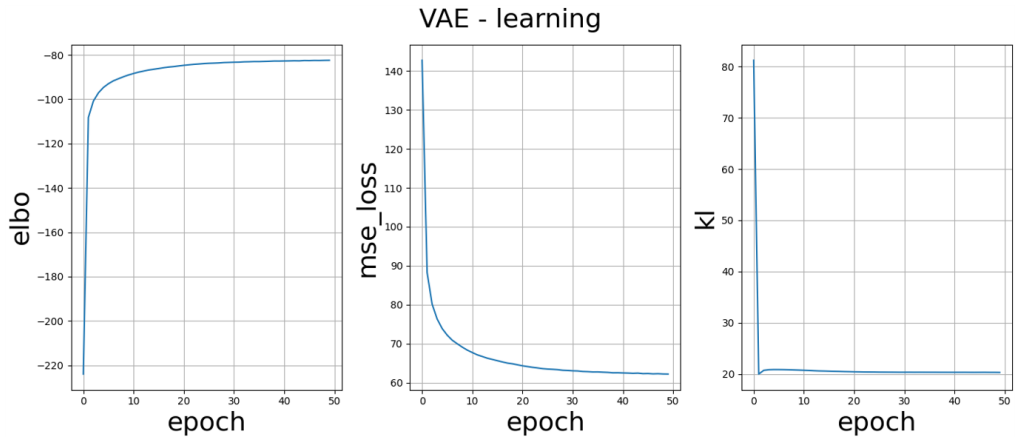

Fig. 2: Learning graphs for the SparseVAE model.

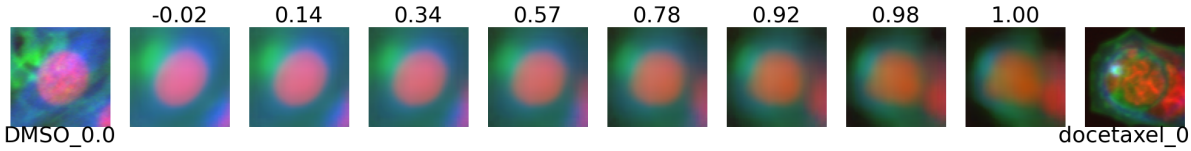

Fig. 3: Interpolation from control (left: DMSO_0.0) to target cell (right: docetaxel˙0.3). Interpolations were created using latent space reconstructions from the SparseVAE model. The leftmost and rightmost images are original images while the images in between are reconstructions.

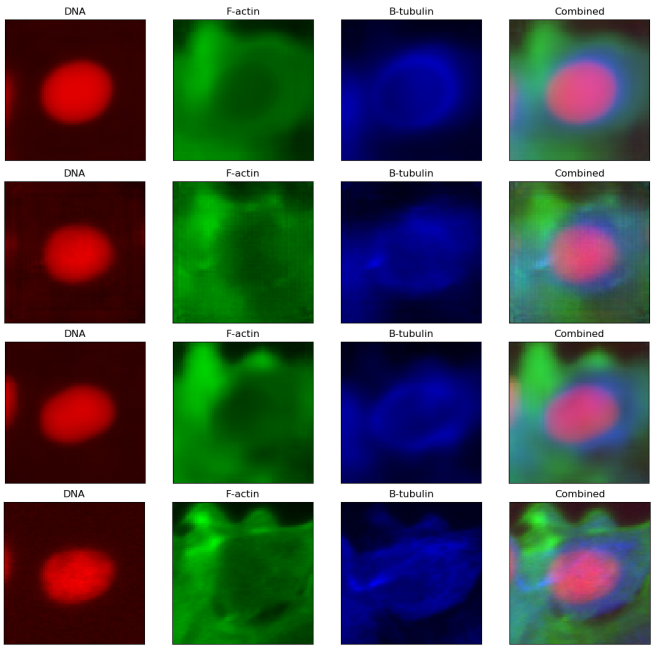

Fig. 4: Top row: latent space reconstructions RGB channels individually and combined of a cell treated with docetael of concentration 0.3 a.u. using the CytoVAE, CytoVAEGAN, and the SparseVAE. Bottom row: original images of the same cell. RGB images mapping: DNA→Red, F-actin→Green, b-Tubulin→Blue.

tent representations between the control and the target cell. The samples were then used for image reconstruction. Interpolating between images of different cells under different treatment conditions can potentially provide interesting insights about how a compound affects cellular structure and provide a visual overview of how a cell can evolve into one particular phenotype. An example of interpolation between two cells is shown fig. 3. The image reconstruction from interpolated values of the latent variables is possible due to the constraints on the prior distribution $p(\mathbf{z})$ enforced by the ELBO loss function. These constraints guarantee that the latent space becomes more compact than with a standard AE.

4.3. Feature disentanglement

The SparseVAE models' ability to disentangle the latent space variable is examined by creating a heatmap showing the absolute correlation between the moas and the latent variables as seen in fig. 5. The color indicates the magnitude of the correlations, which are generally below 0.1, with a few latent variables taking values up to 0.25. It is seen that the moas 'DMSO' and 'Microtubule Stabilisers" generally have latent variables with larger magnitude and have a higher number of significant non-zero values. This can likely be explained by the aforementioned moas are overrepresented in the dataset compared to other moas. This hypothesis can potentially be investigated by training a model using a balanced dataset which is beyond the scope of this work.

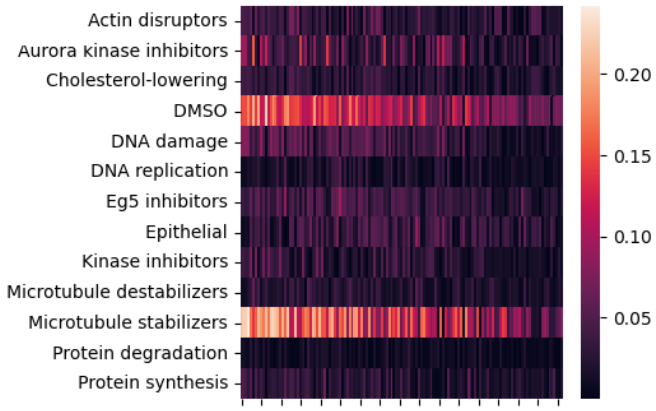

Fig. 5: Heatmaps of sorted latent variables showing the absolute correlation between moa classes vs. latent variables. The latent variables in the x-axis are sorted by decreasing correlation magnitude. The color indicates the magnitude of each latent variable per moa. Only the 136 (out of 256) most information dense latent variables variables are shown.

4.4. Classification

The overarching challenge of the BBBC021 data in the context of generative modelling is to predict the mode-of-action (moa) serving as the label of the compound. A profile of each well was computed as the average of all cells in that particular well. Hereafter, the profile for each unique treatment (unique combination of compound and concentration) was created by computing the median of all wells with that particular treatment, as explained in [2]. Once the treatment profiles were created, a classifier using a KNN=1 nearest neighbor approach was built for evaluating model performance. This was done using the Not-Same-Compound approach suggested by [2] and [9] where all profiles of the same compound (regardless of concentration) were held out to avoid overfitting.

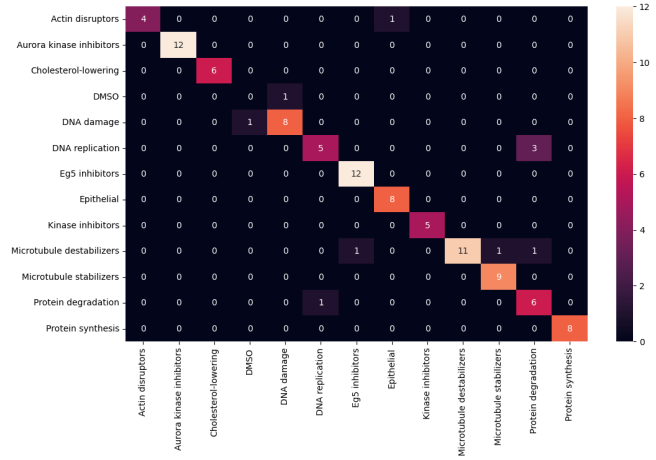

Fig. 6: SparseVAE confusion matrix

The confusion matrix sums to 104 representing the 104 different treatment compounds. Hereby, the classifier attempts to predict what compound belongs to what moa. The confusion matrix forms a clear diagonal with minor misclassifications with DNA replication/Protein degradation as the most significant confusion.

5. CONCLUSION

In this work new, generative models were built using Sparse VAE and GAN model types with the aim of phenotypic profiling of single cell images. The latent space representations of the images were investigated and visualised by interpolating between cell phenotypes in latent space. The importance of the latent variables were examined by creating a heatmap plotting the absolute correlation between the moas vs. the latent variables sorted by order of importance. The moas 'Microtubule Stabilizers' and 'DMSO' had the most significant latent variables, which might be explained by class imbalance. The classifier produced a confusion matrix with good results. Our best performing model was the SparseVAE reached an accuracy of 90.4% only slightly below the accuracy reported by Lafarge et. al. [2] on 90.6%. Models implementing either Sparse VAEs or GANs improved the results and reached classification accuracies of 90.4% compared to VAEs with accuracies of 83.7% to 88.5%. Adding a GAN archietecture to the sparse VAE (SparseVAEGAN) did not improve the results. By training new models doubling the number of epochs from 50 to 100, it is concluded that no significant improvement of model performance is achieved by increasing the training time. In order to construct even better models than suggested in this work, the model architecture would be subject to further investigations.

References

[1] BROAD Institute, "Broad bioimage benchmark collection," https://bbbc.broadinstitute.org/BBBC021, 2012.

[2] C.D. Larfarge, A.B. Smith, and E.F. Roberts, "Capturing single-cell phenotypic variation via unsupervised representation learning," *The Journal of Machine Learning Research (JMLR)*, vol. 102, pp. 315–325, 2019.

[3] Diederik P. Kingma, Tim Salimans, and Max Welling, "Variational dropout and the local reparameterization trick," *Advances in Neural Information Processing Systems 28 (NIPS 2015)*, 2015.

[4] Francesco Tonolini, Bjørn Sand Jensen, and Roderick Murray-Smith, "Variational sparse coding," *"The Journal of Machine Learning Research (JMLR)"*, vol. 115, pp. 690–700, 2020.

[5] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, Cambridge, MA, USA, 2016, http://www.deeplearningbook.org.

[6] Peter Goldsborough, Nick Pawlowski, Juan C Caicedo, Shantanu Singh, and Anne Carpenter, "Cytogan: Generative modeling of cell images," *Workshop On Machine Learning In Computational Biology, Neural Information Processing Systems*, 2017.

[7] C.D. Larfarge, "cytovae - extended variational autoencoder for single-cell representation learning," https://github.com/tueimage/cytoVAE, 2019.

[8] Robert de la Fuente, Alfredo; Aduviri, "Variational-sparse-coding," https://github.com/Alfo5123/Variational-Sparse-Coding, 2019.

[9] Michael D. Ando, Cory McLean, and Marc Bernd, "Improving phenotypic measurements in highcontent imaging screens," 2017.