AIML 425 T2/2022 - ASSIGNMENT 2 - PROBLEM 2

Corvin Idler - ID 300598312

1. INTRODUCTION

This report describes experiments of using a Variational Autoencoder to convert a 2D uniform distribution into a bivariate Gaussian in latent space and back into a 2D uniform distribution.

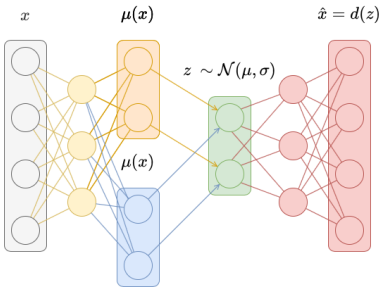
2. THEORY / CONCEPT

2.1. General concept

The task and network configuration set out in the assignment is what is usually described as a Variational Autoencoder (VAE). VAEs are neural networks used to perform (non-linear) dimensionality reduction or manifold learning. They consist of a decoder d and an encoder e. The encoder performs a non-linear transformation $e\colon X \mapsto Z$ to project data into a (often lower-dimensional) latent space. It is desirable that this transformation has the property that similar values in the original space lead to similar values in the latent space. The decoder $d\colon Z \mapsto X$ maps latent variables back to the original input space. An autoencoder is the composition of the encoder and the decoder with the objective of minimizing the reconstruction error. As the assignment called for the latent variables to follow a certain distribution (Gaussian) we need what is called a Variational Autoencoder (VAE) that tries to enforce certain distributional properties on the latent variables on top of the objective of keeping the reconstruction error low. One usually enforces a certain distribution on the latent space and variables to make sure that the learnt manifold representation is continuous over the latent space (and not a collection of disjoint sub-spaces). To achieve this objective on doesn't just map the input x to a latent vector z, but instead to a mean vector $\mu(x)$ and a standard deviation vector $\sigma(x)$ which define a Gaussian distribution $\mathcal{N}(\mu, \sigma)$ from which a latent vector z can then be sampled. One usually introduces a penalty term in the objective function to make sure the distribution $\mathcal{N}(\mu, \sigma)$ doesn't deviate too much from the standard normal distribution $\mathcal{N}(0, 1)$.

2.2. Mathematical problem statement

In mathematical terms the above can be formulated as follows: One assumes that there exists a hidden random variable z that generates x. We can only observe x but would like to know more about z and how x and z are related. The resulting



(a) Problem

Fig. 1: Schematics of the VAE working principle with 2 latent variables. Source: https://avandekleut.github.io/vae

statistical inference problem is to estimate a (posterior) density $p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{z},\mathbf{x})}{p(\mathbf{x})} = \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{p(\mathbf{x})}$ of some latent variables of interest $\mathbf{z}$ given some input data $\mathbf{x}$ [1, 2]. As calculating $p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})dz$ is hard in many cases, it can be costly or impossible to solve this equation computationally with traditional Markov-Chain Monte Carlo (MCMC) methods. Variational inference therefore aims instead at approximating the posterior via an optimization problem by searching over a family $\mathcal{Q}$ of densities over the latent variables [1]. The goal is to find the candidate approximation $q(\mathbf{z}) \in \mathcal{Q}$ that is the closest (in terms of Kullback-Leibler divergence) to the exact posterior [1]. One calls $q(\mathbf{z})$ the variational density as it is the one thing over which we exercise control and which we will vary over the course of the optimisation as we explore the search space given by $\mathcal{Q}$. The resulting optimization problem can be formulated as follows [1]: $q^*(\mathbf{z}) = \underset{q(\mathbf{z})\in\mathcal{Q}}{\operatorname{argmin}} \mathrm{KL}\left(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\right)$. Unfortunately the exact KL divergence can't be calculated either as becomes appareant when written out in its long form: $\mathrm{KL}\left(q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})\right) = \mathbb{E}[\log q(\mathbf{z})] - \mathbb{E}[\log p(\mathbf{z},\mathbf{x})] + \log p(\mathbf{x})$. The equation contains once again $p(\mathbf{x})$ which is (as we know) often hard to calculate [1]. Instead of trying to minimize the exact KL divergence, one can try to maximize what is known as the evidence lower bound (ELBO) [1]: $\mathrm{ELBO}(\mathbf{q}) = \mathbb{E}[\log p(\mathbf{z},\mathbf{x})] - \mathbb{E}[\log q(\mathbf{z})]$. The ELBO is the negative KL divergence plus $\log p(\mathbf{x})$ (a constant). If one rewrites the $\mathrm{ELBO}(\mathbf{q}) = \mathbb{E}[\log p(\mathbf{z})] + \mathbb{E}[\log p(\mathbf{x}|\mathbf{z})] - \mathbb{E}[\log q(\mathbf{z})] = \mathbb{E}[\log p(\mathbf{x}|\mathbf{z})] - \mathrm{KL}\left(q(\mathbf{z})||p(\mathbf{z})\right)$, it becomes apparent that the objective function is a balance between likelihood (1st term)

and the divergence between the variational density and a prior density we need to define over the latent variables. Much like in [3] I use a hyper-parameter β to control the influence of the KL term in comparison to the likelihood function. In the original paper of [3] that was done to overemphasize the Kl term compared to the plain vanilla version of the VAE. The reasoning was to force a disentanglement of latent factors beyond what a normal VAE would have delivered. In very complex reconstruction tasks (like images) the likelihood term might dominate the KL term. In our scenario the opposite is true. The KL term dominates the likelihood term which leads to bad reconstruction results. So I aim to adjust the weight of the KL term downwards with a $\beta < 1$. Instead of setting β to a fixed value, [4, 5] propose an annealing approach for they weighting factor with initially zero weight on the KL factor to allow the network to first focus on good reconstruction and then quickly increasing β to one and finish the optimisation. Early experiments with this approach didn't lead to satisfactory results, so I stuck with a fixed β. The final objective function becomes $\mathbb{E}[\log p(\mathbf{x}|\mathbf{z})] - \beta * \mathrm{KL}\left(q(\mathbf{z})||p(\mathbf{z})\right)$. A second hyper-parameter λ was introduced to control the influence of a L2 weight regularisation.

3. RESULTS OF EXPERIMENTS / CONCLUSION

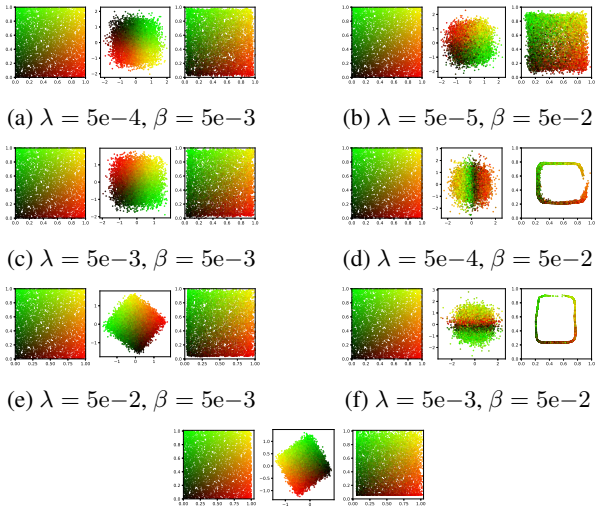
The jupyter-notebook underpinning this report can be found on GitHub: `https://colab.research.google.com/github/econdatatech/AIML425/blob/main/AIML425_Assignement_2.ipynb`.

3.1. Implementation details

All encoder and decoder consist of two hidden layers with 30 nodes each and Relu activation functions. Two different autoencoder were specified, one with two and the other with one latent variable. Two hyper-parameter exist, namely β to control the influence of the KL distance term and λ to control the influence of the L2 weight penalty term.

3.2. Results and discussion

In the visualisations in figure 2 we can see in the left column that increasing the L2 weight regularisation affects the distribution of the latent variables (a distribution that is half way between a uniform and a Gaussian is turned into a "diamond-shaped" one (subfigures a),c),e)g)). In subfigure g) we can also see, that the re-contruction is affected as from $\lambda = 0.25$ as the uniform distribution doesn't contain any values close to zero anymore. The colouring of the different quadrants of the uniform input distribution shows that the points stay grouped together in the latent distribution in some notion of a quadrant, but that a rotation or reflection might take place. The important feature is that points that are close to each other in the source distribution stay close together in the latent distribution. In the right column of 2 we can see that if we increase the weight of the KL regularisation (β) by one order of magnitude, the reconstruction is already "severely" affected with values close to one or zero being absent from the 2D uniform distribution (sub-figure b)). If one also increases the weight of the L2 weight penalty (d) and f) one can see that the reconstruction "breaks down". The latent distribution looks more Gaussian but points that belong to quadrants in the source distribution now below to bands over one dimension of the latent distributions. As a result the information about 2D location of points gets lost. We can see a similar phenomenon in subfigure c) of figure 3 that visualizes the result of using only one single latent variable. Interestingly for low weights of the KL term the autoencoder tries to traverse the 2D space in a "snake fashion" to reconstruct as much of the original 2D distribution with a 1D geometric primitive (line).



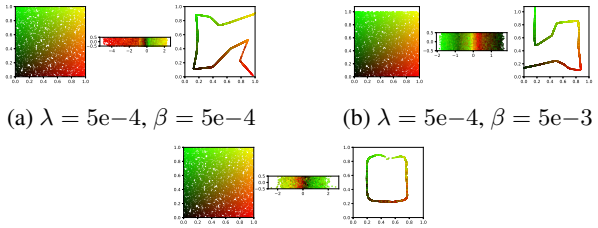
(a) $\lambda = 5\text{e}{-4}$, $\beta = 5\text{e}{-3}$ (b) $\lambda = 5\text{e}{-5}$, $\beta = 5\text{e}{-2}$

(c) $\lambda = 5\text{e}{-3}$, $\beta = 5\text{e}{-3}$ (d) $\lambda = 5\text{e}{-4}$, $\beta = 5\text{e}{-2}$

(e) $\lambda = 5\text{e}{-2}$, $\beta = 5\text{e}{-3}$ (f) $\lambda = 5\text{e}{-3}$, $\beta = 5\text{e}{-2}$

(g) $\lambda = 2.5\text{e}{-1}$, $\beta = 5\text{e}{-3}$

Fig. 2: Visualisation of the effect of L2 and KL regularisation for various settings of λ and β, 2 latent variables



(a) $\lambda = 5\text{e}{-4}$, $\beta = 5\text{e}{-4}$ (b) $\lambda = 5\text{e}{-4}$, $\beta = 5\text{e}{-3}$

(c) $\lambda = 5\text{e}{-4}$, $\beta = 5\text{e}{-2}$

Fig. 3: Visualisation of the effect of L2 and KL regularisation for various settings of λ and β, 1 latent variable. 1D latent distribution with added jitter for better visualisation

Annex

4. REFERENCES

[1] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[2] Diederik P. Kingma and Max Welling, "Auto-Encoding Variational Bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[3] Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *ICLR*, 2017.

[4] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio, "Generating sentences from a continuous space," in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, Berlin, Germany, Aug. 2016, pp. 10–21, Association for Computational Linguistics.

[5] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther, "Ladder variational autoencoders," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2016, NIPS'16, p. 3745–3753, Curran Associates Inc.