Advanced Inference and Representation Learning Methods in Variational Autoencoders

Author: Ignacio Peis Aznarte

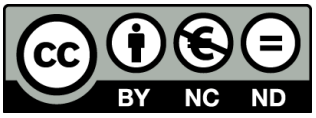
Thesis submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy in Multimedia and Communications

Universidad Carlos III de Madrid

Advisors: Antonio Artés Rodríguez, Pablo Martínez Olmos

Tutor: Antonio Artés Rodríguez

April 2023



Acknowledgements

El día que mi entonces tutor de Trabajo Fin de Grado, Diego Salas, esbozó un círculo en un papel que representaba el conocimiento humano, definitivamente logró iluminar algo en mi mente. Probablemente su objetivo era motivarme lo suficiente para realizar un trabajo largo y tedioso de programación, que no está hecho para impacientes o inconstantes, pero sin duda despertó en mí un interés. Por aquel entonces, tenía que implementar un modelo probabilístico para segmentar tejido cerebral en imágenes MRI.

Mientras dibujaba lóbulos alrededor de la circunferencia, me explicó que la investigación consigue 'estirar' el conocimiento en esas direcciones, y que una tesis doctoral se representa con los lóbulos más grandes. En aquel momento yo estaba sumido en la carrera de Teleco, y recuerdo pensar que parecía el diagrama de radiación de una antena. Ni siquiera me planteaba que tuviera algo que ver conmigo. Meses más tarde, conseguimos publicar los resultados de mi TFG en una prestigiosa conferencia de Neuroimagen celebrada en Estrasburgo, Francia, y seguidamente, una continuación de mi trabajo fue aceptada como artículo en una importante revista científica.

Una de las asignaturas que más disfruté en la carrera en Granada fue 'Comunicaciones II'', en la que estudiamos cómo se realizan transmisiones de información vía electromagnética mediante modulaciones digitales. Me gustaba tanto que fui escribiendo un 'libro' de la asignatura, explicando todos los contenidos con tono docente. En la bibliografía de esta asignatura, nos recomendaban un libro en español, '*Comunicaciones Digitale*s', que encontré extremadamente útil y con el que pasé largas tardes y noches leyendo y estudiando. No es fácil encontrar buenos libros técnicos escritos en el área de Teleco en nuestro idioma. Su primer autor es Antonio Artés Rodríguez.

Cuando comencé la búsqueda de estudios de Máster que me motivaran, encontré un programa bilingüe de doble Máster en la Universidad Carlos III de Madrid que llamó mi atención. Topé con que Antonio era Catedrático en el Departamento de Teoría de la Señal de la UC3M. Tal fue mi entusiasmo tras estudiar su libro, que le contacté por correo por si había alguna oportunidad para trabajar con él. En una respuesta casi ipsofacta, me ofreció varias alternativas, y empezamos a colaborar incluso antes de empezar a cursar el doble Máster. Siempre estaré agradecido por su acogida.

En Madrid comenzó una etapa de cambio en la que conocí a una cantidad abrumadora de personas excepcionales. Compañeros y compañeras de Máster que venían de todas partes de España, con los que compartí innumerables momentos buenos en noches de fiesta o findes de casa rural, y momentos difíciles en exámenes, proyectos y presentaciones. Uno de ellos, Fernando Moreno-Pino, el cordobés con 'alma norteña', se subió al tren del doctorado unos meses más tarde de que yo me decidiera. Aunque vivimos mil momentos juntos, siempre recordaré con especial cariño nuestro viaje de dos semanas a Moscú para asistir a la *Machine Learning Summer School*, en 2019, o nuestros meses de pareja de CrossFit. Talentosos compañeros de Departamento, entre otros Francisco Cadenas, Ignacio Melero, Lara Poveda, Juanjo Campaña-Montes o Alfredo Nazábal, quienes me acogieron desde el primer día y años más tarde fueron tomando distintos caminos laborales, todos ellos admirables. Compañeros que fueron uniéndose al grupo más adelante y en los que me vi reflejado en mis inicios, como Daniel Barrejón-Moreno, Lorena Romero-Medrano, María Martínez-García, Alejando Guerrero-López o Emese Sukei. Con ellos compartí divertidos

ratos de comedor, debates en el *Agora* y algunos viajes para el recuerdo. Grandes investigadores y docentes tanto del grupo de investigación, como Joaquín Míguez, Tobías Koch, Gonzalo Váquez, Luca Martino o Javier López, como del Departamento, como Emilio Parrado-Hernández o Marcelino Lázaro, entre otros muchos, con los compartí enriquecedoras conversaciones y de los que aprendí tanto en innumerables charlas de grupo como de la docencia que impartí junto a ellos.

En Septiembre de 2017 tuve la oportunidad de viajar con Pablo Moreno-Muñoz a Sheffield, Reino Unido, para asistir a una *Summer School* en Procesos Gausianos, momento en el que él se encontraba en la primera etapa de su doctorado. Descubrí rápidamente su tesón y su vocación investigadora. Desde entonces, hemos compartido mil 'batallas', viajes y aventuras por Europa. Pablo Sánchez-Martín nos acompañó en casi todas ellas, incluso después de 'volar' a Alemania al Instituto Max Plank. *Pablito*, a quien llamamos así por ser más joven y para diferenciar entre tanto Pablo, es, al igual que P. Moreno, un gran amigo y enorme investigador, a quien también admiro profundamente. Su capacidad para innovar y su toque característico de improvisación me han enseñado mucho estos años. Ya estamso compartiendo retos apasionantes, y estoy seguro de que habrá muchos más en el futuro. Siempre he admirado la vocación de los dos Pablos, y estoy agradecido por todo lo que he aprendido de ellos y junto a ellos.

Aún así, no tenía claro mi futuro en la investigación. Probablemente lo que vi en P. Moreno me ayudó a apostar por el doctorado, aunque otra serie de acontecimientos me acabaron convenciendo. Primero, conseguí mi tercera publicación con los resultados del primer TFM, en el que aprendí enormemente de la supervisión de Antonio y de la colaboración con prestigiosos investigadores médicos como Enrique Baca-García. A partir de datos de sensores inerciales, conseguimos una estimación razonable de la fecha de alta de pacientes internos con trastorno depresivo. Segundo, para realizar mi segundo TFM, conté con la guía de Pablo M. Olmos, un investigador y profesor que me impresionó por su carácter amigable y su perspicacia. Antonio, Pablo y yo formamos un buen equipo, lo que se evidenció cuando publicamos los resultados de este TFM como mi cuarto artículo científico. Conseguimos predecir la ideación suicida de pacientes con registros de visitas hospitalarias que además usaban una aplicación con cuestionarios en su smartphone. Olmos me inculcó la pasión por las Redes Neuronales y el modelado generativo, y a él también le agradezco que acabara decidiéndome por la investigación, ya que para ello ayudó que propusiera ser mi codirector de mi tesis, junto con Antonio.

Los años de doctorado siempre los recordaré como apasionantes. La simple satisfacción de *entender* artículos y modelos cada vez más complejos, tan solo a base de esfuerzo y dedicación, fue el primer impulso. Después, confirmar que podía dar forma a mis propias ideas y mostrar los resultados como contribuciones de alto impacto, terminó de embaucarme. Tras un primer trabajo ilusionante con mis directores, presentado en esta tesis como UG-VAE, comencé una prometedora colaboración con José Miguel Hernández Lobato, Catedrático en la Universidad de Cambridge y uno de los líderes del grupo de Machine Learning en el *Department of Engineering*. Este fue el reto más apasionante al que me he enfrentado, a la par que exigente. Convivir y trabajar con algunos de los investigadores más importantes del mundo en Machine Learning fue una etapa que me permitió adquirir una nueva perspectiva, y fui capaz de exprimir al máximo aquella oportunidad. Además, conocer la hermosa ciudad de Cambridge, una de las cunas mundiales de la investigación, asistir a *formal dinners* en varios *colleges*, y convivir con personas tan interesantes venidas de todas partes del mundo, fueron experiencias extremadamente enriquecedoras. Me gustaría agradecer a Melanie Fernández-Pradier, antigua doctoranda del grupo GTS, quien me recibió amablemente en las oficinas de Microsoft Research Cambridge, donde trabaja actualmente, y a quien considero un referente. Gracias a la guía de José Miguel y a mi

perseverancia, acabé presentando este proyecto, nombrado HH-VAEM en esta tesis, como artículo en la conferencia NeurIPS, en Nueva Orleans, EE UU, considerada por muchos como la más importante a nivel mundial de la investigación en Machine Learning.

Solo he hablado hasta aquí de mi progreso como científico y lo agradecido que estoy a las personas que he ido encontrando en mi camino a ser Doctor. Además, también agradeceré a continuación a las personas protagonistas en mi vida que han hecho posible que consiga mis retos, y relataré una serie de acontecimientos paralelos a toda esta historia de tesis que pusieron a prueba mi entereza.

Aunque el trabajo duro y la ilusión siempre son los mayores motores, tengo claro que mis padres, Jose y Elena, son el núcleo de todo lo que he ido conseguiendo. Ellos, junto con mis hermanos, Celia y Joaquín, son los mayores partícipes de lo que hoy soy, y me gustaría que supieran que siempre seré consciente de ello. Elena me ha mostrado cuánta entereza y valentía puede haber contenidas en una persona. Desde el primer día, allá por Marzo de 2017, supo plantarle cara a la enfermedad más temida, pelear duro y gritarle sin tapujos que ella era más fuerte. Cuando venció, nos demostró a todos que cuando la vida golpea y los 'porqués' no tienen respuesta, esa es la única actitud posible. Sin embargo, su rival atacó fuerte otra vez, en 2019, en plena pandemia. Sí, pandemia. Sigue sonando a ficción, ahora que empezamos a olvidarla. Una época apocalíptica que contaremos en los libros de Historia dentro de unos años. Si los hospitales no hubieran estado colapsados, si toda la familia hubiera podido estar con nosotros, si hubiéramos tenido el hospital más cerca, si no hubiéramos perdido amigos, todo hubiera sido más fácil. A pesar de todo, consiguió vencer de nuevo. Gracias por haberme enseñado tanto.

Jose me inculcó la humildad, la perseverancia, la protección, el amor incondicional a los tuyos y la dedicación a todo lo que haces. Con él compartí los momentos más difíciles. Me enseñó que siempre hay dónde encontrar fuerza para seguir peleando. Celia, la sucesora médica de Jose, demuestra cada día su rigor clínico y su capacidad de aprender y analizar. Está a punto de empezar a comerse el mundo. Gracias a ella aprendí que todas las cosas buenas que hace un hermano mayor podrían serle útiles como referencia. El mismo sentimiento de ejemplo lo comparto hacia Joaquín, la alegría de la casa, en quien me veo reflejado. Él ha tomado el camino de lo que en su momento, confieso, fue mi segunda opción: la Arquitectura. Su talento y la perseverancia que está mostrando ya en los primeros cursos apuntan, al igual que para Celia, a un futuro exitoso que, como hermano mayor, estoy deseando ver para poder sentirme orgulloso.

Además de elllos, otros protagonistas en mi vida han contribuido a que pueda culminar esta tesis. Ángela, mi compañera, me llena a diario de vida, luz e ilusiones y me aguanta en mis peores y mejores momentos. Mi prima y alma gemela Ali, me entiende sin palabras y comparte conmigo sangre y vida. Mi querida amiga Alicia, con la que llevo compartiendo vivencias, inquietudes y pensamientos profundos desde nuestra época puberal. También mis amigos de Jaén, quienes siempre están disponibles en las buenas y en las malas y me han acompañado en el camino desde que éramos unos críos de 4 años. De ellos destaco su 'Fuga de Cerebros' fallida a Cambridge. ¡Sois muchísimos y ocuparía media tesis nombraros a todos! También quiero agradecer a todos mis tíos, primos y a mi Abuela, por ser ejemplos de familia unida, en la que siempre prevalece el amor y el cuidado mutuo.

Por último, me gustaría recordar a mi querido abuelo, Don Joaquín Peis, que siempre será mi mayor héroe y referente. Él siempre tuvo ese brillo en los ojos que es necesario para conseguir todo lo que uno se proponga. Sus 'polvos mágicos' seguirán dándome fuerza allá donde vaya.

Hard work always give you a chance.

Included herein is a list of publications, presented in bibliographic format, which have resulted from various projects undertaken throughout the course of this doctoral thesis. It is important to note that references 2 and 3, although published during the progression of the doctorate, originated from projects initiated prior to the commencement of the doctoral program. Consequently, these particular references are not extensively discussed within the chapters of the present thesis. For the sake of clarity: *The enumeration of the publications contained on this page shall not be taken into account for referencing purposes within the main body of the text.*

Journals

1. I. Peis, P. M. Olmos and A. Artés-Rodríguez. Unsupervised Learning of Global Factors in Deep Generative Models. In *Pattern Recognition, 134, 109130*, 2023. [pdf] – *Complete* presence in Chapter 4.

2. I. Peis, J. D. López-Moríñigo, M. M. Pérez-Rodríguez, M. L. Barrigón, M. Ruiz-Gómez, A. Artés-Rodríguez and E. Baca-García. Actigraphic recording of motor activity in depressed inpatients: a novel computational approach to prediction of clinical course and hospital discharge. In *Scientific reports, 10. Nature*, 2020. [pdf] – *Partial* presence in Chapter 6.

3. I. Peis, P. M. Olmos, C. Vera-Varela, M. L. Barrigón, P. Courtet, E. Baca-García and A. Artés-Rodríguez. Deep Sequential Models for Suicidal Ideation from Multiple Source Data. In *IEEE Journal of Biomedical and Health Informatics, 23(6), 2286-2293*, 2020. [pdf] – *Partial* presence in Chapter 2.

Conferences

4. I. Peis, C. Ma and J. M. Hernández-Lobato. Missing Data Imputation and Acquisition with Deep Hierarchical Models and Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems (NeurIPS) 35*, 2022. [pdf] – *Complete* presence in Chapter 5.

5. B. Koyuncu, P. Sánchez-Martín, I. Peis, P. M. Olmos and I. Valera. Variational Mixture of HyperGenerators for Learning Distributions Over Functions. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. [accepted] [pdf] – *Partial* presence in Chapter 2 and Chapter 3.

Abstract

Deep Generative Models have gained significant popularity in the Machine Learning research community since the early 2010s. These models allow to generate realistic data by leveraging the power of Deep Neural Networks. The field experienced a significant breakthrough when Variational Autoencoders (VAEs) were introduced. VAEs revolutionized Deep Generative Modeling by providing a scalable and flexible framework that enables the generation of complex data distributions and the learning of potentially interpretable latent representations. They have proven to be a powerful tool in numerous applications, from image, sound and video generation to natural language processing or drug discovery, among others. At their core, VAEs encode natural information into a reduced latent space and decode the learned latent space into new synthetic data. Advanced versions of VAEs have been developed to handle challenges such as handling heterogeneous incomplete data, encoding into hierarchical latent spaces for representing abstract and richer concepts, or modeling sequential data, among others. These advances have expanded the capabilities of VAEs and made them a valuable tool in a wide range of fields.

Despite the significant progress made in VAE research, there is still ample room for improvement in their current state-of-the-art. One of the major challenges is improving their approximate inference. VAEs typically assume Gaussian approximations of the posterior distribution of the latent variables in order to make the training objective tractable. The parameters of this approximation are provided by encoder networks. However, this approximation leads to a lower bounded objective, which can degrade the performance of any task that requires samples from the approximate posterior, due to the implicit bias. The second major challenge addressed in this thesis is related to achieving meaningful latent representations, or more broadly, how the latent space disentangles generative factors of variation. Ideally, the latent space would modulate meaningful properties separately within each dimension. However, Maximum Likelihood optimizations require the marginalization of latent variables, leading to non-unique solutions that may or may not achieve this desired disentanglement. Additionally, properties learned at the observation level in VAEs assume that every observation is generated independently, which may not be the case in some scenarios. To address these limitations, more robust VAEs have been developed to learn disentangled properties at the supervised group (also referred to as global) level. These models are capable of generating groups of data with shared properties.

The work presented in this doctoral thesis focuses on the development of novel methods for improving the state-of-the-art in VAEs. Specifically, three fundamental challenges are addressed: achieving meaningful global latent representations, obtaining highly-flexible priors for learning more expressive models, and improving current approximate inference methods. As a first main contribution, an innovative technique named UG-VAE from Unsupervised-Global VAE, aims to enhance the ability of VAEs in capturing factors of variations at data (local) and group (global) level. By carefully desigining the encoder and the decoder, and throughout conductive experiments, it is demonstrated that UG-VAE is effective in capturing unsupervised global factors from images. Second, a non-trivial combination of highly-expressive Hierarchical VAEs with robust Markov

Chain Monte Carlo inference (specifically Hamiltonian Monte Carlo), for which important issues are successfully resolved, is presented. The resulting model, referred to as the Hierarchical Hamiltonian VAE model for Mixed-type incomplete data (HH-VAEM), addresses the challenges associated with imputing and acquiring heterogeneous missing data. Throughout extensive experiments, it is demonstrated that HH-VAEM outperforms existing one-layered and Gaussian baselines in the tasks of missing data imputation and supervised learning with missing features, thanks to its improved inference and expressivity. Furthermore, another relevant contribution is presented, namely a sampling-based approach for efficiently computing the information gain when missing features are to be acquired with HH-VAEM. This approach leverages the advantages of HH-VAEM and is demonstrated to be effective in the same tasks.

Resumen

Los Modelos Generativos han ganado una gran popularidad en la comunidad de investigación de Aprendizaje Automático desde principios de la década de 2010. Estos modelos permiten generar datos realistas aprovechando la capacidad de las Redes Neuronales Profundas. El campo experimentó un avance significativo cuando se introdujeron los Autoencoders Variacionales (VAEs). Los VAEs revolucionaron los Modelos Generativos Profundos al proporcionar un marco escalable y flexible que permite la generación de distribuciones de datos complejas y el aprendizaje de representaciones latentes potencialmente interpretables. Han demostrado ser una herramienta poderosa en numerosas aplicaciones, desde la generación de imágenes, sonido y video hasta el procesamiento del lenguaje natural o el descubrimiento de medicamentos, entre otros. En su definición básica, los VAEs codifican información natural en un espacio latente reducido y decodifican el espacio latente aprendido en nuevos datos sintéticos. Se han desarrollado versiones avanzadas de VAEs para manejar desafíos como el manejo de datos incompletos heterogéneos, la codificación en espacios latentes jerárquicos para representar conceptos abstractos y más ricos, o el modelado de datos secuenciales, entre otros. Estos avances han ampliado las capacidades de los VAEs y los han convertido en una herramienta valiosa en una amplia gama de campos.

A pesar del progreso significativo en la investigación en VAEs, todavía hay amplio margen para mejorar el estado del arte. Uno de los principales desafíos es mejorar su inferencia aproximada. Los VAEs típicamente asumen aproximaciones Gausianas de la distribución posterior de las variables latentes para hacer que el objetivo de entrenamiento sea computable. Los parámetros de esta aproximación son proporcionados por la red de codificadora. Sin embargo, esta aproximación conduce a un objetivo sesgado, lo que puede degradar el rendimiento de cualquier tarea que requiera muestras de esta distribución posterior, debido al sesgo implícito. El segundo desafío importante abordado en esta tesis se relaciona con lograr representaciones latentes significativas o, más ampliamente, cómo el espacio latente organiza los factores generativos de variación. Idealmente, el espacio latente modularía propiedades significativas por separado en cada dimensión. Sin embargo, las optimizaciones de Máxima Verosimilitud requieren la marginalización de las variables latentes, lo que lleva a soluciones no únicas que pueden o no lograr esta organización deseada. Además, las propiedades aprendidas a nivel de observación en los VAEs asumen que cada observación se genera de manera independiente, lo que puede no ser el caso en algunos escenarios. Para abordar estas limitaciones, se han desarrollado VAEs más robustos para aprender propiedades organizadas a nivel de grupo (también denominado nivel global) de manera supervisada. Estos modelos son capaces de generar grupos de datos con propiedades compartidas.

El trabajo presentado en esta tesis doctoral se centra en el desarrollo de nuevos métodos para mejorar el estado del arte en VAEs. Específicamente, se abordan tres desafíos fundamentales: lograr representaciones latentes globales interpretables, obtener *priors* altamente flexibles para aprender modelos más expresivos y mejorar los métodos de inferencia aproximada actuales. Como primera contribución principal, se presenta una técnica innovadora llamada UG-VAE de Unsupervised-Global VAE, que tiene como objetivo mejorar la capacidad de los VAEs en la captura de factores de variación a nivel de datos

(local) y grupo (global). A través de los experimentos llevados a cabo, se demuestra que UG-VAE es efectivo en la captura de factores globales no supervisados a partir de imágenes mediante el diseño cuidadoso del codificador y decodificador. En segundo lugar, se presenta una combinación no trivial de VAEs jerárquicos altamente expresivos con una inferencia robusta mediante *Markov Chain Monte Carlo* (específicamente *Hamiltonian Monte Carlo*), para la cual se resuelven con éxito importantes problemas. El modelo resultante, denominado HH-VAEM por VAE jerárquico con Hamiltonian Monte Carlo para datos incompletos heterogéneos, aborda los desafíos asociados con la imputación y adquisición de datos perdidos heterogéneos. A través de extensos experimentos, se demuestra que HH-VAEM supera a las alternativas existentes de una capa y basados en aproximaciones Gausianas en las tareas de imputación de datos perdidos y aprendizaje supervisado con datos parciales, gracias a su mejora en la inferencia y expresividad. Además, se presenta como otra contribución relevante, un método basado en muestreo para calcular eficientemente la ganancia de información cuando se adquieren variables perdidas con HH-VAEM. Este enfoque aprovecha las ventajas de HH-VAEM y se demuestra que es efectivo en las mismas tareas.

CHAPTER 1

INTRODUCTION

DEEP Generative Models gained popularity in the Machine Learning research community in the early 2010s, allowing researchers to generate realistic data in an unsupervised manner, leveraging the power of Deep Neural Networks. The field experienced a significant breakthrough when Variational Autoencoders (VAEs) were introduced (Kingma and Welling, 2013). VAEs revolutionized Deep Generative Modeling by providing a scalable and flexible framework that enables the generation of complex data distributions and the learning of potentially interpretable latent representations. They have proven to be a powerful tool in numerous applications, from image (Razavi et al., 2019b; Vahdat and Kautz, 2020; Child, 2020), video (Yan et al., 2021; He et al., 2018; Bhagwatkar et al., 2021), music (Roberts et al., 2018), or text generation (Bowman et al., 2015), to neural machine translation (Sutskever et al., 2014), outlier detection (Chauhan et al., 2022; Denouden et al., 2018; Xiao et al., 2020; Serrà et al., 2019), time-series analysis (Tang and Matteson, 2021a; Chung et al., 2015; Fraccaro et al., 2016) or recommendation systems (Shenbin et al., 2020; Liang et al., 2018). Moreover, their flexibility and interpretability have made them a popular choice for researchers and practitioners alike.

Despite the significant progress made in VAE research, there is still room for improvement in their current state-of-the-art. It is essential to continue exploring and developing new techniques to enhance their capabilities, such as improving sample quality, handling missing data, or enhancing interpretatbility of the latent variables. The present thesis focuses on the development of novel methods for improving the state-of-the-art in VAEs. Specifically, three fundamental challenges are addressed: achieving meaningful latent representations, obtaining highly-flexible priors for learning more expressive models and improving current approximate inference methods. As a first main contribution, innovative techniques for improving the interpretability of the latent spaces in VAEs are proposed. Second, a non-trivial combination of highly-expressive Hierarchical VAEs with robust Markov Chain Monte Carlo inference, for which important issues are successfully resolved, is presented. Overall, the work presented in this doctoral thesis represents a significant contribution to the ongoing effort to improve the state-of-the-art in VAEs and has the potential to pave the way for new breakthroughs in deep generative modeling.

1.1. VAEs and improvement directions

Variational Autoencoders (VAEs) (Kingma and Welling, 2013; Rezende et al., 2014) are likelihood-based deep generative models that approximate the intractable true posterior over latent variables $p(\boldsymbol{z}|\boldsymbol{x})$, where $\boldsymbol{x}$ is an observed datapoint, by introducing an auxiliary model to perform amortized variational inference with an encoder-decoder architecture. The encoder network, parameterized by $\boldsymbol{\phi}$, maps observations $\boldsymbol{x} \in \mathbb{R}^D$ to the parameters of the approximate posterior $q_\phi(\boldsymbol{z}|\boldsymbol{x})$, typically Gaussian, and with lower dimensionality $\boldsymbol{z} \in \mathbb{R}^d$. The decoder network, parameterized by $\boldsymbol{\theta}$, maps the latent variable sampled from this approximate posterior to the parameters of the data likelihood

$p_\theta(\boldsymbol{x}|\boldsymbol{z})$. Ideally, within a maximum likelihood optimization, the log-evidence, $\log p(\boldsymbol{x})$, would be maximized to learn the optimal parameters of the model. However, due to the complexity added by the neural networks, this quantity is intractable. The objective function to be maximized is an approximation, referred to as the Evidence Lower Bound (ELBO),

$$\mathcal{L}(\boldsymbol{x}) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right] - D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})), \tag{1.1}$$

which is a Lower Bound of the intractable log-evidence. The ELBO encourages proper data reconstruction via the first term, whilst minimizing mismatch between the approximate posterior and prior via the second term. Within the simplest setting, the prior $p(\boldsymbol{z})$ is modeled by a standard Gaussian. However, when complex data spaces are to be encoded in the latent space, **more flexible priors**, typically parameterized by Neural Networks, are required to avoid a significant mismatch between the aggregated posterior and the prior, typically referred to as the *holes problem* (Rezende and Viola, 2018). On the contrary, too powerful decoders might lead to uninformative posteriors, i.e. the posterior is ignored by matching it with the prior, and the data is decoded using autoregressive dependencies in $\boldsymbol{z}$, a problematic known in the literature as *posterior collapse*.

1.1.1. Representation learning

Deep Generative Models are designed to uncover the underlying factors present in the data they observe (Bengio et al., 2013). VAEs achieve this by learning a joint distribution of observed data, denoted by $\boldsymbol{x}$, and hidden factors, denoted by $\boldsymbol{z}$, represented as $p(\boldsymbol{x}, \boldsymbol{z})$. The crucial aspect of obtaining a meaningful representation is determining the posterior distribution of the hidden factors, denoted by $p(\boldsymbol{z}|\boldsymbol{x})$. However, as pointed out by (Tomczak, 2021), learning a latent variable model by maximizing the likelihood function may not lead to useful representations. As a result, it can be challenging to learn useful latent representations with latent variable models.

When designing such generative models that compress high dimensional complex data $\boldsymbol{x}$ into a reduced latent space $\boldsymbol{z}$, it is of great interest to achieve **meaningful latent representations** (Mathieu et al., 2019b). The quality of *meaningful* is related to how the latent dimensions explain the underlying generative factors of the data. To provide with an example, a dimension of a latent space for generating face images, might ideally encode features like the angle of the face, skin or hair color.

Obtaining high interpretability of latent dimensions in VAEs is an open research question. One possible solution is to carefully select a suitable class of models, such that the structure of the latent space is designed to obtain meaningful representations with an inductive bias. An example of such models are hierarchical VAEs (Vahdat and Kautz, 2020; Child, 2020; Maaløe et al., 2019), which have recently demonstrated success in learning interpretable latent spaces. This success can be attributed to their similarity with the flow of information in the real world, from general abstract concepts to more specific and unique features. Other example is the usage of mixture models (Dilokthanakul et al., 2016) that allow for clusterizing the latent space leading to richer structures.

A second direction for addressing this challenge is to assess and encourage *disentanglement*, typically by choosing between two different alternatives. First, recent works propose adding additional terms to the ELBO (Mathieu et al., 2019b; Higgins et al., 2018; Tomczak and Welling, 2018), or modifying it with learnable factors (Higgins et al., 2016), in order to balance between learning meaningful representations or achieving more accurate reconstructions.

The third strategy employed in other works rely on enforcing disentanglement by

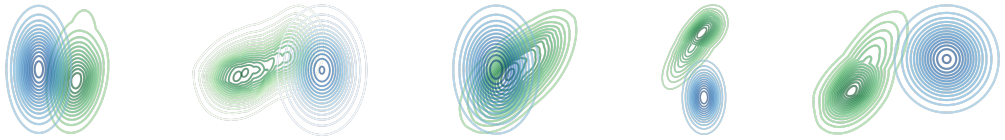


Figure 1.1: Comparison of an accurate (computationally demanding) approximation of the true posterior, $\hat{p}(\boldsymbol{z}|\boldsymbol{x})$ (green contours), and Gaussian proposal learned by the encoder, $q(\boldsymbol{z}|\boldsymbol{x})$ of a VAE (blue contours) with $\boldsymbol{z} \in \mathbb{R}^2$ trained with the MNIST digit dataset.

incorporating extra information through semi-supervision. For example, in (Bouchacourt et al., 2018), authors incorporate group-level information, such as the identity of a face image, to account for a higher level of abstraction in learning generative factors of the faces.

The aforementioned strategies for monitoring deep generative models to achieve meaningful representations have shown important results. However, the current state of the art lacks methods that can learn disentangled representations with any type of supervision. In this thesis, a novel method for achieving this is presented as the **first main research contribution**.

1.1.2. Approximate inference

Another challenge within VAEs to be handled is approximate inference. Due to the intractability of the log-marginal likelihood function $\log p(\boldsymbol{x})$, which is the objective for which the parameters should be ideally optimized following a Maximum Likelihood approach, approximations are required, being the ELBO of (1.1) the typical choice. In the naïve VAE approaches (Kingma and Welling, 2013), Gaussian distributions are employed as the variational proposal $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. Although they generally achieve reasonably accurate approximations (Cremer et al., 2018; Zhang et al., 2018), and have proven to be effective, scalable and valid for training VAEs, the simplicity of this choice compared with the complexity of the true posterior increases with the high dimensionality of complex datasets. As depicted in Figure 1.1, for complex datasets with high-dimensionality, the shape of the true posterior is far from being Gaussian-shaped. Several factors, including the expressive capacity of the encoder/decoder, the flexibility of the prior, or the complexity of the variational proposal, determine the suboptimality of this approximation (Cremer et al., 2018). Hence, reducing this gap or considering **alternatives for better approximate inference** is another relevant research direction found in the literature of VAEs.

For instance, in (Burda et al., 2015), authors propose an alternative objective, inspired by *importance weighting*, and demonstrate its impressive efficacy in getting more accurate approximations of the log evidence. More recent methods have adapted the Importance Weighted Autoencoder (IWAE) to handling incomplete data (Mattei and Frellsen, 2019), which is another topic of interest in this thesis. Another approach is to use Markov Chain Monte Carlo (MCMC) methods to obtain more accurate samples from the posterior distribution (Salimans et al., 2015; Campbell et al., 2021). Several papers have proposed using MCMC methods such as Hamiltonian Monte Carlo (HMC), to modify the ELBO in various ways, including introducing auxiliary inference functions and optimizing the reverse kernels (Salimans et al., 2015; Wolf et al., 2016; Caterini et al., 2018). Others like (Ruiz et al., 2021) focus in obtaining efficient unbiased estimators for directly approximating the gradients of the true log-likelihood.

This thesis is focused on the direction for obtaining more accurate samples that closely follow the true posterior. Concretely, in the **second main contribution**, a novel method for automatically training HMC hyperameters for accurately sampling from the complicated posterior a highly-expressive hierarchical VAE is presented.

1.2. Overview of Models and Contributions

This thesis is oriented to the design, implementation and evaluation of novel deep generative models that overcome the aforementioned challenges of obtaining meaninful latent representations, highly expressive models with flexible priors, and more accurate inference approximations for enhancing VAEs in their current state-of-the-art. The principal contributions are packed into two main relevant works, both accepted as a journal publication (Peis et al., 2023) and a conference paper (Peis et al., 2022). Secondary research contributions out of the main research direction of improving VAEs are mentioned in their related subsections of Chapter 2. In the following, both principal contributions are described.

1.2.1. Unsupervised learning of global factors

The first relevant contribution of this thesis (Peis et al., 2023) is a novel deep generative model based on non i.i.d. variational autoencoders that captures global dependencies among observations in a fully unsupervised fashion. The model is referred to as the **Unsupervised-Global VAE** (**UG-VAE**). In contrast to the recent semi-supervised alternatives for global modeling in deep generative models, this approach combines a mixture model in the local or data-dependent space and a global Gaussian latent variable, which leads to obtain three particular insights. First, the induced latent global space captures interpretable disentangled representations with no user-defined regularization in the evidence lower bound (as in β-VAE and its generalizations). Second, the model performs domain alignment to find correlations and interpolate between different databases. Finally, the global space learns to discriminate between groups of observations with non-trivial underlying structures, such as face images with shared attributes or defined sequences of digits images. These conclusions are conducted through extensive experiments.

1.2.2. Hierarchical VAEs and Hamiltonian Monte Carlo

The second relevant contribution of the present thesis (Peis et al., 2022) is a novel VAE framework for imputing and acquiring heterogeneous missing data. Within this specific application domain, existing VAE methods are restricted by using only one layer of latent variables and strictly Gaussian posterior approximations. To address these limitations, the **HH-VAEM** is presented as a **Hierarchical Hamiltonian VAE model for mixed-type incomplete data** that uses Hamiltonian Monte Carlo with automatic hyper-parameter tuning for improved approximate inference. Troughout extensive experiments, it is demonstrated that HH-VAEM outperforms existing baselines in the tasks of missing data imputation and supervised learning with missing features, thanks to its improved inference and expressitivy. Further, another relevant contribution, namely a sampling-based approach for efficiently computing the information gain when missing features are to be acquired with HH-VAEM, is jointly presented, leveraging the advantages of HH-VAEM.

1.3. Thesis Organization

The present doctoral manuscript is divided into four main chapters, that we shortly review in the following paragraphs for a better comprehension of the document and its organization. The references to the principal published pieces of work, where contributions were initially presented, are included in the introductory lines of Chapters 4 and 5, as well as in related subsections in Chapters 2 and 3, where secondary contributions are further referenced.

Chapter 2: Deep Generative Models

This chapter serves as an introduction to deep generative modeling, which comprises the main framework of Variational Autoencoders. The first section motivates the use of Probabilistic Machine Learning as a robust perspective against deterministic approaches. A concise revision of Probability Theory is included, covering several distribution types, divergence for similarity measures, Bayesian inference, and Monte Carlo approximations. The second part introduces generative modeling, including classic unsupervised methods for generating data from latent variables. The third section briefly describes the field of Deep Learning, culminating in a motivation for utilizing its advantages in generative modeling. Finally, the chapter provides a succinct overview of Deep Generative Models, classifying them and emphasizing the advantages of each type.

Chapter 3: Variational Autoencoders

This chapter centers around Variational Autoencoders, the focal models of this thesis. The chapter begins with an introductory presentation of deterministic autoencoding frameworks, leading to the presentation of the base VAE model, which includes amortized variational inference and the definition of the Evidence Lower Bound. The chapter then briefly reviews the typical issues, applications, and directions for improving VAEs, culminating in sections closely related to the contributions of this thesis, including prior design, hierarchical VAEs, representation learning, and approximate inference. The chapter concludes by describing current methods for adapting VAEs to handle heterogeneous incomplete data and the balancing between increasing model flexibility and reducing inference bias.

Chapter 4: Unsupervised learning of global factors in VAEs

This chapter presents the first of the principal contributions of the present thesis. The Unsupervised Global VAE (UG-VAE) is introduced as a novel method for learning meaningful, highly interpretable latent representations at local and global level from batches of data randomly selected. An overview on related previous methods that utilize semi-supervision, ELBO modifications or inductive bias via model design is provided. Afterwards, the method is presented by describing its parts, including the generative and inference models. An extensive experimental sections provides empirial evidence of the contributions.

Chapter 5: Hierarchical VAEs and Hamiltonian Monte Carlo

This chapter presents the second of the main principal contributions of this thesis. A novel method, referred to as HH-VAEM from Hierarchical Hamiltonian VAE for mixed-type incomplete data, is presented. After reviewing the related work and state-of-the-art

in VAEs for incomplete data, hierarchical VAEs and introducting Hamiltonian Monte Carlo, the components of HH-VAEM and the importance of solved challenges are carefully described. In a separate section, a novel method for acquiring missing data and performing active learning in a sampling-based aproach is presented, that leverages the advantages of HH-VAEM. In the experimental section, results show the superiority of the proposed model and active learning method with respect to the alternatives.

Chapter 6: Conclusions and Future Work

The thesis is concluded in this chapter by surveying the main technical contributions, as well as the possible future research directions for the principal contributions, conveying technical and applied research.

CHAPTER 2

Deep Generative Models

The exponential growth of Machine Learning techniques and their application in a vast list of topics in the recent decades has been significant. The field of Pattern Recognition focuses on finding patterns in data through computer algorithms, allowing for useful predictions based on these discovered regularities. As an example, consider the problem of categorizing handwritten digits, depicted in Figure 2.1. Each digit is portrayed as a 28 x 28 pixel image, which can be represented by a 784-dimensional vector. The objective is to develop a machine that can take this data vector $\boldsymbol{x}$ as input and output the digit's identity, y, ranging from 0 to 9. Due the variability in handwriting, this is an arduous task to solve. Intuitively, one might think that the solution would be to create custom rules or heuristics based on the shapes or strokes of the digits, but this approach leads to a complex network of rules and exceptions, resulting in limited performance.

A more effective method is to use a Machine Learning approach, where a large set of N digits $(\boldsymbol{x}, ..., \boldsymbol{x}_N)$ referred to as a **training set** is utilized to optimize the parameters of an learnable model. The digits' categories in the training set are pre-determined, usually through manual inspection and labeling. The category of each digit can be represented using a target vector, $\boldsymbol{y}$. The Machine Learning algorithm can be defined as the process of learning an accurate function $\boldsymbol{y} = f(\boldsymbol{x})$ that takes in a new digit image $\boldsymbol{x}$ as input and outputs a vector $\boldsymbol{y}$ that tries to predict the original labels of the training set. The specific form of the function $\boldsymbol{y} = f(\boldsymbol{x})$ is established during the training stage. For instance: a linear function is considered in the toy example of Figure 2.2a. Once the model is trained, it can predict the category of new digit images, non observed by the model during training, that form the **test set**. The capability of accurately categorizing new examples that are different from the training data is referred to as **generalization**. In real-world applications, the variability of the input vectors is such that the training data is only a tiny fraction of all possible input vectors, making generalization a major goal in Machine Learning and pattern recognition.

The previous example focuses on one of the two main types of Machine Learning models: **supervised learning**, which assumes that both input data $\boldsymbol{x}$ and target data $\boldsymbol{y}$ are available during training, and the goal is to learn a mapping from $\boldsymbol{x}$ to $\boldsymbol{y}$ in order to make predictions. When the target is discrete, i.e. $y \in \{1, ..., C\}$, the considered task is **classification**, whilst if the target is continuous, $y \in \mathbb{R}^D$, the task is referred to as **regression**. In contrast, in **unsupervised learning**, the data is $\boldsymbol{x}$ is available and the goal is to discover the underlying structure of the data itself, by learning the parameters θ of a model in order to express the probability of the data: $p(\boldsymbol{x}|\theta)$. In other terms, the considered task is density estimation. It is more related to how human and animal learning occurs, and it avoid the costs of manually labeling data under supervised scenarios. Within the unsupervised setting, Generative Models are a type of machine learning model that aim to learn the underlying distribution of the data in order to generate new samples that are similar to the original data.

In the context of generative models, Deep Neural Networks (DNNs) can be used as powerful function approximators that parameterize probability distributions. This has

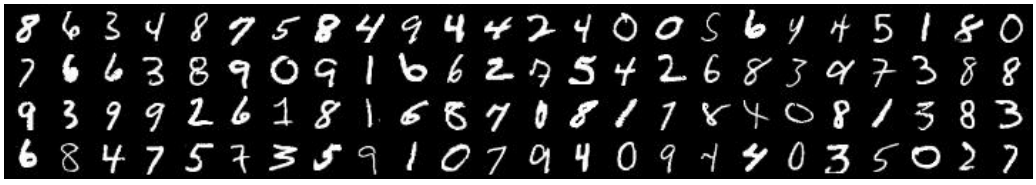


Figure 2.1: Training samples from the MNIST handwritten digits dataset (LeCun, 1998).

led to the development of Deep Generative Models (DGMs), which are generative models that use DNNs as their main building blocks. DGMs come in different types, such as Variational Autoencoders (VAEs), Generative Adversarial Networks (GANs), and Flow-based nodels, Diffusion models, etc. They are capable of generating high-quality, realistic samples from complex distributions, such as images, audio, video or text, among others.

In this chapter, a comprehensive examination of deep generative models is presented, beginning with a broad overview that may assist readers without prior knowledge to develop an intuitive understanding of deep generative modeling. The objectives of this chapter are: i) to introduce and motivate a Probabilistic Machine Learning perspective, ii) to provide a concise summary of recent Deep Learning approaches, and iii) to explain the categorization of current deep generative models as probabilistic models that exploit the capabilities of deep neural networks.

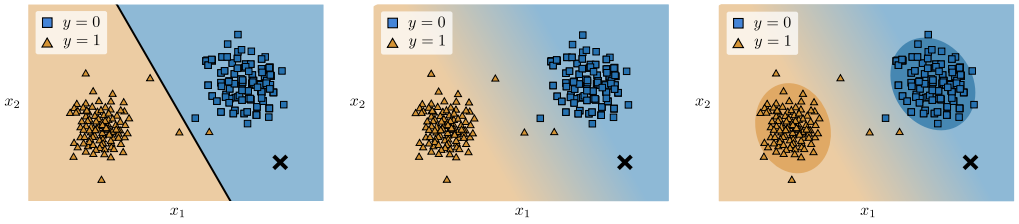
2.1. Probabilistic Machine Learning

In general, it is of vital importance for Machine Learning models to possess the capacity to quantify their level of uncertainty in regards to the environment in which they operate. The trustworthiness of a system that lacks this ability is questionable, as even a slight disturbance in its internal beliefs may result in a shift in its level of confidence and decision-making outcomes. Furthermore, effective communication with a system that is unable to clearly express its views on the novelty of its surroundings becomes a challenge.

To emphasize the significance of **uncertainty quantification** in the decision-making process, an example of three systems that categorize objects into two groups, namely orange and blue, is depicted in Figure 2.2. Given a training set $\mathcal{D} = \{\boldsymbol{x}_{1:N}, y_{1:N}\}$ of two-dimensional data $\boldsymbol{x}_{1:N} = \{x_1, x_2\}_{1:N}$ with $\boldsymbol{x}_i \in \mathbb{R}^2$ and binary targets $y_{1:N}$ with $y_i \in \{0, 1\}$, and a new data point $\boldsymbol{x}^*$ (represented by a black cross), we can adopt three methods to make a decision. Firstly, in 2.2a, a linear classifier is established by modeling the predicted class with a **deterministic** function of the inputs. All the samples that lied into the blue region will be categorized as negative samples, modeled with $f(\boldsymbol{x}) = 0$, independently of their distance to the boundary. Consequently, if the new sample was positive, this model would be incorrectly certain about classifying it as negative. This simple model can be trained by minimizing the classification error in the training set (or maximizing the accuracy). In contrast, in 2.2b, a *logistic regression* is considered, where a Bernoulli likelihood function (Section 2.1.1), parameterized by a learnable linear regression, models the probability of the output assigned class. Here, the model can be trained by maximizing the likelihood probability, or equivalently, as it will be discussed later in Section 2.1.1, by minimizing the *binary cross entropy* loss. When a test sample falls far to the linear decision boundary, the probability of the assigned class increases, thus introducing a simple method for measuring uncertainty.

In Figure 2.2c, a probabilistic model on both the predictions and input data is consid-

ered, which additionally learns a distribution over the input data, $p(\boldsymbol{x})$. Here, additional information can be captured: the black cross is not only more likely to be negative, but also distant from the region of high probability mass. The joint $p(\boldsymbol{x}, y)$ will be low, as it accounts for both criteria, ending a highly uncertain decision.



(a) $f(\boldsymbol{x})$ (b) $p(y|\boldsymbol{x})$ (c) $p(y, \boldsymbol{x}) = p(y|\boldsymbol{x})p(\boldsymbol{x})$

Figure 2.2: Uncertainty quantification example in a simple binary classification scenario with 2-dimensional data. Deterministic classifier (a) vs Logistic Regression (b) and (c) a generative approach. Background color indicates $p(y|\boldsymbol{x})$.

Overall, fitting probability distributions with machine learning models enables the quantification of uncertainty over the predicted target value given any new input value, which is crucial for many downstream tasks. Typically, probabilistic machine learning models are trained by maximizing the (log) probability of the observed data with respect to the learnable parameters θ, i.e., $\log p_\theta(\mathcal{D})$, where $\mathcal{D}$ refers to a dataset that includes input data $\boldsymbol{x}$ and optionally supervised labels $\boldsymbol{y}$, depending on the considered task. As discussed later in this thesis, $\log p_\theta(\mathcal{D})$ can be extended by introducing additional variables and designing their probabilistic connections. Learning the parameters of these models is addressed by maximizing this function, a strategy referred to as **Maximum Likelihood** (**ML**). Depending on the nature of the observed data, different probability distributions are considered, which are described in the following subsections. Probability distributions can be primarily divided into two classes: those for discrete data and those for continuous data.

2.1.1. Probability Distributions

In mathematical terms, a **random variable** is a function that maps outcomes of a random event or experiment to numerical values. Given a sample space of possible outcomes, a random variable assigns a unique number to each outcome. The set of all possible values that the random variable can take on is called its range or *support*. The random variable can be either discrete or continuous, depending on whether its range consists of a finite or countably infinite number of values, or a continuous range of values, respectively.

Formally, a random variable $\boldsymbol{X}$ can be defined as a function $\boldsymbol{X} : \Omega \to \mathbb{R}^D$, where Ω is the sample space of the random experiment and $\mathbb{R}^D$ is the D-dimensional set of real numbers. Given an outcome ω in Ω, the value $\boldsymbol{X}(\omega)$ is the numerical value assigned by the random variable to that outcome. The distribution of a random variable is then described by its **probability mass function** (**pmf**) for discrete random variables or **probability density function** (**pdf**) for continuous random variables. The former assigns probability mass to regions in Ω, whilst the latter to each possible value of the random variable.

In the following sections, the probability distributions considered throughout the development of the doctoral thesis are briefly described.

Distributions for continuous data

Let $\boldsymbol{X}$ be a continuous random variable with range (or support) $\mathbb{R}^D$, and let $f(\boldsymbol{x})$ be a function defined for all $\boldsymbol{x} \in \mathbb{R}$ such that:

- $f(\boldsymbol{x}) \geq 0$ for all $\boldsymbol{x} \in \mathbb{R}^D$, meaning that the function is non-negative over the entire range of $\boldsymbol{X}$.
- The integral of $f(\boldsymbol{x})$ over the range $\mathbb{R}^D$ is equal to 1, i. e. $\int_{\boldsymbol{x}} p(\boldsymbol{x}) d\boldsymbol{x} = 1$, which represents the total probability of $\boldsymbol{X}$ taking a value in the range $\mathbb{R}^D$.
- For any interval $(\boldsymbol{a}, \boldsymbol{b})$ within the range $\mathbb{R}^D$, the probability that $\boldsymbol{X}$ takes on a value in the interval is given by the definite integral of $f(\boldsymbol{x})$ over the interval:

$$P(\boldsymbol{a} < \boldsymbol{X} < \boldsymbol{b}) = \int_{\boldsymbol{a}}^{\boldsymbol{b}} f(\boldsymbol{x}) d\boldsymbol{x} \tag{2.1}$$

In this way, the pdf provides a way to determine the probability that a continuous random variable takes on a value in a given interval by computing the area under the curve of the pdf over that interval.

The Gaussian distribution

The Gaussian, also referred to as Normal distribution, is a specific example of a broad class of distributions with interesting properties, called the *exponential family* (Duda et al., 1973; Bernardo and Smith, 2009), is the most widely employed mathematical function for modeling the probability of continuous variables. A vast amount of real-world phenomena can be modeled as a Gaussian distribution, including measurement errors, sensor noise, financial returns, and many other types of data. Mathematically, the multivariate Gaussian is defined as:

$$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2} \, |\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})\right) \tag{2.2}$$

The simplicity of the mathematical form of the Gaussian allows for relatively straightforward usage in all its applications, more specifically in Bayesian inference 2.1.3, for the interest of this thesis. Other interesting properties are:

- *Symmetry*: the symmetry around its mean makes it easy to understand and interpret. Additionally, the mean, median, and mode of the distribution are all equal to the same value.
- *Well-defined statistics*: which allow for the calculation of probabilities and the quantification of uncertainty.
- *Central Limit Theorem*: which states that the sum of many independent and identically distributed random variables approaches a normal distribution, regardless of the distribution of the individual variables (Giné, 1983; Liapounoff, 1900).

While there are multiple types of probability distributions for continuous data (Bishop and Nasrabadi, 2006; Murphy, 2012), this thesis will primarily use the Gaussian distribution for modeling the probability of any continuous variable, including positive continuous data. The logarithm transformation will be employed to rearrange the transformed data in the real number space.

Distributions for discrete data

Let X be a discrete random variable with possible outcomes $x_1, x_2, ..., x_K$. The probability distribution of X is a function, denoted by $P(X)$, and referred to as probability mass function, or pmf, that assigns a probability to each possible outcome of X, such that:

- $P(X = x_i) \geq 0$ for all $i = 1, 2, ..., K$, or the probabilities are non-negative.
- The sum of all probabilities is equal to 1: $\sum_{i=1}^{K} P(X = x_i) = 1$.

The Bernoulli distribution

The Bernoulli distribution is a discrete probability distribution that models the outcome of a binary experiment, where there are only two possible outcomes, such as success or failure, heads or tails, black or white pixel level, etc. It is a special case of the binomial distribution, where there is only one trial. The pmf of a Bernoulli distribution is defined as follows:

$$P(X = x) = p^x(1 - p)^{1-x} \tag{2.3}$$

where $x \in \{0, 1\}$. The Bernoulli distribution is characterized by a single parameter, p, which is the probability of success (and thus, $1 - p$ is the probability of failure) in the binary experiment. The mean of a Bernoulli random variable is given by $\mathbb{E}[X] = p$, and the variance is given by $Var[X] = p(1 - p)$. In this thesis, the Bernoulli distribution will be employed for modeling the likelihood of black/white image pixels, as well as for variables of binary nature in tabular datasets.

It is essential to note that when the Bernoulli distribution is used to model the likelihood of data, the following loss function is typically minimized with respect to the model parameters:

$$\mathbb{H}_p(x) = \mathbb{E}[-\log p(x)] = -x \cdot \log p - (1 - x) \cdot (1 - p) \tag{2.4}$$

This function is known as the **Binary Cross Entropy** and measures the uncertainty handled by the binary distribution. If the observation is positive ($x = 1$), the second term is canceled, and the first term is minimized. Conversely, if the observation is negative ($x = 0$), the first term is canceled, and the second term is minimized. In simple terms, the parameters are optimized to make the model more accurate in assigning probabilities closer to 0 (for negative observations) and 1 (for positive observations).

The Categorical Distribution

The Categorical distribution generalizes the Bernoulli to multi-class outcomes. A sample $x \in \{1, ..., K\}$ indicates a category over K possibilities. The Categorical distribution is parameterized by a vector of probabilities per category, $\boldsymbol{p}$, with $K - 1$ degrees of freedom, since the K-th element is determined as $p_K = 1 - \sum_{k=1}^{K-1} p_k$. The probability mass function of a Categorical distribution is given by

$$p(X = x) = \prod_{k=1}^{K} p_k^{[\![x=k]\!]} \tag{2.5}$$

where $[\![\cdot]\!]$ denotes the indicator function, which returns 1 when k coincides with the category of the outcome x. In this thesis, the Categorical distribution will be employed for modeling the likelihood of variables of categorical nature in tabular datasets.

2.1.2. Kullblack-Leibler divergence and Mutual Information

The dissimilarity (or *divergence*) between probability distributions $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ can be quantified with the **Kullblack-Leibler divergence** (or **KL divergence**), also denoted as **relative entropy**. For continuous variables, it is defined as

$$D_{\mathrm{KL}}(p(\boldsymbol{x})||q(\boldsymbol{x})) = \int_x p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})} d\boldsymbol{x}, \tag{2.6}$$

whilst for discrete variables, the integral gets replaced by a sum. Some remarkable properties are:

- The KL divergence is not symmetric in the two distributions, meaning that generally $D_{\mathrm{KL}}(p(\boldsymbol{x})||q(\boldsymbol{x})) \neq D_{\mathrm{KL}}(q(\boldsymbol{x})||p(\boldsymbol{x}))$. Consequently, it is not a distance metric.
- The KL divergence is always positive, $D_{\mathrm{KL}}(p(\boldsymbol{x})||q(\boldsymbol{x})) \geq 0$ with equality if and only if $p = q$.

For some distributions, the KL divergence is properly defined in closed-form. For instance, consider two D-dimensional Gaussians, $p(x) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$ and $q(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$. The expression of the KL divergence between them is:

$$D_{\mathrm{KL}}(p(\boldsymbol{x})||q(\boldsymbol{x})) = \frac{1}{2}\left[\log \frac{|\boldsymbol{\Sigma}_q|}{|\boldsymbol{\Sigma}_p|} - D + (\boldsymbol{\mu}_p - \boldsymbol{\mu}_q)^T \boldsymbol{\Sigma}_q^{-1}(\boldsymbol{\mu}_p - \boldsymbol{\mu}_q) + tr(\boldsymbol{\Sigma}_q^{-1}\boldsymbol{\Sigma}_p)\right] \tag{2.7}$$

More specifically, if $q(\boldsymbol{x})$ is the **standard Gaussian** $\mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, \boldsymbol{I})$, i.e. $\boldsymbol{\mu}_q = \boldsymbol{0}, \boldsymbol{\Sigma}_q = \boldsymbol{I}$ and $p(\boldsymbol{x})$ is a **factorized Gaussian** $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\sigma}_p^2 \boldsymbol{I})$, where $\boldsymbol{\sigma}_p^2 = [\sigma_1^2, ..., \sigma_D^2]$, then (2.7) becomes:

$$D_{\mathrm{KL}}(p(\boldsymbol{x})||q(\boldsymbol{x})) = \frac{1}{2}\sum_{d=1}^{D}\left(-\log\left(\sigma_d^2\right) - 1 + \mu_d^2 + \sigma_d^2\right), \tag{2.8}$$

as demonstrated in (Kingma and Welling, 2013). This expression will be recurrently used throughout this thesis, as it will be discussed in Chapters 3, 4 and 5.

Another important measure, when considering two random variables X and Y, that is more robust measure of statistical dependence, compared to the linear correlation coefficient, is the **mutual information**, or **MI**, which is closely related to the KL divergence,

$$\mathbb{I}(X;Y) = D_{\mathrm{KL}}(p(X,Y)||p(X)p(Y)) = \iint_{X,Y} p(\boldsymbol{x},\boldsymbol{y}) \log \frac{p(\boldsymbol{x},\boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})} d\boldsymbol{x}d\boldsymbol{y}. \tag{2.9}$$

The mutual information can be understood as the similarity between the joint distribution $p(X,Y)$ and the factored distribution $p(X)p(Y)$. In the extreme case of independency, we obtain that $p(X,Y) = p(X)p(Y)$ and so $\mathbb{I}(X;Y) = 0$.

2.1.3. Bayesian Inference

Likelihood distribution

The likelihood distribution is a function that measures the goodness of fit of a statistical model to the observed data. Given a set $\mathcal{D}$ and a statistical model with parameters θ, the likelihood distribution maps the model parameters to the probability of observing $\mathcal{D}$ for a particular set of parameters, in the form $p(\mathcal{D}|\theta)$. The likelihood can be used to perform both the **learning** task, which consists on finding the model parameters that fit

best to $\mathcal{D}$, via **maximum likelihood**, or the **inference** task: which consists on obtaining the posterior probability of the parameters θ of interest after observing $\mathcal{D}$, i.e. $p(\theta|\mathcal{D})$. It is particularly useful in situations where the data generating process is unknown and must be estimated from the observed data.

Although θ has been used in the definition as the set of parameters of interest for conditioning data, the likelihood can be also employed for conditioning the data generation on unobserved variables of a statistical model. As it will be discussed later in this thesis, and specifically in Section 2.2.1, in latent variable models, the considered *likelihood* would also be $p_\theta(\boldsymbol{x}|\boldsymbol{z})$, parameterized by θ, and the *marginal likelihood* $\int_z p_\theta(\boldsymbol{x}, \boldsymbol{z})d\boldsymbol{z}$ would be only on the parameters, after integrating the latent variables. Throughout this thesis, both definitions will be employed at different points, always trying to clearly differentiate between them.

Bayes theorem

The **Bayes theorem**, also known as Bayes rule, is a fundamental result in probability theory that provides a way to update beliefs about an event based on new evidence. It relates the *prior* probability, which measures the initial belief about an event before any new information has been taken into account, to the *posterior* probability, which refers to the updated belief after the new information has been incorporated.

Mathematically, it can be defined as follows. Let A and B be two random variables. The **posterior** distribution of A indicates how this variable is distributed after observing B, i.e. based on the evidence, and is given by

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)}. \tag{2.10}$$

The first term of the numerator $p(B|A)$ is referred to as the *likelihood*, already described in Section 2.1.3, whilst the second term of the numerator, $p(A)$, is the **prior**, which models the initial belief of the probability of A. The numerator can also be expressed as $p(A, B) = P(B|A)P(A) = P(A|B)P(B)$, which is referred to as the **joint** distribution of A and B.

Bayesian Inference

Under a probabilistic model framework, the Bayes theorem can be utilized as follows.

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)\ p(\theta)}{p(\mathcal{D})}. \tag{2.11}$$

The first term of the numerator is the *likelihood* of the data, already described in Section 2.1.3, whilst the second term of the numerator is referred to as the **prior** of the parameter, which models the initial belief of the probability of θ. The denominator is referred to as the **evidence** term, or the **marginal likelihood**, since it is the marginal distribution $p(\mathcal{D}) = \int_\theta p(\mathcal{D}, \theta)d\theta$. As a consequence, it is the normalizing constant of the posterior distribution, typically denoted by $\mathcal{Z}$ in the literature.

Although the notation θ typically refers to the parameters of a model, the Bayes theorem can be used for the inference of unknown or latent variables. Latent Variable Models (Section 2.2.1) are one of the most crucial types of generative models, and comprise a fundamental pillar of the present doctoral thesis. Within these models, latent variables

$\boldsymbol{z}$ of reduced dimension are involved in the generation of data via $p_\theta(\mathcal{D}|\boldsymbol{z})$. Hence, the posterior of the latent variables can be obtained using the Bayes theorem

$$p(\boldsymbol{z}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{z})\, p(\boldsymbol{z})}{p(\mathcal{D})}. \tag{2.12}$$

One of the key advantages of employing Bayesian inference is that it provides a coherent way of incorporating prior knowledge into the modeling process. This prior knowledge can take many forms, such as expert opinions, previous studies, domain-specific knowledge or inductive bias and it can be expressed as a probability distribution over the unknown variables.

Another advantage of Bayesian inference is that it allows for the quantification of *uncertainty* in the estimates of the model parameters or latent variables. This is achieved by computing the posterior probability distribution, which provides a range of plausible values for the parameters along with their associated probabilities. This posterior distribution can be used to make probabilistic predictions about future observations, to assess the impact of different sources of uncertainty on the model, and to perform sensitivity analyses.

Typically the numerator in (2.12) is known, i.e. the **unnormalized posterior** can be evaluated, since the likelihood and prior in $p(\mathcal{D}, \boldsymbol{z}) = p(\mathcal{D}|\boldsymbol{z})p(\boldsymbol{z})$ are parameterized. In many scenarios, to learn the parameters of a model, θ, this function needs to be integrated to obtain the named **log marginal likelihood**

$$\log p(\mathcal{D}) = \log \int_{\boldsymbol{z}} p_\theta(\mathcal{D}, \boldsymbol{z}) d\boldsymbol{z}, \tag{2.13}$$

which coincides with the normalizing constant of the posterior. In the simplest cases, these parameterizations are such that this constant (and so the posterior) is tractable. However, as it will be discussed later, more complex function approximators (e.g. deep neural networks) lead to the intractability of the posterior. Some efficient techniques allow for sampling from approximately the posterior distribution when knowing the unnormalized posterior.

Monte Carlo approximation

As discussed in the previous section, Bayesian Inference allows for uncertainty quantification by using a posterior distribution based on the evidence and prior knowledge. Although Monte Carlo methods can be applied to a wide range of problems such as integration, optimization, simulation, this thesis focuses on its application to Bayesian inference. For the case of interest, MC methods are involved in the approximation of the expectation under intractable distributions. For instance, in latent variable models, the named **posterior predictive** is given by

$$\mathbb{E}_{p(\boldsymbol{z}|\boldsymbol{x})}\left[p(\boldsymbol{x}^*|\boldsymbol{z})\right] = \int_{\boldsymbol{z}} p(\boldsymbol{x}^*|\boldsymbol{z}) p(\boldsymbol{z}|\boldsymbol{x}) d\boldsymbol{z}, \tag{2.14}$$

where the notation $\mathbb{E}_{p(\boldsymbol{z})}\left[f(\boldsymbol{x}, \boldsymbol{z})\right]$, refers to the **expectation** operator of a function $f(\boldsymbol{x}, \boldsymbol{z})$ that depends on a random variable $\boldsymbol{z}$ under a probability distribution $p(\boldsymbol{z})$.

Two problems need to be addressed for solving (2.14). First, the tractability of the expectation, which occurs only for simplistic models and thus, typically requires an approximation. Second, the posterior distribution is neither tractable in practical cases,

where likelihood distributions are parameterized by complex functions, as it will be discussed later.

Regarding the former, in general, the expectation of a function depending on a random variable $\boldsymbol{z}$, under a distribution $p(\boldsymbol{z})$ can be estimated by means of the simplest **Monte Carlo** (MC) approximation (Metropolis and Ulam, 1949)

$$\mathbb{E}_{p(\boldsymbol{z})}\left[f(\boldsymbol{z})\right] = \int_{\boldsymbol{z}} f(\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} \approx \frac{1}{S}\sum_{s=1}^{S} f(\boldsymbol{z}_s), \tag{2.15}$$

which consists on generating a large number of samples from the target distribution and use them to approximate the quantity of interest by uniformly weighting each term, i.e. by averaging. The accuracy of the approximation depends on the number and quality of samples. For complex distributions, poor samples that not accurately follow the true target lead to biased, high-variance estimations of the quantity of interest.

Further, the computation of the posterior distribution, $p(\boldsymbol{z}|\boldsymbol{x})$, in both cases where the variables to be inferred are parameters, as in (2.11), or latent variables, as here and in (2.12), requires to obtain the named *evidence* constant by marginalizing the variables of interest. For this purpose, the following integral needs to be resolved

$$\mathcal{Z} = p(\boldsymbol{x}) = \int_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z})d\boldsymbol{z}, \tag{2.16}$$

which in some cases can be extremely hard to solve or actually computationally or mathematically intractable, and consequently, posterior evaluation is not tractable.

Considering this two problematics, the basic idea presented in (2.15) is not generally sufficient for performing Bayesian Inference. More advanced MC methods aim at obtaining these high-quality samples when complex distributions are to be sampled from. For instance, in **importance sampling**, the variance of the estimation is reduced by using samples from a second distribution, $q(\boldsymbol{z})$, referred to as *proposal*, and measure their *importance* by computing $\frac{p(\boldsymbol{x},\boldsymbol{z})}{q(\boldsymbol{z}|\boldsymbol{x})}$. The resulting approximation is

$$\mathbb{E}_{p(\boldsymbol{z}|\boldsymbol{x})}\left[p(\boldsymbol{x}^*|\boldsymbol{z})\right] \approx \sum_{s=1}^{S} \tilde{w}_s \; p(\boldsymbol{x}^*|\boldsymbol{z}_s) \tag{2.17}$$

where $w_s = \frac{p(\boldsymbol{x},\boldsymbol{z})}{q(\boldsymbol{z}|\boldsymbol{x})}$ and $\tilde{w}_s = \frac{w_s}{\sum_{s=1}^{S} w_s}$. This idea, as it will be discussed in Section 3.5 is exploited in the IWAE model (Burda et al., 2015) for improving the lower-bounded estimation of the log-likelihood of the model.

Markov Chain Monte Carlo

The expectation is computed by summing the integrand in (2.15) for infinitesimal volumes $d\boldsymbol{z}$ following $p(\boldsymbol{z})$, and approximated by discretizing the parameter space using samples. As discussed in (Betancourt and Girolami, 2015), for the regions around the mode, whilst the density is larger, the volume contribution is low. On the contrary, regions far from the probability mass have lower contributions on the density, but their volume is higher. Further, this problematic increases with the dimension of the target space. In MC approximations, samples from these two regions can lead to poor approximations. However, there is a specific region that dominates the expectations, named the *typical set* (Betancourt, 2017), and Monte Carlo MC methods focus on approximating expectations by using samples from the typical set.

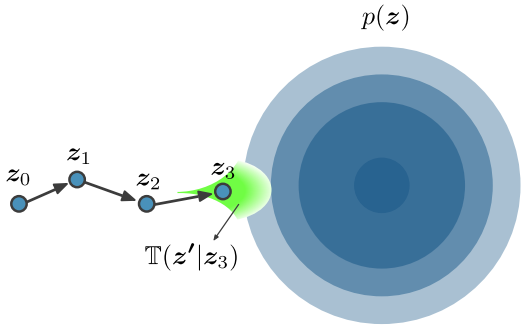


Figure 2.3: Illustrative example of MCMC for sampling from target $p(\boldsymbol{z})$.

In **Markov chain Monte Carlo** (**MCMC**), a Markov chain is employed for exploring the typical set that dominates the expectation estimations, and thus, it allows for more accurate approximations. An illustrative example is provided in Figure 2.3. A properly designed Markov chain can progress through the parameter space by sequentially accepting/rejecting new states proposed by a random map, denoted as Markov transition. Mathematically defined as $\mathbb{T}(\boldsymbol{z}'|\boldsymbol{z})$, it defines the probability distribution of the next state $\boldsymbol{z}$, given the current state $\boldsymbol{z}$, and it preserves the target distribution, meaning

$$p(\boldsymbol{z}) = \int_{\boldsymbol{z}} p(\boldsymbol{z}')\mathbb{T}(\boldsymbol{z}|\boldsymbol{z}')d\boldsymbol{z}'. \tag{2.18}$$

In broader terms, applying the transition over a set of samples from $p(\boldsymbol{z})$ would lead to a new set of samples that follows the same distribution $p(\boldsymbol{z})$.

Of remarkable consideration among the MCMC variants is **Hamiltonian Monte Carlo** (**HMC**) (Neal et al., 2011; Betancourt and Girolami, 2015), due to its efficiency in exploring high-dimensional targets by exploiting the geometry of the typical set, that leads to an improved computational efficiency when compared with the alternatives and proved improvement of the estimator. *This technique will be extended in Chapter 5 of this thesis, when presenting the second principal contribution (Peis et al., 2022), where a careful design allows for using HMC in combination with powerful probabilistic models such as Hierarchical Variational Autoencoders.*

2.2. Generative Models

This section introduces the concept of generative modeling, The model illustrated in Figure 2.2c can also be referred to as a **generative model**. The reason is that, once properly trained, the learned distributions can be used for sampling new synthetic data, that ideally would look like the observed real data. Specifically, from $p(\boldsymbol{x}, \boldsymbol{y})$, $p(\boldsymbol{x})$ can be used for getting new data $\boldsymbol{x}^*$, whilst $p(\boldsymbol{y}|\boldsymbol{x}^*)$ can be used for generating the label for each new datapoint. Although a large list of potential benefits of using this generative approach could be listed, some of the most important ones are:

- It allows for detecting **outliers**, i.e. points that does not follow the true manifold of the data. The black cross in Figure 2.2 represents this concept. Observed datapoints where $p(\boldsymbol{x}^*)$ is low can be detected for requesting their labels or regenerating them. Also called anomaly detection, the detection of datapoints with low probability

is relevant to several usecases, like network intrusion detection, medical diagnosis, fraud detection or manufacturing defect detection, among others.

- In the context of **decision making**, it can properly weight a decision. In the binary classification example, instead of using a hard decision of positive/negative, regardless of how far a test point is from the decision boundary, a generative model accounts for the distance to the boundary and in general, the *likelihood* under the learned distributions. This can be of vital importance in critical decisions such as, for example, determining whether a patient is at high risk of developing a certain disease, or assessing the risk of an investment, taking into account the uncertainty of the future market conditions.

- The **uncertainty** captured by the model can be employed for assessing real uncertainty about the environment. To give an example: detecting consecutive outliers in a sensing system could be a potential indicator of sensor failure.

- Generative models can be used to perform **data augmentation**, i.e. artificially expand the size and diversity of datasets by generating synthetic examples that mimic the existing data. This can help to improve the accuracy and robustness of machine learning models, overcoming limitations such as data scarcity and imbalanced datasets. By generating synthetic data, generative AI models can help to improve decision making and predictions.

2.2.1. Latent Variable Models

In **Latent Variable Models** (**LVMs**), complex distributions at the observation level are captured by simpler conditional distributions at the latent level, which usually have lower dimensionality. The latent space has the potential to capture the real underlying processes that generated the observed data. As an illustration, in Figure 2.1, the handwritten MNIST images may have a latent code that represents the digit's shape, width, orientation, etc., and thus, a trained model would distribute the latent space based on these factors. This key feature of latent variable models allows for potential interpretability of their latent space when compressing generative factors. This relationship between the latent properties and generative factors is commonly known as **disentanglement**, which will be further discussed in Section 3.8.1 of this thesis.

In general, when defining a latent variable model, one of the final objectives is to train it properly for getting meaningful posteriors over observations, whilst accurately reconstructing observed data. In mathematical terms, given a datapoint $\boldsymbol{x}$, the posterior $p(\boldsymbol{z}|\boldsymbol{x})$ can be obtained using the Bayes theorem (2.12) and ideally would be informative of the characteristics of $\boldsymbol{x}$. The marginal likelihood distribution is obtained by computing the expectation $\mathbb{E}_{p(\boldsymbol{z}|\boldsymbol{x})}(p(\boldsymbol{x}|\boldsymbol{z}))$.

The nature of the latent variables can be of different types. On the one hand, discrete latent variables can account for multiple modalities in the data, as in the case of Mixture Models, which will be described below. On the second hand, continuous latent variables can model the mentioned underlying factors being compressed into a reduced latent space, being Probabilistic Principal Component Analysis the simplest approach.

Mixture Models

In **Mixture Models**, a set of K components is modeled by a Categorical latent variable $p(z) = \text{Cat}(\boldsymbol{\pi})$ (Section 2.1.1), where the discrete parameter space is $z \in \{1, ..., K\}$.,

and $\boldsymbol{\pi}$ is the learnable vector of probabilities per component, being $\pi_k = p(z = k)$. With probability π_k, a generated point would follow the k-th distribution $p(\boldsymbol{x}|\boldsymbol{\theta}_k)$, where $\boldsymbol{\theta}_k$ is the set of parameters that define the corresponding likelihood distribution. The joint probability of the model is given by

$$p(\boldsymbol{x}, z) = \prod_{k=1}^{K} \left(\pi_k \; p(\boldsymbol{x}|\boldsymbol{\theta}_k)\right)^{[\![z=k]\!]} \tag{2.19}$$

Note that to evaluate the joint distribution, z requires to be known. Nevertheless, it is defined as a latent variable (unobserved), and due to this reason, the marginal likelihood can be obtained by integrating it, which in this case, since z is discrete, is equivalent to summing over the parameter space

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \; p(\boldsymbol{x}|\boldsymbol{\theta}_k). \tag{2.20}$$

Another quantity of interest, as mentioned in the previous subsection, is the posterior over the latent variables $p(z|\boldsymbol{x})$. Using the Bayes theorem, it can be expressed as

$$p(z = k|\boldsymbol{x}) = \frac{\pi_k \; p(\boldsymbol{x}|\theta_k)}{\sum_{k=1}^{K} \pi_k \; p(\boldsymbol{x}|\theta_k)}. \tag{2.21}$$

each of the posterior probabilities,

To provide with an illustrative example, in Figure 2.4 a **Gaussian Mixture Model** (**GMM**) is depicted, where $K = 3$ and each of the components is a Gaussian distribution, i.e. $\boldsymbol{\theta}_k = \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$ and thus $p(\boldsymbol{x}|\boldsymbol{\theta}_k) = \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$. The probability mass assigned to the k-th Gaussian component is modulated by the prior probability π_k. In the example, $\boldsymbol{\pi} = [0.4, 0.3, 0.3]$.

Given a dataset $\mathcal{D} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$, the optimal parameters of the mixture, namely $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$, would be those that maximize the log marginal likelihood of the observed data

$$\log p(\mathcal{D}; \theta) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} p(\boldsymbol{x}_n|z_k) \right). \tag{2.22}$$

By setting the derivatives of (2.22) to zero with respect to each parameter, it is obtained

$$\begin{aligned} \boldsymbol{\mu}_k &= \frac{1}{N_k} \sum_{n=1}^{N} p(z_n = k|\boldsymbol{x}) \boldsymbol{x}_n, \\ \boldsymbol{\Sigma}_k &= \frac{1}{N_k} \sum_{n=1}^{N} p(z_n = k|\boldsymbol{x}) (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T \end{aligned} \tag{2.23}$$

which, as it can be appreciated, does not constitute a closed-form solution, since the optimal parameters depend on the posteriors $p(z_k|\boldsymbol{x})$, that also depends on the parameters themselves. To solve this issue, an elegant method named **Expectation-Maximization** (**EM**) is typically employed for performing the following two steps in an iterative manner:

1. The **E-step**, where the posterior probabilities per datapoint and component are obtained as $p(z_k^{(n)}|\boldsymbol{x}^{(n)})$ fixing θ.

2. The **M-step**, where fixing $p(z_k^{(n)}|\boldsymbol{x}^{(n)})$, the parameters θ are updated using (2.23).

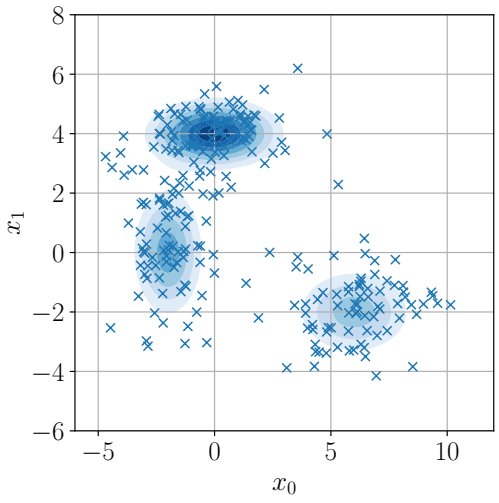


Figure 2.4: A Gaussian Mixture Model of $K = 3$ components, with $\boldsymbol{\pi} = [0.4, 0.3, 0.3]$.

The log-likelihood is guaranteed to increase within each iteration of the algorithm. The EM method can also be adapted to a wide spectrum of latent variable models, where in the E-step the posterior over latents is obtained, and in the M-step the parameters are updated. In this thesis, a mixture model will be employed in Chapter 4 for increasing the flexibility of the latent space. Nevertheless, due to the usage of more complex probabilistic connections (via neural networks), the optimization of its parameters will not follow the EM algorithm, but other methods such like amortized variational inference that will be extensively discussed in the corresponding sections.

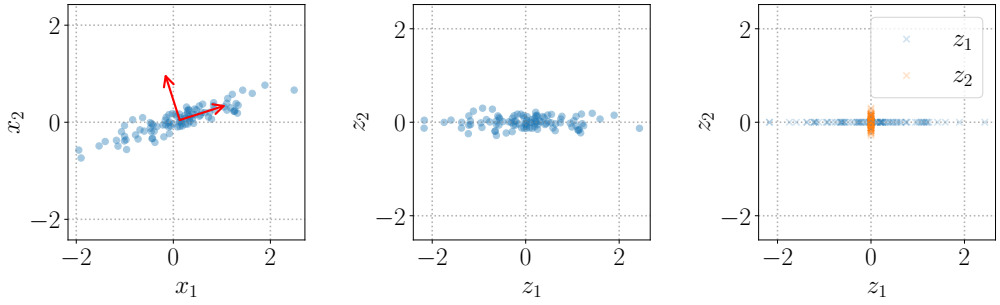
Probabilistic Principal Component Analysis

Principal Component Analysis (**PCA**) is one of the most widely employed techniques for dimensionality reduction, lossy data compression, feature extraction and data visualization. It is based on the *Karhunen-Loève* transform, and consists on learning the (linear) orthogonal projection of the data onto a lower dimensional latent space that leads to the minimum reconstruction error, measured as the mean squared error between the data and the projections, or equivalently, the maximum variance in the projected space.

Mathematically, PCA can be defined as follows. Given a dataset $\boldsymbol{X} = \{\boldsymbol{x}_1^T, ..., \boldsymbol{x}_N^T\}$ with points $\boldsymbol{x}_n \in \mathbb{R}^D$, represented with blue dots in Figure 2.5a, the PCA solution is the matrix $\boldsymbol{U}$ with dimensions $D \times M$, whose columns are the orthonormal vectors $\{\boldsymbol{u}_1, ..., \boldsymbol{u}_M\}$, represented with red arrows in Figure 2.5a, such the transformed data, $\boldsymbol{Z} = \{\boldsymbol{z}_1^T, ..., \boldsymbol{z}_N^T\}$ with coordinates $\boldsymbol{z}_n = [\boldsymbol{u}_1^T \boldsymbol{x}_n, ..., \boldsymbol{u}_M^T \boldsymbol{x}_n]$ (or expressed in matrix notation, $\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{U}$), exhibits the maximum variance, as depicted in Figure 2.5b. It can be demonstrated (Bishop and Nasrabadi, 2006; Murphy, 2012) that these optimal vectors $\boldsymbol{u}_m$ that maximize the variance and minimize the reconstruction error of the transformed data are the eigenvectors of the empirical covariance matrix, $\boldsymbol{S}$, defined as

$$\boldsymbol{S} = \frac{1}{N} \sum_{i=1}^{N} (\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^T, \tag{2.24}$$

where $\bar{\boldsymbol{x}}$ is the empirical mean given by $\bar{\boldsymbol{x}} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}_n$. The order of the PCA projection



(a) Data space (b) Transformed space (c) Projections

Figure 2.5: PCA illustrative example. A bivariate synthetic data (blue dots) is depicted in (a). PCA principal components $\boldsymbol{u}_1, \boldsymbol{u}_3$ are plotted as red vectors. In (b), tranformed points $\boldsymbol{z}$ are plotted. In (c), projections z_1, z_2 are plotted separately using different colors.

vectors is given by the bigger eigenvalues of the same matrix, that also quantify for the *explained variance*, i.e. the amount of variance that is accounted within each projection.

PCA can be also expressed in terms of a probabilistic latent variable model. In such definition, named **Probabilistic PCA** (Tipping and Bishop, 1999). Instead of considering a linear deterministic compression, the data is linearly compressed into a parameterized Gaussian latent space (posterior), with standard prior $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Data is generated by decompressing from the latent space, using a linear transformation for obtaining the Gaussian likelihood parameters

$$p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu}, \sigma^2\boldsymbol{I}). \tag{2.25}$$

where the mean is given by the linear operation $\boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu}$, governed by the matrix $\boldsymbol{W}$ of dimensions $D \times M$. The choice of the variance, $\sigma^2\boldsymbol{I}$ makes the dimensions of $\boldsymbol{x}$ to be factorized. The role of $\boldsymbol{W}$ can be compared to the named $\boldsymbol{U}$ in the non-probabilistic version of PCA, since they define a linear data subspace. The goal is to learn $\boldsymbol{W}$, $\boldsymbol{\mu}$ and σ^2. Once the parameters $\boldsymbol{W}, \boldsymbol{\mu}$ and σ^2 are obtained for a dataset, new similar data can be simply generated by using $\boldsymbol{x} = \boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is a sample from Gaussian noise, i.e. with zero mean and covariance $\sigma^2\boldsymbol{I}$.

To find the optimal parameters, a Maximum Likelihood strategy can be derived for Probabilistic PCA. The data log likelihood follows

$$\log p(\boldsymbol{X}; \boldsymbol{W}, \boldsymbol{\mu}, \sigma^2) = \sum_{n=1}^{N} \log \mathcal{N}(\boldsymbol{x}_n; \boldsymbol{\mu}, \boldsymbol{C}), \tag{2.26}$$

where the parameters $\boldsymbol{\mu}, \boldsymbol{C}$ of the distribution $p(\boldsymbol{x}_n) = \int_{\boldsymbol{z}} p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} = \mathcal{N}(\boldsymbol{x}_n; \boldsymbol{\mu}, \boldsymbol{C})$ can be obtained in closed form, thanks to the simplicity of the linear operations. As it will be discussed later, more complex transformations (i.e. given by neural networks) will make this integral intractable, and will lead to the impossibility of obtaining the data likelihood in close form. The covariance matrix $\boldsymbol{C}$ is given by $\boldsymbol{C} = \boldsymbol{W}\boldsymbol{W}^T + \sigma^2\boldsymbol{I}$. It can be demonstrated (full derivation can be read in Bishop and Nasrabadi (2006)) that the

ML solution is given by

$$\boldsymbol{\mu}_{ML} = \bar{\boldsymbol{x}} = \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x}_n \tag{2.27}$$

$$\sigma^2_{ML} = \frac{1}{D-M}\sum_{i=M+1}^{D}\lambda_i \tag{2.28}$$

$$\boldsymbol{W}_{ML} = \boldsymbol{U}_M(\boldsymbol{L}_M - \sigma^2\boldsymbol{I})^{1/2}\boldsymbol{R}, \tag{2.29}$$

where λ_i are the eigenvalues, making σ^2_{ML} is the average variance of the discarded dimensions. The matrix $\boldsymbol{U}_M$ with dimensions $D \times M$ has at columns the subset of M eigenvectors of the data covariance matrix $\boldsymbol{S}$. The diagonal matrix $\boldsymbol{L}_M$ contains the corresponding eigenvalues λ_i in its diagonal, and $\boldsymbol{R}$ is any rotation (orthogonal) matrix, leading to rotation invariance in the compressed space. It its simplest form, $\boldsymbol{R} = \boldsymbol{I}$.

Although the ML solution can be computed in closed-form, the EM algorithm presented in Section 2.2.1 can be adapted to PCA and its variants when spaces of high-dimensionality increase the computational cost of the ML solution (mainly matrix inversion and eigendecomposition). In the E-step, the posterior parameters are obtained for the corresponding parameters state. In the M-step, the posterior parameters are used for updating the learnable parameters.

Thanks to the design of the model, by using the Bayes theorem, the posterior distribution $p(\boldsymbol{z}|\boldsymbol{x})$ can be derived (using 2.116 in Bishop and Nasrabadi (2006)) as

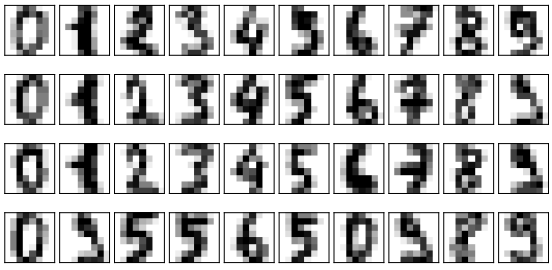
$$p(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z};\ \boldsymbol{M}^{-1}\boldsymbol{W}^T(\boldsymbol{x}-\boldsymbol{\mu}),\ \sigma^2\boldsymbol{M}^{-1}). \tag{2.30}$$

Having learned the parameters, the posterior mean depends on the datapoint $\boldsymbol{x}$ (the posterior covariance is independent). By graphically representing the two first components of the posterior mean, meaningful interpretations can be obtained. In Figure 2.6 a simple illustrative example is depicted, where different images representing the same digit are projected into close regions of the latent space, separated from regions for other digits.

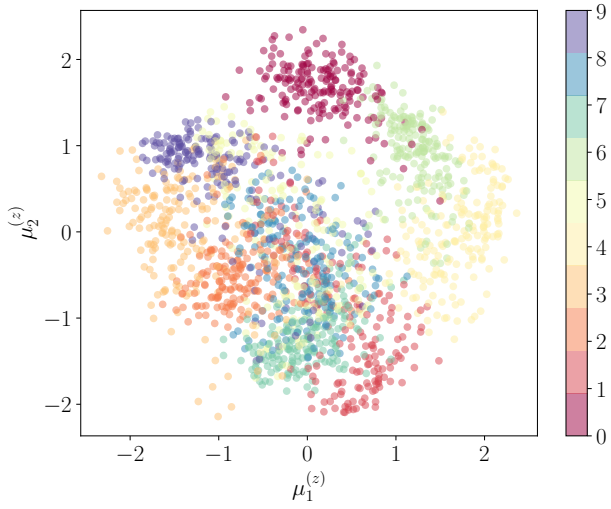
2.3. Deep Learning

In the previous sections of this thesis, the complexity of the functions involved in parameterizing probability densities has been mentioned several times. In general, the complexity can be extended to any type of function that capture essential information from the data to perform any desired task. This complexity is referred to as the ability to extract useful information from the data. Nevertheless, it is of vital importance that the data itself contains useful information. In other terms, designing the appropriate set of features to tackle a Machine Learning task is a common way to solve it. For instance, obtaining the frequency spectrum of an audio signal via the Fourier transform is an useful set of features in identifying speaker gender in sound recordings. However, it can be challenging to determine the necessary features for many tasks, such as recognizing cars in photographs, where it is difficult to define the characteristics of an object, such as wheels, in pixel terms. Shadows, glare, and obstructions can further complicate matters.

One solution is **representation learning**, which uses machine learning to identify the mapping between representation and output, as well as the representation itself. This approach has several advantages, including better performance and rapid adaptation to new tasks. Manually designing features for a complex task requires a significant amount



(a) Data space



(b) Latent space

Figure 2.6: Probabilistic PCA illustrative example. In (a) 8×8 image samples are showed from a training set of digitis ($\boldsymbol{x} \in \mathbb{R}^{64}$). In (b), the bidimensional latent space $\boldsymbol{z} \in \mathbb{R}^2$ of Probabilistic PCA is depicted by plotting the mean of the posterior of (2.30).

of time and effort, sometimes spanning decades. Representation learning algorithms can find appropriate features for simple tasks in minutes or complex tasks in hours, with minimal human intervention.

When designing features or algorithms for learning features, our goal is usually to separate the **factors of variation** that explain the observed data. Such factors are often not quantities that are directly observed. Instead, they may exist either as unobserved objects or unobserved forces (*latent*) in the physical world that affect observable quantities. The goal when designing features or algorithms for learning features is to separate the factors of variation that explain the observed data. These factors refer to separate sources of influence, and they can be objects or forces in the physical world or constructs in the human mind that provide useful explanations or inferred causes. They can help make sense of the data's rich variability. However, many of these factors influence every piece of data we observe, making it difficult to disentangle them and focus on what is essential.

Identifying high-level, abstract features from raw data can be challenging, and some

factors require sophisticated, nearly human-level comprehension. When obtaining a representation is almost as difficult as solving the original problem, representation learning may not seem helpful. However, **deep learning** solves this issue by introducing abstractions expressed in simpler representations, allowing the computer to construct complex concepts automatically.

In Figure 2.7 an illustrative example is provided. At the right part of the image, a dataset is showed, composed by two radial groups of datapoints with binary target. Thus, binary classification is considered. If a linear model (logistic regression) of type $p(y = 1) = \sigma\left(w_0 + w_1 x_1 + w_2 x_2\right)$ was to be learned from the data, being $p(y = 1)$ the positive class probability and σ the sigmoid function, it would be not possible to find any line that defined a decision boundary for separating the two classes. Additional feature engineering would be required. For example, by transforming the original data to be expressed in polar coordinates, namely, computing the radius by $r = \|\boldsymbol{x}\|$, and the angle $a = \arctan(\frac{x_2}{x_1})$, each datapoint $[x_1, x_2]$ would be transformed to $[r, a]$. Furthermore, even by using only the radius r accurate classifications would be performed. On the contrary, the Multi-Layer Perceptron model depicted in Figure 2.7 automatically performs the representation learning when fed with the $[x_1, x_2]$ coordinates. By stacking three hidden layers of $[4, 4, 2]$ neurons respectively, each of them activated by simple linear operations on the previous layer, followed by a non-linear activation, a test error of 0.0 is achieved after a few optimization steps.

A more detailed discussion on Multi-Layer Perceptron and more advanced deep learning architectures is provided in the following subsections.

2.3.1. Multi-Layer Perceptrons

Multilayer Perceptrons (**MLPs**), also referred to as deep feedforward neural networks, are the pillar of deep learning models. Their goal is to approximate some function f^* for mapping inputs to outputs, following $\boldsymbol{y} = f^*(\boldsymbol{x})$. For instance, revisiting the example from Figure 2.7, the function to approximate the mapping from bidimensional inputs to a binary target.

To do so, MLPs stack a set of L **hidden layers** in a chain. The length of this chain defines the **depth** of the network, being the last layer typically referred to as **output layer**. If $L = 3$, the learned function that approximates the unknown f^* is $f^{(3)}(f^{(2)}(f^{(1)}(\boldsymbol{x})))$. The behavior of the hidden layers, which do not directly receive desired output from the training data, must be determined by the learning algorithm to achieve the accurate approximation of f^*. The output of each hidden layer l of the network is a vector-valued, denoted as $\boldsymbol{h}^{(l)}$, composed by a set of **hidden units**, and the dimensionality of these hidden layers determines the width of the model. Each element of the vector may be interpreted as playing a role analogous to a neuron, and the layer can be thought of as consisting of many units that act in parallel. The idea of using many layers of vector-valued representation is drawn from neuroscience. The functions used to compute these representations are loosely guided by neuroscientific observations about the functions that biological neurons compute. When MLPs incorporate feedback connections, they are referred to as recurrent neural networks (Section 2.3.3).

The function applied at each layer is typically given by

$$\boldsymbol{h}^{(l)} = f^{(l)}(\boldsymbol{h}^{(l-1)}) = g(\boldsymbol{W}^{(l)}\boldsymbol{h}^{(l-1)} + \boldsymbol{b}^{(l)}), \tag{2.31}$$

where g applies for any non-linear function with some required properties, typically referred to as **activation function**. Widely employed activation functions include the

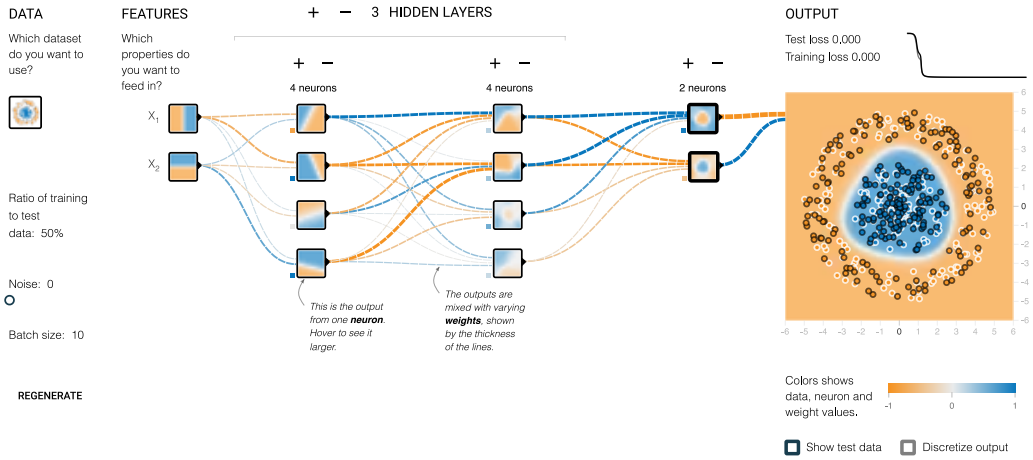


Figure 2.7: Multi-Layer Perceptron example. An MLP of 3 hidden layers with $[4, 4, 2]$ units, respectively, using hyperbolic tangents as activation, is trained to classify data $\boldsymbol{x} \in \mathbb{R}^2$ into two categories, $y \in \{-1, 1\}$. The colormap shows the function $p(y|\boldsymbol{x}) = f(\boldsymbol{x})$ model individually by each unit and the final predictive distribution $p(y|\boldsymbol{x})$. Example obtained from https://playground.tensorflow.org/.

Hyperbolic Tangent, or more advanced techniques such as the Rectified Linear Units (ReLU, Nair and Hinton (2010)). In other words, each hidden layer is obtained by, first, using a linear transformation of the previous layer, and second, applying a non-linearity in order to represent non-linear functions of the input. In (2.31), $\boldsymbol{W}^{(l)}$ and $\boldsymbol{b}^{(l)}$ represent, respectively, the weight matrix and the biases for the l-th linear transformation. Thus, the set of parameters to be learned during training a MLP are $\theta = \{\boldsymbol{W}_{l=1}^{L}, \boldsymbol{b}_{l=1}^{L}\}$.

2.3.2. Convolutional Neural Networks

When the data to extract information from is of grid-like structure, **Convolutional Neural Networks** (**CNNs**, LeCun et al. (1989)), are the neural architecture that excels. This data type includes time-series data, which can be thought of as a 1D grid sampled at regular (or irregular) time intervals, and image data, which can be represented as a 2D grid of pixels. An example of a CNN architecture (the well-known VGG16 network from Simonyan and Zisserman (2014)) is included in figure 2.8. CNNs are based on a mathematical operation known as **convolution**, which is a linear operation that involves sliding a **filter** (also referred to as kernel) over the input data to produce a feature map. For instance, the output of a 2D convolution over a discretized grid is

$$Y(i,j) = (X * K)(i,j) = \sum_{w=1}^{W} \sum_{h=1}^{H} X(i+w, j+h) K(w,h), \tag{2.32}$$

where (W, H) are the width and height parameters that define the kernel size. This operation is used in place of the general linear transformation discussed for MLPs in Section 2.3.1. To fully describe a CNN, additional parameters involved in the convolution operation need to be considered. The **padding**, defined by sizes (p_i, p_j), adds a frame to the borders of the input grid, being typically zero-valued, in order to avoid shrinking the input size. The **stride** parameter defines the resolution of the indices i, j, or the shift

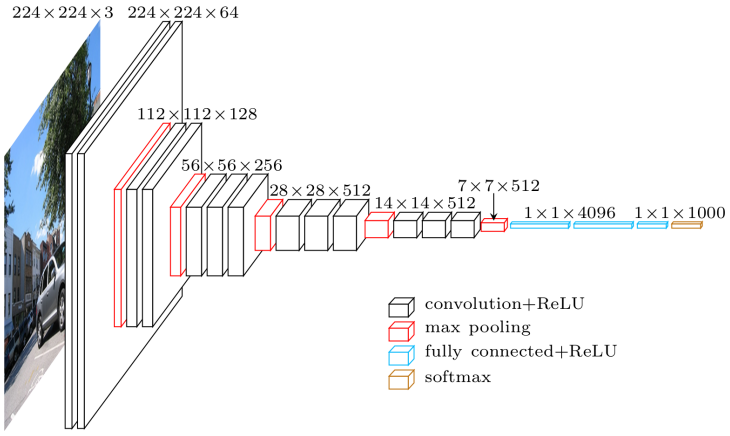


Figure 2.8: Convolutional Neural Network example. The VGG16 network architecture (Simonyan and Zisserman, 2014).

centers to apply the kernels. In other words, by choosing a stride of (s_i, s_j), the indices would be $i' = s_i \times i$ and $j' = s_j \times j$.

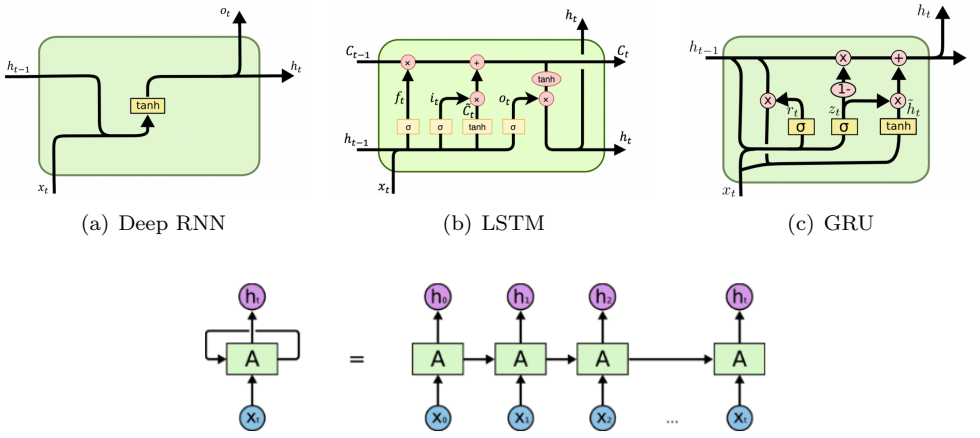
Stacked to the convolutional operation, a second stage with a non-linear activation is applied, with the same motivation than in MLPs. In a third stage, typically a **pooling** operation is included, that replaces the convolutional outputs at each location by a statistical summary of their neighbors. **Max pooling** or **average pooling** are typical choices, where the latter outputs the average of the selected neighbors, and the former outputs the maximum. Thanks to this last operation, a strong prior is added to the task of learning patterns from grid-type data, in the sense that those learned patterns must be invariant to small translations. These three stacked operations define a hidden layer in CNNs.

For MLPs, as discussed in Section 2.3.1, the hidden units are the outputs of each simple operation on the previous set of hidden units performed within each layer. Grouping the units in a vector gives $\boldsymbol{h}^{(l)} = f^{(l)}(f^{(l-1)}(...f^{(1)}(\boldsymbol{x})))$. Equivalently, in the context of CNNs, the hidden units are typically referred to as **output channels**. To learn each channel, a parallel kernel (applied over the same input) with independent learnable weights is defined. Thus, the group of parallel convolutional operations outputs a set of images that can be stacked, typically referred to as the hidden layer or representation.

CNNs have proven to be highly effective in real-world applications when spatial or temporal patterns are to be captured for performing downstream tasks. To list some examples, (Krizhevsky et al., 2017; Simonyan and Zisserman, 2014; Szegedy et al., 2015; He et al., 2016a) are successful models in image classification, (Zhao et al., 2017; Pelletier et al., 2019) achieve astonishing results using CNNs for time series, and (Goodfellow et al., 2020; Radford et al., 2015; Van den Oord et al., 2016; Salimans et al., 2017) build highly expressive generative models based on CNNs (this last example will be extended in Section 2.4 below).

2.3.3. Deep Recurrent Neural Networks

Deep Recurrent Neural Networks (**RNNs**) are a type of neural network designed for modeling sequential data. These networks are better designed to handle sequences of values, similarly to how CNNs handle grids of values such as images. They present a



(a) Deep RNN (b) LSTM (c) GRU

(d) A general diagram for RNNs

Figure 2.9: Examples of Recurrent Neural Neworks. In (a), (b) and (c), the most extended architectures for the function to be applied at each state are represented. In (d), 'rolled' and 'unrolled' version for sequences is depicted, being the green boxes one of the architectures in (a), (b) and (c).

parallel advantage when compared to CNNs: they easily scale to sequences of variable length, and further, much longer sequences than regular networks that are not sequence-based. The way they achieve this is by sharing parameters.

The idea of sharing parameters across different parts of the data or the model can be placed back to the machine learning research of the 1980s (Rumelhart et al., 1985). It allowed to build models that learn from observations of variable size, i.e. sequences with different lengths. Parameter sharing enables to generalize across positions in time, which would not be possible if separated parameters were applied to different time indices. i.e. if MLPs where to be used for learning from sequences. This sharing of statistical strength is especially crucial when specific information can occur at multiple positions within the sequence and allows to avoid overfitting to the training set and to generalize to new examples. The origins of parameter sharing for discovering termporal patterns can be placed to the 1980s, when several variants were studied, being Elman networks (Elman, 1990), Jordan networks (Jordan, 1990), echo state networks (Jaeger, 2001) and time-delay neural networks (Lang, 1988; Waibel et al., 1989; Lang et al., 1990) the most importante ones.

CNNs would apply for parameter sharing in temporal sequences if their 1D version is considered. Nevertheless, the field is restricted to learn only from a set of neighbors at each step. RNNs operate in a different manner, more adequate to sequential data. The output of each hidden function directly depends on all the previous outputs (Figure 2.9d), and is produced by the same update rule

$$\boldsymbol{h}^{(t)} = f_{\boldsymbol{\theta}}(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)}), \tag{2.33}$$

thus sharing the parameters. The unfolded function at each state can be expressed as

$$\boldsymbol{h}^{(t)} = g_{\boldsymbol{\theta}}^{(t)}(\boldsymbol{x}^{(t)}, \boldsymbol{x}^{(t-1)}, \boldsymbol{x}^{(t-2)}, ..., , \boldsymbol{x}^{(1)}). \tag{2.34}$$

More specifically, the recurrent functions to be applied at each state are in the form of

the presented ones for MLPs and CNNs, i.e.

$$\boldsymbol{h}^{(t)} = \sigma(\boldsymbol{x}^{(t)}\boldsymbol{W}_{ih}^{T} + \boldsymbol{b}_{ih} + \boldsymbol{h}^{(t-1)}\boldsymbol{W}_{hh}^{T} + \boldsymbol{b}_{hh}) \tag{2.35}$$

where σ is again a non-linear activation, typically *tanh* or *ReLU* (Nair and Hinton, 2010).

This approach allows for learning recurrent patterns from considerably deeper sequences than with a CNN approach. Thus, RNNs stand out in applications where sequences are of high length, e.g. Electronical Health Records (EHR).

In a paper published in the initial stage of this thesis, (Peis et al., 2019), RNNs were employed to learn hidden representations from two asynchronous sources of sequential data from patients: EHR data from the hospital, and personalised data (Ecological Momentary Assessment, EMA) collected via a smartphone application. These hidden representations we employed as inputs to a classifier for predicting suicidal ideation probability in a cohort of psychiatric patients.

More advanced to this presented initial setting for deep RNNs are the named gated RNNs, being the most extended ones the **Long Short-Term Memory networks** (**LSTMs**, Hochreiter and Schmidhuber (1997)) and the **Gated Recurrent Unit**s (**GRUs**, Chung et al. (2014)). They stand due to solving one of the main issues associated to training vanilla RNNs. As it will be discussed in the next subsection 2.3.5, the gradients of the loss to be minimized with respect to the parameters of the model, are computed by creating backwards paths and using the chain rule. When these temporal paths are too deep, the propagated derivatives might vanish or explode, ending in the two problems referred to as **vanishing gradients** and **exploding gradients**, respectively.

RNNs and its advanced versions have been successfully applied to neural translation via sequence-to-sequence (Sutskever et al., 2014; Bahdanau et al., 2014), text or handwriting generation (Graves, 2013), speech recognition (Graves et al., 2013; Graves and Jaitly, 2014) or audio generation (Oord et al., 2016), to name a few examples.

2.3.4. Advanced architectures

Below, three of the architectures that have resulted the most relevant in the recent literature are briefly described.

Transformers

Transformer Networks are a type of sequential neural architecture that have revolutionized the field of natural language processing (NLP) and other sequence related fields. They were firstly introduced by Vaswani et al. (2017), and have since become the backbone of striking state-of-the-art generative models such as BERT (Devlin et al., 2018) or GPT (Radford et al., 2018) and its continuations (Radford et al., 2019; Brown et al., 2020; OpenAI, 2023), AlphaFold (Jumper et al., 2021) or Jukebox (Dhariwal et al., 2020), to list some examples. The key innovation of Transformer Networks is their ability to capture long-range dependencies in text by using self-attention mechanisms. Self-attention allows the model to weigh the importance of different words in a sentence when making predictions, which is particularly useful for tasks such as language translation and question answering.

The architecture of a Transformer Network is composed of an encoder and a decoder, each consisting of multiple layers of self-attention and feedforward neural networks. During training, the model learns to encode the input text into a sequence of vectors that preserve the semantic meaning of the words in the text. These encoded vectors are then used by the decoder to generate the output text. One of the advantages of Transformer

Networks is that they are parallelizable, making them computationally efficient for training on large datasets. Additionally, they have been shown to be effective at transfer learning, where a model trained on one task can be fine-tuned for a different task with minimal additional training.

Overall, Transformer Networks represent a major advancement in sequence modeling and have opened up new research directions in the field.

ResNets

Residual Networks (ResNets, He et al. (2016a)) are a type of deep neural network architecture that were introduced to address the vanishing gradient problem in very deep networks. The vanishing gradient problem arises when the gradient of the loss function with respect to the parameters becomes very small, making it difficult to update the weights in the earlier layers of the network.

In ResNets, a residual block is introduced that allows for shortcut connections to bypass some layers in the network. These shortcut connections add the output of a previous layer to the output of a later layer, effectively allowing the network to learn residual functions. By using these shortcut connections, ResNets are able to train very deep neural networks with hundreds of layers, while still achieving state-of-the-art performance on a variety of image recognition tasks.

ResNets have been used in many applications, such as image generation and classification (He et al., 2016a;b; Xie et al., 2017; Chen et al., 2017b; Child, 2020), object detection (Ren et al., 2015), and semantic segmentation (Ronneberger et al., 2015), and have set the standard for deep neural network architectures.

HyperNetworks

Hypernetworks (Ha et al., 2016) refer to the approach of utilizing one network, called the hypernetwork, to produce the weights for another network. This technique offers an abstraction that mimics the natural relationship between a genotype, represented by the hypernetwork, and a phenotype, represented by the main network. Hypernetworks have been recently showcased their capability of generating weights for various types of networks, ranging from simple modules like MLPs to more complex ones such as LSTMs. These Hypernetworks have achieved remarkable results on different sequence modelling tasks including handwriting generation, character-level language modelling, and neural machine translation. By doing so, Hypernetworks have challenged the traditional weight-sharing approach for recurrent networks.

In one of the mentioned articles at the beginning of this thesis, a recent coauthored contribution (Koyuncu et al., 2023), currently under review as a conference paper, we demonstrate that the methodology of Variational Autoencoders can be applied in conjunction with hypernetworks for generating weights of a MLP module, thus efficiently learning to generate continuous functions $p(\boldsymbol{y}|\boldsymbol{x})$ that map any desired coordinate $\boldsymbol{x}$ to features $\boldsymbol{y}$.

2.3.5. Training deep neural networks

A method commonly used to efficiently compute the gradients of the objective with respect to the parameters is **backpropagation**. This technique, introduced by (Rumelhart et al., 1986; Williams and Zipser, 1989; Werbos, 1988), is also known as the backward pass of the network. Backpropagation involves repeated application of the chain rule for

partial derivatives. Typically, **forward propagation** is the term employed for describing how information flows through the network when an input $\boldsymbol{x}$ generates an output $\boldsymbol{y} = f(\boldsymbol{x})$. This forward propagation produces a scalar cost $J(\boldsymbol{\theta})$ to be minimized with respect to the parameters. To do so, gradients $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ could be computed using an analytical expression, that is relatively easy to be computed. Nevertheless, the numerical evaluation of these quantities can result in computationally expensive optimizations.

The backpropagation algorithm (often referred to as **backprop**), provides a simple and efficient procedure for flowing the information backwards through the networks, from the cost function to the data, in order to compute the gradient. The first step is to compute the derivatives of the cost function relative to the output units. Second, by recursively applying the Chain Rule in each layer of the model, the expression for the gradient of each scalar with respect to any node in the computational graph is recursively computed. More detailed information on the backwards algorithm can be found at (Goodfellow et al., 2016).

Instead of using the entire dataset to update the model parameters in each iteration, neural networks make use of **Stochastic Gradient Descent** (**SGD**) for updating the weights by taking a small random sample of the data called a **minibatch**. The gradients of the loss function with respect to the weights are calculated using this minibatch, and the weights are then updated in the direction of the negative gradient. This process is repeated recurrently until the network reaches a point of convergence where the loss function is minimized. Using minibatches allows for faster convergence, reduces the memory requirements for training, and allows for parallelization of the computations. SGD has been the backbone of training deep neural networks and has proven to be highly effective. However, a vast amount of variations of SGD such as Adaptive Moment Estimation (Adam, Kingma and Ba (2014)) or Root Mean Square Propagation (RMSProp, Hinton et al. (2012)) have been developed to further improve training speed and accuracy.

2.4. Deep Generative models

Thanks to the exponential growth of computational capacity, and the consequent possibility of training deeper neural networks, deep generative modeling has become one of the leading AI research directions in the last few decades. The power of neural networks for applying highly complex non-linear functions, added to the possibility of easily propagating gradients of a considered objective with respect to their parameters following the Backwards algorithm (Rumelhart et al., 1985), has made them a natural choice for learning complex probabilistic dependencies in high dimensional spaces and solving tasks in typical modalities considered in Machine Learning. Several strategies for learning generative models with complex dependencies modeled by neural networks have been developed in the recent years. Mainly, as depicted in Figure 2.10, they can be divided into two groups. Firs, *explicit models* are those where the density $p(\boldsymbol{x})$ is explicitly designed and can be evaluated or approximated. Within this group, we can differentiate between models with a *tractable density*, and models with *intractable density*, i.e. where some approximation is required due to the intractability of the marginal log-likelihood function. Second, *implicit models* focus in obtaining good quality samples, and typically require adversarial training methods that are potentially unstable.

In the upcoming sections, a brief description of each mentioned deep generative model is included. More details on all of them can be found in resources like (Tomczak, 2022). This thesis focuses in explicit, non-tractable density models such as Variational Autoencoders (VAEs), and includes an extended presentation on their fundamentals and state

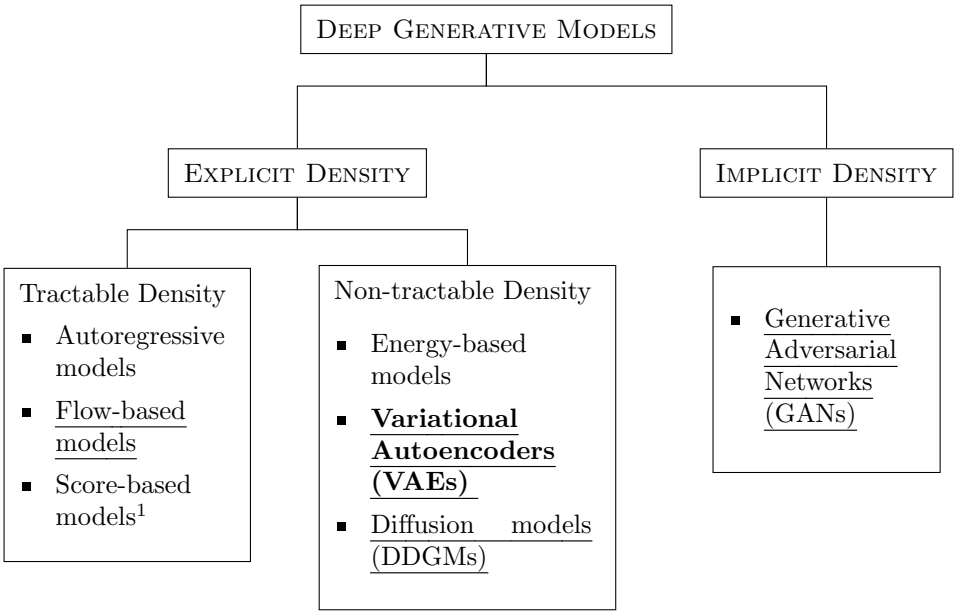


Figure 2.10: Deep Generative Models classification.

Model	Stable training	Exact likelihood	Fast sampling	Easy reconstruction	Invertible	Compression
Autoregressive	✓	✓	✗	–	–	✗
Flows	✓	✓	✓	✓	✓	✗
Score-based	✓	✓	✗	–	–	✗
Energy-based	✗	✗	✗	–	–	✗
VAEs	✓	✗	✓	✓	✗	✓
DDGMs	✓	✗	✗	✗	✗	✗
GANs	✗	✗	✓	✗	✗	✗

Table 2.1: Comparison among deep generative models with explicit density.

of the art in Chapter 3.

2.4.1. Deep Autoregressive models

In autoregressive modeling (ARM), the density $p(\boldsymbol{x})$ is represented by recurrently conditioning each element in $\boldsymbol{x}$ on the previous elements:

$$p(\boldsymbol{x}) = p\left(x_0\right) \prod_{i=1}^{D} p\left(x_i \mid \boldsymbol{x}_{<i}\right) \tag{2.36}$$

where x_0 refers to the first element, $\boldsymbol{x}_{<i}$ denotes all dimensions of $\boldsymbol{x}$ to the left of element x_i, and the order for indexing elements $i = \{1, ..., D\}$ will depend on the nature of the considered data, i.e. time order in sequences (where index t is typically preferred), or pixel order in images. In the context of deep generative models, functions for conditional distributions $p(x_i \mid \boldsymbol{x}_{<i})$ can be modeled by training deep neural networks, being Recurrent Neural Networks (RNNs) and their improved versions the typical choice, and choosing an appropriate likelihood functions for the considered data. Although learning all conditional distributions in these models would be computationally inefficient, such

complexities are relaxed by taking advantage of *causal convolutions*[1]. Recent examples of learning the AR dependencies using RNNs can be found in (Salinas et al., 2020; Oord et al., 2016; Moreno-Pino et al., 2023), for sequences generation, solving tasks like time series forecasting, audio or generalized sequence generation, and (Van Den Oord et al., 2016; Van den Oord et al., 2016), for image generation via conditioning pixels using an AR path. Adaptations of non-recurrent architectures can also be considered, as proposed recently in (Salimans et al., 2017), where Convolutional Neural Networks (CNNs) are applied sequentially over the image using masked convolutions.

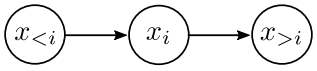


Figure 2.11: Autoregressive dependencies.

Generation can be performed by, starting with a sample from the prior $p(x_0)$, recurrently obtaining the parameters of $p(x_i \mid \boldsymbol{x}_{<i})$ using the trained NN and sampling from the considered distribution. Another interesting property of these models is the possibility of conditional generation.

ARMs are robust density estimators, mainly because they can learn long-range statistics. Likelihood computation is tractable in ARMs by simply evaluating (2.36). Nevertheless, their main drawback is that, for generation, they require recurrently sampling and parameterizing the conditionals $p(x_i \mid \boldsymbol{x}_{<i})$, ending in a slow process. Further, unlike latent variable models, they lack a reduced latent representation, which makes them not suitable for compression or dimensionality reduction.

2.4.2. Flow-based models

Normalizing Flows utilize the idea of the change of variables to model flexible and high-dimensional distributions over images audio or other data sources. In essence, they are a hierarchical model where transformations between layers are invertible. Considering a standard prior $p(\boldsymbol{z}_0)$ over a starting latent variable, the set of invertible transformations are applied sequentially:

$$p(\boldsymbol{x}) = p(\boldsymbol{z}_0 = f^{-1}(\boldsymbol{x})) \prod_{i=1}^{K} |\boldsymbol{J}_{f_i}(\boldsymbol{z}_{i-1})|^{-1}, \tag{2.37}$$

where $\boldsymbol{J}_f$ denotes the Jacobian matrix, and the functions f_i are parameterized using invertible deep neural networks that allow for relatively easy Jacobian computation. Linear, volume-preserving transformations were firstly considered by (Dinh et al., 2014; Tomczak and Welling, 2016). Further non-linear extensions utilized theorems on matrix determinants resulting in specific transformations, namely Planar Flows (Rezende and Mohamed, 2015b) and Sylvester Flows (Berg et al., 2018; Hoogeboom et al., 2020). A different line of work focuses on formulating invertible transformations for which the Jacobian-determinant could be calculated easily, like coupling layers in RealNVP (Dinh et al., 2016).

Flow-based models have become popular in recent years due to their ability to compute exact likelihoods. However, these models are limited by the use of specific invertible neural networks with tractable Jacobian-determinants. These transformations, such as

[1]Causal convolution ensures that the prediction $p(\boldsymbol{x}_t \mid x_{<t})$ does not depend on data at future timesteps $x_{\geq t}$. *Masked convolution* is equivalently considered for images.

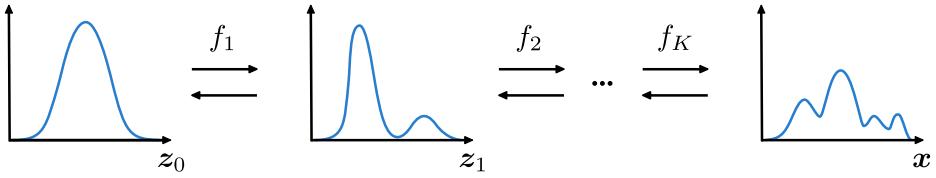


Figure 2.12: Unidimensional example with invertible transformations for shaping multimodal data densities from the unimodal latent space.

affine coupling layers and autoregressive networks, are often computationally expensive and can restrict the expressive power of the model. Additionally, flow-based models do not have an interpretable latent space, which can make it difficult to understand the learned representations. Moreover, unlike other generative models, such as variational autoencoders, flow-based models do not explicitly allow for data compression, which can be important for applications such as image or audio compression. These limitations make flow-based models less suitable for certain tasks and highlight the importance of considering the strengths and weaknesses of different generative models when choosing a model for a specific application.

2.4.3. Score-based models

In score-based models, the key idea is to model the gradient of the log-probability density function, $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$, a quantity often known as the (Stein) score function (Liu et al., 2016), depicted as a vector field in Figure 2.13. They are not required to have a tractable normalizing constant, and can be directly learned by score matching (Hyvärinen and Dayan, 2005; Vincent, 2011). Once fitted $\boldsymbol{s}_\theta(\boldsymbol{x}) \approx \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$, by using Langevin dynamics (Parisi, 1981; Grenander and Miller, 1994), samples from $p(\boldsymbol{x})$ can be obtained by initializing the MCMC chain from an arbitrary prior distribution $\boldsymbol{x}_0 \sim \pi(\boldsymbol{x})$, and then iterating like the following:

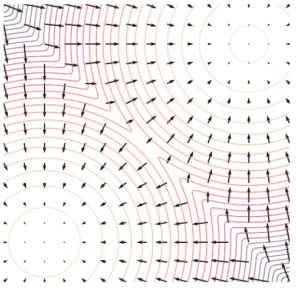


Figure 2.13: Score function (vector field) for a mixture of two Gaussians (contours).

$$\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \boldsymbol{\epsilon} \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}) + \sqrt{2\epsilon}\boldsymbol{z}_i, \quad i = 0, 1, \cdots, K \tag{2.38}$$

where $\boldsymbol{z}_i \sim \mathcal{N}(0, I)$. When $\epsilon \to 0$ and $K \to \infty$, $\boldsymbol{x}_K$ converges to a sample from true $p(\boldsymbol{x})$, under some regularity conditions. The error is minimal when considering a sufficiently small ϵ and sufficiently large K.

To avoid that only in regions with probability mass are accurately captured (like the modes in Figure 2.13), data is perturbed with a set of noise levels $\sigma_1 < ... < \sigma_L$, and using as objective the following sum of Fisher divergences:

$$\sum_{i=1}^{L} \lambda(i) \mathbb{E}_{p_{\sigma_i}(\boldsymbol{x})} \left[\| \nabla_{\boldsymbol{x}} \log p_{\sigma_i}(\boldsymbol{x}) - \mathbf{s}_\theta(\boldsymbol{x}, i) \|_2^2 \right]. \tag{2.39}$$

where $\lambda(i)$ is a positive weighting function, typically chosen to be $\lambda(i) = \sigma_i^2$. This method is known as Annealed Langevin Dynamics (Figure 2.14). Further, by extending Score-based models with stochastic differential equations (Song et al., 2020), the number of

noise scales can be generalized to infinity, which allows for higher quality samples, exact log-likelihood computation, and controllable generation for inverse problem solving.

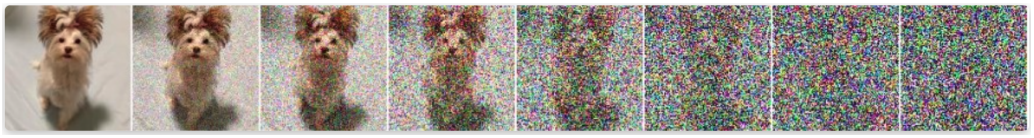


Figure 2.14: Sampling images with multiple Gaussian noise levels using Annealed Langevin Dynamics.

Score-based models have achieved state-of-the-art performance on many downstream tasks and applications. To list some of them: image generation (Song and Ermon, 2019; 2020; Ho et al., 2020; Song et al., 2020; Dhariwal and Nichol, 2021), audio synthesis (Chen et al., 2020; Kong et al., 2020; Popov et al., 2021), shape generation (Cai et al., 2020), and music generation (Mittal et al., 2021). Additionally, inverse problem solving allows for conditional generation, with interesting applications such as image inpainting (Song and Ermon, 2019; Song et al., 2020), image colorization (Song et al., 2020) or medical image reconstruction (Jalal et al., 2021), among others.

The two main drawbacks of using score-based generative models are, first, the slow sampling speed due to the Langevin MCMC sampler, and second, scores are only defined for continuous distributions, thus preventing to use score-based models for generating discrete data.

2.4.4. Energy-based models

Energy-based models (EBMs) (LeCun et al., 2006) are based Boltzmann machines, that were firstly proposed by (Ackley et al., 1985; Smolensky, 1986), with ideas taken from statistical physics and formulated by cognitive scientists. In contrast with likelihood-based models, instead of proposing a specific distribution, an energy function $E(\boldsymbol{x})$ is considered for assigning an energy value to a given input state. This energy function is the exponent of the density, in the form:

$$p(\boldsymbol{x}) = \frac{1}{\mathcal{Z}} e^{-E(\boldsymbol{x})}, \tag{2.40}$$

where $\mathcal{Z} = \int_x e^{-E(\boldsymbol{x})} d\boldsymbol{x}$ is the normalization constant, also known as partition function, that normalizes the distribution. The main advantage of learning the Energy function is that, differently from density functions, there are no restrictions on the normalization, so it can be parameterized it with neural networks.

Energy-Based Models (EBMs) offer a versatile and attractive means of representing uncertainty. Nonetheless, despite recent progress, training EBMs on high-dimensional data remains a difficult problem (Song and Kingma, 2021; Duvenaud et al., 2021) as the current leading approaches are expensive, fragile, and demand substantial expertise in the domain and extensive parameter tuning to achieve successful outcomes.

2.4.5. Variational Autoencoders

Variational Autoencoders (VAEs) (Kingma and Welling, 2013; Rezende et al., 2014) are likelihood-based deep generative models that compress and decompress data following an autoencoder framework. Their operation is illustrated with a diagram in Figure 2.15. VAEs are latent variable models that extend Probabilistic PCA (Section

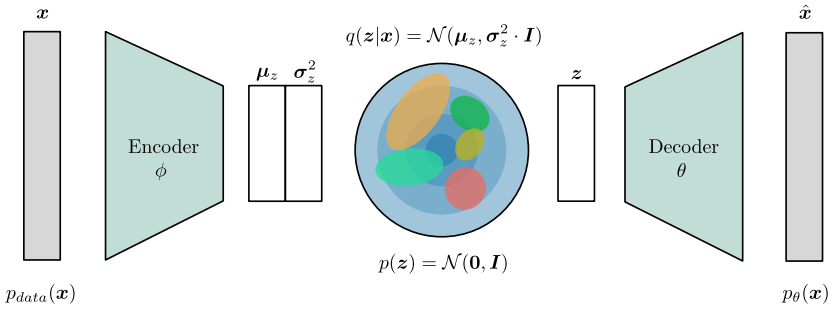


Figure 2.15: Variational Autoencoder.

2.2.1) by using complex non-linear functions parameterized by deep neural networks. It is this complexity added by the networks that makes the inference of the true posterior over latent variables $p(\boldsymbol{z}|\boldsymbol{x})$ intractable. Thus, they require to introduce an approximation by training an auxiliary model to perform amortized variational inference (Cremer et al., 2018; Zhang et al., 2018) with an encoder-decoder architecture. The encoder network, with parameters $\boldsymbol{\phi}$, maps observations $\boldsymbol{x} \in \mathbb{R}^D$ to the parameters of the approximate posterior $q_\phi(\boldsymbol{x}|\boldsymbol{z})$, typically Gaussian, and with lower dimensionality $\boldsymbol{z} \in \mathbb{R}^d$. The decoder network, with parameters $\boldsymbol{\theta}$, maps the latent variable sampled from this approximate posterior to the parameters of the data likelihood $p_\theta(\boldsymbol{x}|\boldsymbol{z})$. Under this approximation, the objective function is the Evidence Lower Bound (ELBO) on the true marginal log likelihood, given by

$$\mathcal{L}(\boldsymbol{x}) = \mathbb{E}_{q_\phi}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right] - D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})), \tag{2.41}$$

where the first term, named *reconstruction*, maximizes the likelihood of the data for given samples of the approximate posterior, and the second term, encourages that the mismatch between approximate posterior and prior is as small as possible.

The flexibility of VAEs makes them a powerful class of models. Differently from flow-based models, they do not require the invertibility of neural networks, thus, we can use any arbitrary architecture for encoders and decoders, with the cost of lower bounds of the log marginal likelihood being required. In contrast to ARMs, they learn a low-dimensional data representation and we can control the bottleneck (i.e., the dimensionality of the latent space), which adds extra interpretability of the way the latent space is structured in order to represent hidden factors of the data being encoded. Nonetheless, they also suffer from several issues. Apart from the one mentioned before (a gap between the ELBO and the true log-likelihood), other problems to be considered are: the necessity of an efficient integral estimation for computing the first term of 2.41, the *posterior collapse*, the *hole problem*, the *mode collapse* or the control of the gap by carefully designing of the variational posterior and the prior. VAEs are more extensively discussed in Chapter 3 of this thesis.

Hierarchical VAEs

Among the recent variations of VAEs, of special interest in this thesis are **Hierarchical VAEs**, which are extensively reviewed in Section 3.7, and are one of the leading topic directions in research for improving VAE frameworks. In this type of VAEs, a set of autoregressive variables, organized hierarchically, conform a more structured latent space, thus providing a much more flexible prior. Their increased flexibility makes them a

powerful and robust choice when high dimensional, diverse multimodal datasets are to be modeled.

2.4.6. Difussion Models

Diffusion-based Deep Generative Models (**DDGMs**) learn to generate data by recurrently denoising samples of a latent hierarchy. They can also be understood as a variation of hierarchical VAEs where the bottom-up path that defines the variational posteriors is defined by a diffusion process $q(\boldsymbol{z_i}|\boldsymbol{z}_{i-1})$, and the top-down path $p_\theta\left(\boldsymbol{z}_i|\boldsymbol{z}_{i+1}\right)$ is parameterized by deep neural networks that reverse the diffusion. Differently from Hierarchical VAEs, the bottom-up path does not need to contain any learnable parameters. Consequently, DDGMs avoid the posterior collapse problem in deeper layers. The joint distribution is modeled as a first-order Markov chain with Gaussian transitions, namely:

$$p_\theta\left(\boldsymbol{x}, \boldsymbol{z}_{1:T}\right) = p_\theta\left(\boldsymbol{x} \mid \boldsymbol{z}_1\right)\left(\prod_{i=1}^{T-1} p_\theta\left(\boldsymbol{z}_i \mid \boldsymbol{z}_{i+1}\right)\right) p_\theta\left(\boldsymbol{z}_T\right), \tag{2.42}$$

where $p_\theta(\boldsymbol{z_0}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and differently from VAEs, the latent variable has the same dimension than the data, i. e. $\boldsymbol{x}, \boldsymbol{z} \in \mathbb{R}^D$. The posterior distributions (intractable) is approximated by the diffusion process:

$$q_\phi\left(\boldsymbol{z}_{1:T}|\boldsymbol{x}\right) = q_\phi\left(\boldsymbol{z}_1|\boldsymbol{x}\right)\left(\prod_{i=2}^{T} q_\phi\left(\boldsymbol{z}_i|\boldsymbol{z}_{i-1}\right)\right) \tag{2.43}$$

where $q_\phi\left(\boldsymbol{z}_i|\boldsymbol{z}_{i-1}\right) = \mathcal{N}\left(\boldsymbol{z}_i|\sqrt{1-\beta_i}\boldsymbol{z}_{i-1}, \beta_i\boldsymbol{I}\right)$, and $\boldsymbol{z}_0 = \boldsymbol{x}$. In broader terms, each layer scales the sample from previous layer by $\sqrt{1-\beta_i}$ and adds a new noise term with variance β_i. This diffusion-based inference is comparable to the data perturbation in score-based models, depicted in Figure 2.14. The variances $\beta_{1:T}$ can be learned, fixed or linearly increased over a considered interval, as proposed in (Ho et al., 2020).

DDGMs are a specific variant of Hierarchical VAEs, and whether it is indeed more beneficial to use fixed variational posteriors by sacrificing the possibility of having a bottleneck is an open question. Flow-based models are also connected with DDGMs, in the sense that both classes of models map noise to data through a sequence of layers. In Flow-based models, the exact likelihood can be computed, but calculating the expensive Jacobian determinant is the price to pay. DDGMs are also closely related to stochastic differential equations, and thus, their theoretical properties seem to be especially interesting (Huang et al., 2021; Song et al., 2021; Tzen and Raginsky, 2019). They have also obtained astonishing results for image (Ho et al., 2020; Kingma et al., 2021; Saharia et al., 2022), audio (Kong and Ping, 2021), and text (Austin et al., 2021; Hoogeboom et al., 2021) synthesis while being relatively simple to implement. Nevertheless, whilst DDGMs have recently become one of the leading directions in DGMs research, their main drawbacks are, mainly, the lack of compression, and the slow sampling process.

2.4.7. Generative Adversarial Networks

Generative Adversarial Networks (**GANs**) are a powerful class of implicit generative models, introduced by (Goodfellow et al., 2020). As depicted in Figure 2.16, they consist of two neural networks: a generator and a discriminator, that are trained in an adversarial manner. The generator network, $G_\theta(\boldsymbol{z})$, learns to transform samples from a latent variable into synthetic data that ideally is indistinguishable from the real data,

while the discriminator network $D_\phi(\cdot)$ is trained to distinguish between the synthetic data produced by the generator and the real data. The idea of the adversarial problem could be traced back to (Schmidhuber, 1990), and leads to a Nash equilibrium, where the generator produces synthetic data that is indistinguishable from the real data by the discriminator. In other terms, the goal is that i) the discriminator is good at detecting fake samples that come from the generator, i.e. $D_\phi(G_\theta(\boldsymbol{z})) = 0$, and $D_\phi(\boldsymbol{x}) = 1$, and ii) the generator is good at cheating the discriminator, i.e. $D_\phi(G_\theta(\boldsymbol{z})) = 1$. This leads to the following adversarial loss:

$$\min_\theta \max_\phi \quad \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \left[\log D_\phi(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})} \left[\log\left(1 - D_\phi\left(G_\theta(\boldsymbol{z})\right)\right)\right]. \tag{2.44}$$

GANs have been successfully applied to various applications, obtaining outstanding results in image synthesis (Radford et al., 2015; Karras et al., 2019; Brock et al., 2018), style transfer (Gatys et al., 2015; Johnson et al., 2016a), and representation learning (Donahue et al., 2017; Miyato and Koyama, 2018). The generator network can be any type of neural network architecture, making GANs highly flexible and suitable for a wide range of tasks. Additionally, GANs can generate high-quality samples and have been shown to capture the underlying structure of the data distribution in many cases. However, GAN training can be challenging due to instability and mode collapse, and careful design of the architecture and training procedure is often required to achieve good results. Furthermore, due to their implicit design, in their naïve configuration, they do not allow for any inference-related task. Recent approaches as the Bidirectional GAN (Donahue et al., 2017) made it possible to learn the inverse mapping from data to latent representation.

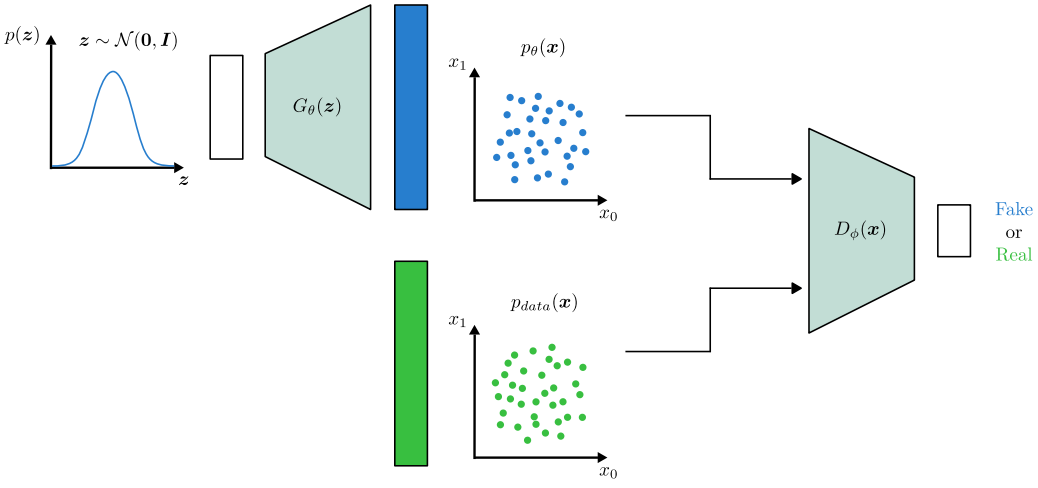


Figure 2.16: Generative Adversarial Network.

2.4.8. Inductive Bias in Latent Variable Models

Deep LVMs aim to uncover the underlying factors in the data they observe (Bengio et al., 2013). To achieve this, as discussed in this chapter, they learn a joint distribution of the observed data, $\boldsymbol{x}$, and hidden factors, $\boldsymbol{z}$, which is represented as $p(\boldsymbol{x}, \boldsymbol{z})$. The key to obtaining a meaningful representation is to determine the posterior distribution of the hidden factors, $p(\boldsymbol{z}|\boldsymbol{x})$. However, determining what constitutes a"useful" representation

is not a straightforward task. A useful representation may refer to a latent representation that is easily interpretable or captures meaningful information about the generative process. For example, in a DGM trained to generate images of bedrooms, a dimension of $\boldsymbol{z}$ could encode the color of the bedroom.

As pointed out by (Tomczak, 2021), maximizing the likelihood function to learn a latent variable model might not result in useful representations. The training problem of learning parameters θ can also be reconsidered as an unconstrained optimization problem with the following objective:

$$D_{\mathrm{KL}}\left(p_{\mathrm{data}}\left(\boldsymbol{x}\right)\|p_\theta(\boldsymbol{x})\right) = -\mathbb{H}\left[p_{\mathrm{data}}\left(\boldsymbol{x}\right)\right] + \mathbb{CE}\left[p_{\mathrm{data}}\left(\boldsymbol{x}\right)\|p_\theta(\boldsymbol{x})\right], \tag{2.45}$$

where the entropy of the empirical distribution (first term) does not depend on the parameters, and thus, is constant. The cross-entropy (second term) can be derived as

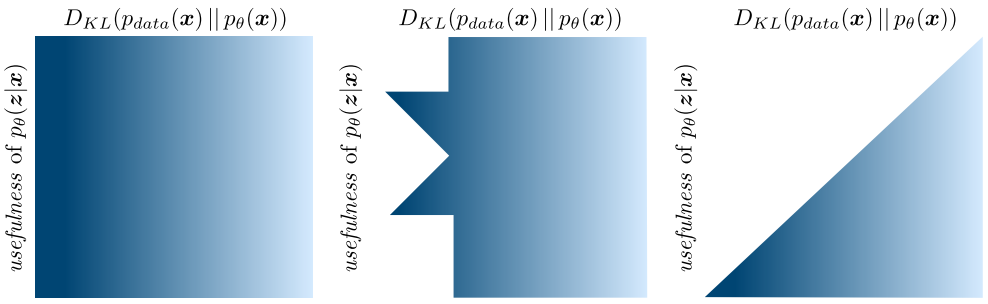
$$\mathbb{CE}\left[p_{\mathrm{data}}\left(\boldsymbol{x}\right)\|p_\theta(\boldsymbol{x})\right] = -\int_{\boldsymbol{x}} p_{\mathrm{data}}\left(\boldsymbol{x}\right)\log p_\theta(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = -\frac{1}{N}\sum_{n=1}^{N}\log p_\theta\left(\boldsymbol{x}_n\right), \tag{2.46}$$

which corresponds to the negative log-likelihood function, and as observed, it is independent of the hidden factors $\boldsymbol{z}$, due to their marginalizion for obtaining $p_\theta(\boldsymbol{x}_n) = \int_z p_\theta(\boldsymbol{x}_n, \boldsymbol{z})d\boldsymbol{z}$. The intuition behind this conclusion is that in DGMs, the marginal over observable variables are optimized, due to not having access we to actual values of latent variables. As a consequence, controlling the way DGMs capture the hidden information is typically arduous, since Maximum Likelihood optimization only encourages accurate data reconstructions. Furthermore, some models lead to possible interpretations on how they organize the latent space. Within potential pitfalls, a latent variable model can learn to disregard the latent variables completely.

With illustrative purposes, in Figure 2.17a, a fictitious search space of all possible LVMs and their evaluation in terms of goodness of fit (2.45) in the horizontal axis, and "usefulness" in the vertical axis, is depicted. An ideal model would be considered as the one that lied in the top-left corner, since it would optimize both criteria. Nevertheless, as discussed above, the maximum likelihood objective in (2.46) is not sufficient for accomplishing the two objectives. It is possible to find a model that completely disregards the latent variable while maximizing the fit to data (bottom-left corner). In fact, infinite models would be equally good with respect to (2.46) but with completely different posteriors over latent variables.

In contrast, in practice, we observe that properly learned latent variables are useful. To illustratively represent this paradox, an example is included in Figure 2.17b, where a type of LVM is considered by choosing a neural architecture for parameterizing a probabilistic model. For instance, using some a bottleneck design for compressing data into a latent space, like in VAEs, might lead to a situation where latent variables contain useful information about observed data. This design restricts the search space to 2.17b, where two spikes represent those hypothetical models that optimally minimize the KL whilst achieving meaningful representations.

Again, theoretically, the power of DNNs can be put to rebut this hypothesis. In Figure 2.17c this problematic is depicted. Choosing a specific architecture modifies the search space in comparison with 2.17a. Nevertheless, in contrast with 2.17b, if the conditional likelihood $p(\boldsymbol{x}|\boldsymbol{z})$ is parameterized by an infinitely flexible (enormous DNN), the model could learn to mimic the data distribution $p_{data}(\boldsymbol{x})$ almost perfectly, thus completely disregarding the latent space. In these hypothetical large models, there would be a trade-off between i) accurately replicating the training distribution (bottom-left corner) with



(a) All LVMs (b) A class of LVM (c) LVMs with flexible $p(\boldsymbol{x}|\boldsymbol{z})$

Figure 2.17: Illustrative diagram of the "usefulness" (vertical axis) of latent representations deep generative models trained for mimicking the true data distribution (horizontal axis and color intensity). In (a), no restrictions are consider such that infinitely good models can perfectly mimic the true data distribution. In (b), deep neural networks restrict the search space to possibly find optimal configurations. In (c), extremely flexible DNNs are considered such that a model can disregard latent variables.

meaningless latent variables, versus ii) reducing this accuracy but achieving useful latent representations (top-right corner), represented by the linear tendency in Figure 2.17c. Unfortunately, it remains unclear how to find points where meaningful representations and accurate data fitting are simultaneously achieved.

The picture presented in the previous paragraph can be pessimistically interpreted, in the sense that choosing a proper class of models that allow achieving useful latent representations, is a highly non-trivial task. Nevertheless, within the context of VAEs, which comprises a pillar of the present thesis, this problem can be handled by mainly three alternatives, as it will be extensively covered in Chapter 3. First, some previous work formulated a constrained optimization problem (Phuong et al.; Rezende and Viola, 2018), modifications of the loss function (Higgins et al., 2016; Burgess et al., 2018) or added an auxiliary regularizer (Sinha and Dieng, 2021; Tomczak, 2016) to implicitly define and increase *usefulness* of the latent space. Second, semi-supervision can be incorporated to encourage for meaningful representations (Bouchacourt et al., 2018). Third, by carefully designing the latent space *usefulness* can be implicitly enforced.

This thesis is focused on the third approach. The knowledge that is incorporated into model components through their design is typically referred to as the **inductive bias**. As it will be discussed in several sections, designing the structure of the latent space lead to not only useful, but also highly interpretable latent spaces. For instance, as it will be discussed in Chapters 3 and 5, a hierarchical latent space design encourages deeper latent variables to capture more abstract or general information, while the shallowest layers encode specific information. Similarly, by creating a global space, shared by multiple observations, global factors can be represented (see Chapter 4).

CHAPTER 3
Variational Autoencoders

In Section 2.2.1, the unsupervised PCA method was introduced as a means of linearly projecting observed data into a reduced space. This is accomplished via an optimal linear transformation that reduces the explained variance of the transformed data into fewer dimensions than the original space. Probabilistic PCA, the probabilistic version, was also discussed as a generative, latent variable model that transforms a latent space of reduced dimensionality into the observed space using linear transformations. The linearity enables analytical expressions for all the relevant probability distributions, including the marginal likelihood, $p(\boldsymbol{x})$, and the posterior of the latent variable, $p(\boldsymbol{z}|\boldsymbol{x})$, given $p(\boldsymbol{x}|\boldsymbol{z})$. The former is necessary for optimizing the parameters of the linear transformation using maximum likelihood, while the latter ideally captures useful information about the generative factors of the data.

Chapter 2 presented Deep Learning as a field that allows for the efficient learning of complex non-linear functions of high-dimensional data, such as tabular data, images, or sequences, thanks to the power of Deep Neural Networks. In Section 2.4, the various types of Deep Generative Models were introduced, along with their principal characteristics, advantages, applications, and limitations. Among these variants, Variational Autoencoders (Kingma and Welling, 2013; Rezende et al., 2014) share similar concepts with Probabilistic PCA. They employ a reduced latent space and aim to learn the functions that map observed data to the latent space, as well as generate new data from it. The primary distinction is that VAEs are not restricted to using linear functions, but instead use Neural Networks, including all the architectures discussed in Section 2.3, to parameterize the distributions. Compared to a single linear transformation, Neural Networks have a vast potential since the data typically encountered in the real world is far from being linearly generated.

The automatic discovery of non-linearities through NNs is a highly valuable technique and has been extensively studied in recent years. Variational Autoencoders have achieved outstanding results in numerous applications. To list some of the most important ones, the following recent works are referred for **image generation** (Razavi et al., 2019b; Vahdat and Kautz, 2020; Child, 2020), **video generation** (Yan et al., 2021; He et al., 2018; Bhagwatkar et al., 2021), text generation (Bowman et al., 2015), neural machine translation (Sutskever et al., 2014), music generation (Roberts et al., 2018), outlier detection (Chauhan et al., 2022; Denouden et al., 2018; Xiao et al., 2020; Serrà et al., 2019) time-series analysis (Tang and Matteson, 2021a; Chung et al., 2015; Fraccaro et al., 2016), recommendation systems (Shenbin et al., 2020; Liang et al., 2018). However, the usage of NNs also comes with issues that must be addressed, as will be discussed further in this chapter.

This doctoral thesis presents several research contributions, detailed in Chapters 4 and 5, that utilize VAEs as a generative model. Therefore, the upcoming sections provide a comprehensive discussion of VAEs. Section 3.1 begins with the definition of a Deterministic Autoencoder as a starting point. Next, Section 3.2 discusses the base definition of a Variational Autoencoder. Brief comments on typical challenges of VAEs are provided

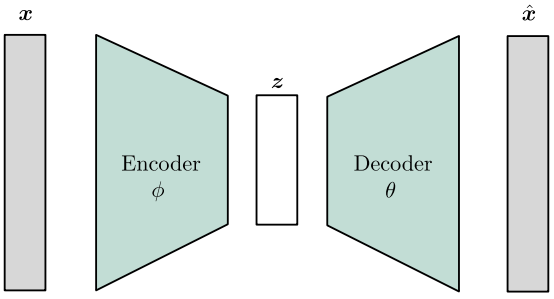


Figure 3.1: Diagram of a Deterministic Autoencoder. An encoder maps original data $\boldsymbol{x}$ to latent codes $\boldsymbol{z}$, such that a decoder learns to reconstruct them to $\hat{\boldsymbol{x}}$.

in Section 3.3. Finally, Sections 3.4 to 3.8 review all the challenges addressed in the contributions of this thesis.

3.1. The Autoencoding Framework

In this first section, the **Deterministic Autoencoder**, also known as **Denoising Autoencoder**, **Deep Autoencoder** or simply **Autoencoder** is presented as an introduction to VAEs. As a non-probabilistic model, it is not necessarily required to include any probability distribution to be parameterized. The focus is on the idea that it can be compared to PCA, since both are non-probabilistic methods that are typically employed for dimensionality reduction or feature learning. The main difference is that NNs are used by Autoencoders to approximate the functions learned for compressing the data into the hidden space and reconstructing data from the hidden representation.

In the Autoencoder context (depicted in Figure 3.1), the operation of mapping the observed data to the hidden space is known as **encode**, and the corresponding function is learned by the **encoder** neural network. Similarly, the **decoder** neural network is used for modeling the function that decodes hidden codes into data in the observed space. The parameters of the encoder E_ϕ and decoder D_θ can be denoted by θ, ϕ, and the reconstruction of a datapoint can be expressed as:

$$\hat{\boldsymbol{x}} = D_\theta(E_\phi(\boldsymbol{x})), \tag{3.1}$$

which is also depicted in Figure 3.1. To learn the parameters θ and ϕ, the error on all the reconstructions is minimized, and it can be defined as:

$$\mathcal{L}(\boldsymbol{x}; \theta, \phi) = \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}n(\boldsymbol{x}, ; D\theta(E_\phi(\boldsymbol{x}))). \tag{3.2}$$

Using the backprop algorithm for computing the gradients, and SGD for iteratively approaching the minimum, a Deep Autoencoder can be easily trained. Nevertheless, due to its non-probabilistic nature, diversity in data generation is complicated to achieve. As it will be discussed, using a probability perspective over the autoencoding idea provides an extra degree of flexibility to the model, as well as the capability of quantifying uncertainty.

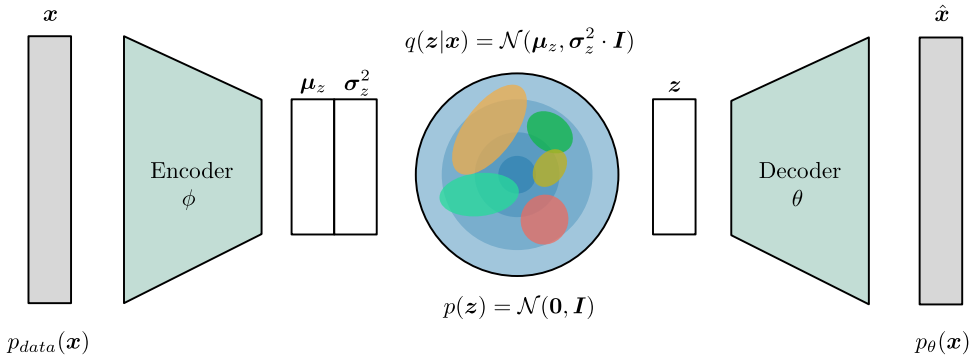


Figure 3.2: Detailed diagram of a Variational Autoencoder. A latent space with standard Gaussian prior probability (blue contour) generates realistic data by decoding $\boldsymbol{z}$ samples. The parameters of the approximate posterior per observation (colored contours) are outputs of the encoder.

3.2. Auto-encoding Variational Bayes

As previously noted, the **Variational Autoencoder** (VAE), illustrated in Figure 3.2, is considered a non-linear extension of Probabilistic PCA, which approximates complex non-linear functions using neural networks. The marginal likelihood that is optimized for parameter learning via maximum likelihood remains the same:

$$p(\boldsymbol{x}) = \int_{\boldsymbol{z}} p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z}, \tag{3.3}$$

however, the parameterization of $p(\boldsymbol{x}|\boldsymbol{z})$ in VAEs is so complex that the integral becomes intractable. A simple Monte Carlo approach can be used, as shown in (2.15), where the integral is approximated using samples from the prior $\boldsymbol{z}_s \sim p(\boldsymbol{z})$. However, this approach is computationally expensive and exploring the latent space becomes even more challenging as the dimensionality of $\boldsymbol{z} \in \mathbb{R}^d$ increases.

It should be noted that more flexible priors with learnable parameters $p_{\theta_z}(\boldsymbol{z})$ can be designed for generating data with advantages, but obtaining samples from them is not straightforward. Advanced Monte Carlo techniques such as Hamiltonian Monte Carlo can be used, but they also suffer from the curse of dimensionality. In Chapter 5, one of the main contributions of this thesis, which is the efficient use of Hamiltonian Monte Carlo to enhance the inference of VAEs, is discussed.

To tackle this problem, Variational Inference is employed in VAEs, as discussed in the subsequent sections. Variational Inference achieves an efficient approximation of the log evidence $\log p(\boldsymbol{x})$.

3.2.1. Variational Inference

Variational Inference, firstly introduced by Jordan et al. (1999), is an approximate inference technique that utilizes a family of distributions $q_\phi(\boldsymbol{z})$, parameterized by ϕ, to approximate the intractable posterior distribution $p(\boldsymbol{z}|\boldsymbol{x})$ of the latent variables. The requirement for this family of distributions is that they assign non-zero probability to any $\boldsymbol{z}$ in its domain. A typical choice for $q_\phi(\boldsymbol{z})$ is the Gaussian distribution with parameters $\phi = \boldsymbol{\mu}_z, \boldsymbol{\sigma}_z^2$, resulting in $q_\phi(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_z, \boldsymbol{\sigma}_z^2\boldsymbol{I})$. This is motivated by the fact that

the Gaussian distribution is a widely used distribution that is simple and flexible. The choice of this distribution also leads to tractable computation of the required parameters of $q_\phi(\boldsymbol{z})$, and low variance estimators using the reparameterization trick (Kingma and Welling, 2013).

Having defined this **approximate posterior**, typically the KL divergence of q from p is the choice of dissimilarity function to be minimised,

$$D_{\mathrm{KL}}(q_\phi(\boldsymbol{z})\|p(\boldsymbol{z}|\boldsymbol{x})) = \int_{\boldsymbol{z}} q_\phi(\boldsymbol{z}) \log \frac{q_\phi(\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{x})} d\boldsymbol{z} \tag{3.4}$$

which provides a way to approximate the intractable true posterior by means of the chosen family, being this approximation more or less accurate depending on several aspects, like the choice of the variational family or the complexity of the true posterior.

3.2.2. The Evidence Lower Bound

In order to learn the parameters of a model, an approximation of the log evidence, $\log p(\boldsymbol{x})$ is required via Variational Inference. As discussed earlier, this evidence is related to the posterior by Bayes theorem (Equation (2.12)). By substituting this identity into (3.4), we obtain the following expression

$$D_{\mathrm{KL}}(q_\phi(\boldsymbol{z})|p(\boldsymbol{z}|\boldsymbol{x})) = \int_{\boldsymbol{z}} q_\phi(\boldsymbol{z})[\log q_\phi(\boldsymbol{z}) - \log p(\boldsymbol{x}, \boldsymbol{z})]d\boldsymbol{z} + \log p(\boldsymbol{x}), \tag{3.5}$$

where the third step uses the fact that $p(\boldsymbol{x})$ is constant with respect to $\boldsymbol{z}$, so $\int_{\boldsymbol{z}} q_\phi(\boldsymbol{z}) \log p(\boldsymbol{x}) = \log p(\boldsymbol{x})$. By moving $\log p(\boldsymbol{x})$ to the left-hand side and expressing in terms of expectations, we obtain the following expression

$$\log p(\boldsymbol{x}) = D_{\mathrm{KL}}(q_\phi(\boldsymbol{z})||p(\boldsymbol{z}|\boldsymbol{x})) + \mathbb{E}_{q_\phi(\boldsymbol{z})}[\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q_\phi(\boldsymbol{z})]. \tag{3.6}$$

The first term is intractable due to the true posterior being intractable. However, since the KL divergence is strictly positive by definition, it follows that

$$\log p(\boldsymbol{x}) \geq \mathcal{L}(\boldsymbol{x}) = \mathbb{E}_{q_\phi(\boldsymbol{z})}[\log p(\boldsymbol{x}, \boldsymbol{z}) - \log q_\phi(\boldsymbol{z})], \tag{3.7}$$

where $\mathcal{L}(\boldsymbol{x})$ refers to the **Evidence Lower Bound** (**ELBO**), depicted in Figure 3.3. VAEs use the ELBO as a training objective that approximates the intractable log-evidence. It can also be expressed in terms of a new KL

$$\mathcal{L}(\boldsymbol{x}; \theta, \phi) = \mathbb{E}_{q_\phi(\boldsymbol{z})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - D_{\mathrm{KL}}(q_\phi(\boldsymbol{z})||p(\boldsymbol{z})), \tag{3.8}$$

which penalizes the dissimilarity between the approximate posterior and the prior of the latent variable. The first term in (3.8) is typically approximated via simple Monte Carlo, resulting in the following expression

$$\hat{\mathcal{L}}(\boldsymbol{x}; \theta, \phi) = \frac{1}{S} \sum_{s=1}^{S} \log p_\theta(\boldsymbol{x}|\boldsymbol{z}_s) - D_{\mathrm{KL}}(q_\phi(\boldsymbol{z})||p(\boldsymbol{z})). \tag{3.9}$$

More advanced Monte Carlo techniques are discussed in Section 3.5 for approximating the ELBO and reducing the gap between it and the intractable log-evidence.

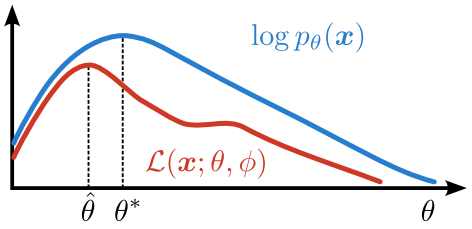


Figure 3.3: The bias of the Evidence Lower Bound. Suboptimal $\hat{\theta}$ maximizes the ELBO objective (red), but is biased respect to the true optimal parameters θ^* that maximize the intractable log-evidence (blue).

3.2.3. Amortized Variational Inference

In classical variational inference, the parameters ϕ_n for each observation $\boldsymbol{x}_n$ would require to be optimized using (3.8). Nevertheless, the expressiveness of NNs can be considered in order to perform **amortized variational inference**, by learning a mapping from observations to the corresponding variational parameters, via $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. Within this setup, ϕ denotes the parameters of the considered NN, which is also referred to as **encoder**, in the VAE terminology. Similarly, θ defines the NN for parameterizing the likelihood given a latent sample, via $p_\theta(\boldsymbol{x}|\boldsymbol{z})$. The resulting ELBO is

$$\mathcal{L}(\boldsymbol{x};\theta,\phi) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - D_{\mathrm{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{x})\|p(\boldsymbol{z})). \tag{3.10}$$

As a result, an autoencoding framework is obtained, with the difference that both the encoder and decoder are stochastic, as illustrated in Figures 3.1. The first term of the ELBO is referred to as the **reconstruction term**, that encourages a good reconstruction of the data by maximizing the likelihood given samples from the approximate posterior. Typically one sample is sufficient for obtaining reasonably accurate approximations. The second term can be understood as a **regularizer**, since, as said before, it encourages the posteriors (colored contours in Figure 3.2) to not lie far from the prior. For more complex models, i.e. when the prior is also parameterized and learnable, as it will be discussed later in Chapter 5, this KL may not be interpreted as a regularizer.

The goodness of the amortized variational inference is mainly characterized by two factors: a) the capacity of the variational distribution to represent an approximation of the intractable true posterior and b) the flexibility of the encoder to properly parameterize the variational distribution of each observation (Cremer et al., 2018).

3.2.4. The reparameterization trick

The naïve gradient estimator of the ELBO in (3.10) would suffer from high variance. Typically, the variational posteriors place the probability mass in much smaller regions than the prior does, i.e. they are peaky distributions. To compute the gradients over ϕ, the backprop algorithm would firstly compute gradients over $\boldsymbol{z}$, and due to the stochasticity of $\boldsymbol{z}$ and the peaky posterior distribution, gradients with high variance would be backpropagated. Mathematically, a simple MC estimator of the gradients would be derived as

$$\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = \mathbb{E}_{q_\phi(\mathbf{z})}\left[f(\mathbf{z})\nabla_{q_\phi(\mathbf{z})}\log q_\phi(\mathbf{z})\right] \simeq \frac{1}{L}\sum_{l=1}^{L} f(\mathbf{z})\nabla_{q_\phi(\mathbf{z}^{(l)})}\log q_\phi\left(\mathbf{z}^{(l)}\right), \tag{3.11}$$

where the gradient $\nabla_{q_\phi(\boldsymbol{z}^{(l)})}$ would be the one referred to suffer from high variance. To solve that, Kingma and Welling (2013) proposed an efficient solution referred to as the **reparameterization trick**, consisting on firstly sampling an auxiliary variable $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and secondly obtaining $\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{x})$ by reparameterizing

$$\boldsymbol{z} = \boldsymbol{\mu}_z + \boldsymbol{\sigma}_z \cdot \boldsymbol{\epsilon}. \tag{3.12}$$

Thanks to this trick, the randomness comes from an independent source of the parameters to be optimized, and thus, the variance of the gradients is drastically reduced. A related technique will be employed in the second contribution of this thesis, discussed in Chapter 5, for relaxing the posterior distribution of a hierarchical latent space and efficiently incorporating a HMC sampler.

3.3. Challenges in VAEs

As discussed above, VAEs are a highly flexible and powerful type of model that allows for arbitrary encoder and decoder architectures without requiring neural networks to be invertible. Unlike ARMs, Flows or Diffusion-based models, VAEs can learn low-dimensional data representations and we can control the dimensionality of the latent space. However, they are not without their issues. In addition to the previously mentioned challenges, such as requiring efficient integral estimation and a gap between the ELBO and the log-likelihood function for overly simplistic variational posteriors, there are several other potential problems to consider.

Posterior collapse

One issue is related to the regularization term and the ELBO. In case the decoder is too powerful and treats $\boldsymbol{z}$ as noise, the regularization term will be minimized such that the posterior and prior are too close, leading to uninformative posterior, or the **posterior collapse** problem (Alemi et al., 2018). Within this problematic, the decoder acts as an ARM, combining dimensions of $\boldsymbol{z}$ to reconstruct data.

Numerous concepts have been put forth to address the issue of posterior collapse. One such concept, as outlined in (He et al., 2019), involves updating variational posteriors more frequently than the decoder. In (Dieng et al., 2019), an alternative decoder architecture was suggested, featuring skip connections that promote a more seamless flow of information (and gradients) within the decoder.

As it will be discussed later, Hierarchical VAEs are prone to suffer from the posterior collapse problem, in case deep layers in the latent hierarchy are ignored and act as noise. Some recent works addressed this problem by adding top-down inference paths (Maaløe et al., 2019; Vahdat and Kautz, 2020; Sønderby et al., 2016; Child, 2020).

The holes problem

This occurs when the aggregated posterior does not match the prior, resulting in regions where the prior assigns high probability but the aggregated posterior $q_\phi(\boldsymbol{z}) = \frac{1}{N} q_\phi(\boldsymbol{z}|\boldsymbol{x}_n)$ assigns low probability, or vice-versa. Consequently, sampling from these regions, named *holes*, produces unexplored latent values and unrealistic or poor quality images from the decoder. This phenomena is referred to as the **hole problem** (Rezende and Viola, 2018), and is depicted in Figure 3.4. Typically this problem is handled by increasing the flexibility of the prior by employing learnable distributions, as it will be discussed in Section 3.4.

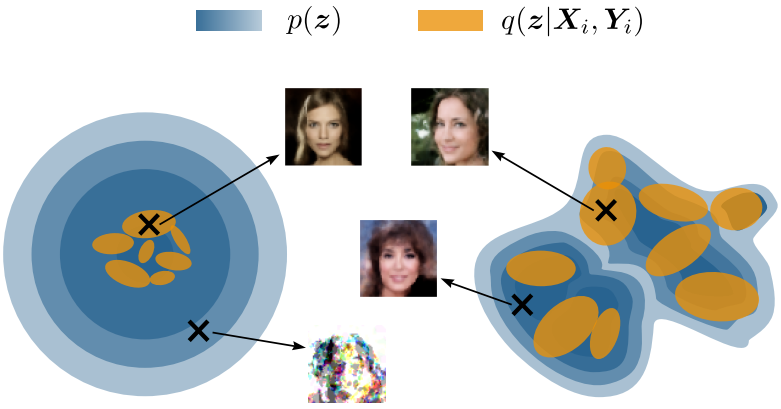


Figure 3.4: Illustration of the *hole problem*. Blue contours are the prior $p(\boldsymbol{z})$, whilst orange contours are approximate posteriors $q_\phi(\boldsymbol{z}|\boldsymbol{x}_n, \boldsymbol{y}_n)$. Left: simple standard prior does not accurately cover the encoder complexity. Decoding samples from the prior that fall far from the aggregated posterior from training data gives unrealistic images. Right: a more flexible prior properly matches the complexity of the encoder, leading to better quality of the images generated from the prior. Figure from (Koyuncu et al., 2023).

Outlier Detection

Variational Autoencoders (VAEs) are not typically well-suited for outlier detection because they are designed to model the underlying distribution of the training data, rather than explicitly identifying outliers (Chauhan et al., 2022). By evaluating the likelihood, VAEs can be employed for anomaly detection. However, their training assumes that the occurrence of the data is representative of the underlying distribution, and any data that falls outside of that distribution is likely to be poorly reconstructed by the decoder (Denouden et al., 2018). While VAEs can identify points that are dissimilar to the training data, they are not optimized specifically for this task. In other terms, VAEs can calculate likelihoods which may be useful for identifying outliers when dealing with unlabeled data. However, prior research (Chauhan et al., 2022; Denouden et al., 2018; Xiao et al., 2020; Serrà et al., 2019) has revealed that these likelihoods are not dependable and can be easily skewed by simple modifications to the input data. Furthermore, as said before, VAEs can suffer from the **hole problem**, where there are regions of the latent space that are not well-represented by the training data. This can make it difficult to distinguish between outliers and regions of the latent space that the model has not encountered during training.

In summary, despite VAEs can be used for outlier detection, they may not be the best choice of model for this task and may require additional modifications or use in combination with other models to effectively identify outliers.

MC inference

Advanced MC methods have been recently proposed for enhancing the approximate inference in VAEs, mainly by improving the approximation of the log-evidence (Burda et al., 2015; Salimans et al., 2015), its gradients (Ruiz et al., 2021; Caterini et al., 2018), or

with similar benefits, the quality of the samples from the approximate posterior (Campbell et al., 2021).

This direction is directly related with one of the contributions of this thesis (Peis et al., 2022), presented in Chapter 5. Therefore, it is more extensively discussed in a separate Section 3.5 below and in the mentioned chapter.

Representation Learning

In the context of deep generative models, Representation Learning is the Machine Learning area that studies the usefulness of learning latent representations. By designing the structure of the latent space, meaningful representations can be obtained with the help of inductive bias. This definition is specified to the relationships between input data and latent factors that represent generative factors of variation (Mathieu et al., 2019b; Locatello et al., 2019b) within unsupervised or semi-supervised settings (Bouchacourt et al., 2018).

As stated before, this direction is directly related with one of the contributions of this thesis (Peis et al., 2023), presented in Chapter 4. Therefore, it is more extensively discussed in a separate Section 3.8 below and in the mentioned chapter.

Variational design

An important research direction in VAEs is based on enhancing the flexibility of the encoder, in order to obtain better approximations of the true posterior. The most widely employed variational approximation, i.e. a factorize (diagonal covariance) Gaussian, might be insufficient for modeling the posterior high-dimensional latent spaces of complex data.

Of special importance among the alternatives to parameterize the approximate posterior is the employment of Normalizing Flows. As stated before in Section 2.4.2, the main advantage of Flows is that they are invertible, thus they allow for exact inference. The way they operate is by starting from a simpler, factorized Gaussian distribution, to be transformed into more complex distributions by means of invertible transformation. Using the determinant of the Jacobian of (2.37), and considering a group of L Flow layers, the ELBO under this setup can be expressed as

$$\log p(\boldsymbol{x}) \geq \mathbb{E}_{q(\boldsymbol{z}^{(0)}|\boldsymbol{x})}\left[\log p(\boldsymbol{x}|\boldsymbol{z}^{(L)}) + \sum_{l=1}^{L} \log \left|\frac{\partial \boldsymbol{f}^{(l)}}{\partial \boldsymbol{z}^{(l-1)}}\right|\right] - D_{\mathrm{KL}}(q(\boldsymbol{z}^{(0)}|\boldsymbol{x})||p(\boldsymbol{z}^{(L)})) \quad (3.13)$$

Different types of Flow-based transformation have been considered, being the most important ones the Inverse Autoregressive Flow (Kingma et al., 2016), Householder flows (Tomczak and Welling, 2016), Sylverster flows (Berg et al., 2018) or generalized Sylvester flows (Hoogeboom et al., 2020).

Using bigger architectures

As stated before, once the variational distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is chosen, any architecture can be utilised for parametering it, and similarly for the data likelihood distribution $p_\theta(\boldsymbol{x}|\boldsymbol{z})$. ARMs (Van Den Oord et al., 2016; Van den Oord et al., 2016; Salimans et al., 2017), ResNets Vahdat and Kautz (2020); Child (2020) or Transformers (Tang and Matteson, 2021b) are powerful examples that have been recently considered. The balance needed between encoder and decoder architectures is studied in (Cremer et al., 2018).

Prior design

By designing more flexible priors, deep generative models achieve a higher expressivity and can learn to generate more complex data. Several degrees of flexibility have been considered, starting from mixtures of Gaussians (Dilokthanakul et al., 2016; Tomczak and Welling, 2018), to more complex models like Flow-based (Koyuncu et al., 2023; Chen et al., 2017b; Gatopoulos and Tomczak, 2021), Diffusion-based (Zeng et al., 2022), or Hierarchical VAEs (Vahdat and Kautz, 2020; Child, 2020; Maaløe et al., 2019).

This latter direction is directly related with one of the contributions of this thesis, presented in Chapter 5. Therefore, it is more extensively discussed in a separate Section 3.4 below and in the mentioned chapter.

Geometry of the latent space

The Euclidean space is the typical choice for the latent space. Nonetheless, the VAE framework enables exploration of alternative spaces. As examples, a hyperspherical latent space was employed in (Davidson et al., 2018; 2019), and a hyperbolic latent space was utilized in (Mathieu et al., 2019a).

Discrete latent spaces

Although only continuous latent variables have been considered up to this point in the text for VAEs, recently, potential studies have incorporated discrete latent variables for generating data. From simpler Categorical variable that models components of mixture models (Dilokthanakul et al., 2016; Peis et al., 2023), to more flexible varieties of VAEs that utilizes vector quantization for obtaining a discrete latent representation, like the VQ-VAE (Van Den Oord et al., 2017). The introduction of vector quantization techniques enables to avoid the issue of posterior collapse mentioned in Section 3.3. By combining these representations with an autoregressive prior, the model can generate high-quality images, videos, speech, and perform top-notch speaker conversion and unsupervised phoneme learning.

One of the contributions of this thesis is an example of using discrete latent variables in VAEs (Peis et al., 2023). Concretely, the considered discrete latent variable models the components of a Gaussian Mixture, that allows for achieving disentanglement of global and local latent spaces, as it will be discussed later in Chapter 4.

3.4. Prior design

The ELBO encourages to properly reconstruct data via the first term, while minimizing mismatch between posterior and prior via the second term. In the base definition, the prior $p(\boldsymbol{z})$ is modeled by a standard Gaussian. Nevertheless, when complex data spaces are to be encoded in the latent space, more flexible priors are required in order to avoid a big mismatch between the aggregated posterior and the prior. In previous works, this issue has been alleviated by several strategies, among others: using a Gaussian Mixture prior (Dilokthanakul et al., 2016), using multimodal priors mimicking the aggregated posterior (VampPrior, Tomczak and Welling (2018)), or training flow-based (Rezende and Mohamed, 2015a; Kingma et al., 2016; Papamakarios et al., 2021; Gatopoulos and Tomczak, 2021), autoregressive (Chen et al., 2017a), hierarchical (Klushyn et al., 2019; Maaløe et al., 2019; Peis et al., 2022), or diffusion-based (Zeng et al., 2022) priors.

3.4.1. Standard Gaussian prior

In VAEs, the standard Gaussian distribution $p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ is the vanilla option for the prior over the latent variables. The main reason is because it is uncomplicated and requires no additional parameters. Nonetheless, as discussed previously, using the standard Gaussian distribution can result in poor hidden representations with *holes* due to the mismatch between the aggregated posterior and the prior (see Section 3.3). This problematic worsens with higher dimensional data. Thus, more flexible priors are required, typically those with learnable parameters. In the following subsections, recent approaches are briefly discussed.

3.4.2. Mixture of Gaussians prior

The Gaussian Mixture Model as prior of a VAE was firstly proposed by (Dilokthanakul et al., 2016) as the GMVAE. Thanks to this approach, the latent space flexibility can be easily enhanced. For example, when training a GMVAE with MNIST, the approximate posterior of $\boldsymbol{z}$ is automatically clusterized to separate codes of different digist in different clusters or regions in the latent space. This is achieved by defining a latent variable d with Categorical prior $p(d) = \text{Cat}(\boldsymbol{\pi}_d)$, where in its simplest approach, $\boldsymbol{\pi}_d$ is defined to make the mixture uniform, and the posterior is approximated by a new variational proposal $q_{\phi_d}(d) = \text{Cat}(\hat{\boldsymbol{\pi}}_d)$, where the parameters $\hat{\boldsymbol{\pi}}_d$ are given by a NN.

VampPrior

Of special consideration is the Variational Mixture of Posterior Prior (VampPrior, Tomczak and Welling (2018)), where an approximation of the aggregated posterior is considered as the prior of the model. Considering a set of *pseudo-inputs*, i.e. learnable points in the observational space $\boldsymbol{u}_k \in \mathcal{X}^D$, the learnable prior is expressed as

$$p_\lambda(\boldsymbol{z}) = \frac{1}{K} \sum_{k=1}^{K} q_\phi(\boldsymbol{z}|\boldsymbol{u}_k) \tag{3.14}$$

where $\lambda = \{\phi, \boldsymbol{u}_k\}$ are the set of prior parameters. In (Alemi et al., 2018), a generalized approach by also learning the weights of the mixture was presented,

$$p_\lambda(\boldsymbol{z}) = \sum_{k=1}^{K} w_k q_\phi(\boldsymbol{z}|\boldsymbol{u}_k) \tag{3.15}$$

with an information-theoretic perspective of the VAE to select relevant pseudo-inputs for defining the prior. In both works, author demonstrate the effectiveness in covering the posterior and reducing the *hole* effect.

3.4.3. Flow-based prior

Flows are also adequate for modeling the prior of latent variables in VAEs. By sampling from the initial distribution $p(\boldsymbol{z}^{(0)})$, typically from a standard Gaussian, successive layers transform to $p_{\theta_z}(\boldsymbol{z}^{(L)})$ with learnable parameters θ_z. These parameters are learned via the KL term in the ELBO,

$$\mathcal{L}(\boldsymbol{x}; \theta, \phi) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})] - D_{\text{KL}}(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \| p_{\theta_z}(\boldsymbol{z}^{(L)})). \tag{3.16}$$

In Flow terminology, this KL is typically referred to as the *reverse KL*, since the target distribution is known. Examples of works using flows as the prior of VAEs are (Chen et al., 2017a; Gatopoulos and Tomczak, 2021).

As an example, in a coauthored work published at the final stage of this thesis, (Koyuncu et al., 2023), where a VAE framework was designed to generate weights of a neural network, the posterior model was of such complexity that using a standard prior leaded to unrealistic samples, decoded from holes of the latent space. This problem, explained above in Section 3.3, was solved by incorporating a Real NVP (Real-valued, Non-volume Preserving) Flow, mainly a Flow that incorporates affine coupling layers.

3.4.4. Diffusion-based prior

Recently, diffusion-based models have been proposed for incorporating extra flexibility in the prior of the latent variables. In (Zeng et al., 2022), a hierarchical VAE for generating 3D point clouds is pretrained using a standard prior $p(\boldsymbol{z})$, and during a second stage, a Diffusion-based model is trained for denoising $p(\boldsymbol{z}^{(0)})$ to $p_{\theta_z}(\boldsymbol{z}^{(T)})$ to match the learned (approximate) posterior.

3.4.5. Hierarchical prior

Hierarchical priors are inherent to Hierarchical VAEs, where a group of L latent variables $\{\boldsymbol{z}_1, ...\boldsymbol{z}_L\}$ are hierarchically organized such that, starting from the deepest $p(\boldsymbol{z}_L) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, each layer generates the prior of the next one in the hierarchy, following the autoregressive procedure $p(\boldsymbol{z}_l|\boldsymbol{z}_{l+1}) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{z}}(\boldsymbol{z}_{l+1}),\ \boldsymbol{\sigma}_z^2(\boldsymbol{z}_{l+1}) \cdot \boldsymbol{I})$, where $\boldsymbol{\mu}_{\boldsymbol{z}}(\cdot)$ and $\boldsymbol{\sigma}_z^2(\cdot)$ account for functions modeled by Neural Networks. Parameterizing the prior in this way leads to a highly interpretable manner in which information flows from the latents to the observational space, similarly to the way information is organized in the real world. Figure 3.6 illustrates this fact. Recent works like (Child, 2020; Vahdat and Kautz, 2020; Maaløe et al., 2019; Sønderby et al., 2016) have shown astonishing results using these hierarchical latent spaces in image generation.

In the second principal contribution of the present thesis (Peis et al., 2022), presented in Chapter 5, Hierarchical VAEs are enhanced with Hamiltonian Monte Carlo, addressing all the problems that appear when sampling from such complex autoregressive densities.

3.5. Advanced inference methods in VAEs

As discussed before, inference of the latent variable distributions is required for performing ML optimization in VAEs. Due to the usage of NNs for parameterizing the distributions, computation of the evidence is intractable, therefore, an approximation is required, being Amortized Variational Inference the typical choice (Kingma and Welling, 2013). Within this method, reparameterized samples from the approximate posterior are decoded for approximating the ELBO via Monte Carlo. The variational inference approximation introduces bias to the objective (Cremer et al., 2018), as depicted in Figure 3.3.

A group of recent works put effort in improving the quality of this approximated objective. To list some recent examples, in (Rezende et al., 2014; Burda et al., 2015), the following importance weighting estimator is proposed as an alternative for the ELBO

$$\log p(\boldsymbol{x}) \geq \log \frac{1}{K} \sum_{s=1}^{S} \frac{p(\boldsymbol{x}, \boldsymbol{z}_s)}{q(\boldsymbol{z}_s|\boldsymbol{x})} \tag{3.17}$$

where $\boldsymbol{z}_s$ is a sample from the approximate posterior $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. Differently from the ELBO of (3.10), the logarithm is computed outside the expectation. Using a sufficiently large number of samples S leads to accurate estimations of the log-likelihood, like $S = 512$ for the MNIST dataset. In (Mattei and Frellsen, 2019), authors adapt this model, named Importance Weighted Autoencoder (IWAE, Burda et al. (2015)), to handling incomplete data, as it will be discussed in next Section 3.6.

Another important approach is to leverage MCMC methods for obtaining better samples from the posterior. By defining $q^{(0)}(\boldsymbol{z}|\boldsymbol{x})$ as the initial proposal, obtaining $q^{(T)}(\boldsymbol{z}|\boldsymbol{x})$ after T cycles from a MCMC method gives samples $\boldsymbol{z}^{(T)}$ that more accurately follow the true posterior. For instance, in (Salimans et al., 2015), authors opened the direction for merging variational inference and MCMC methods. They propose to modify the ELBO using

$$\begin{aligned}\log p(\boldsymbol{x}) \geq \mathbb{E}_q \left[\log p\left(\boldsymbol{x}, \boldsymbol{z}_T\right) - \log q\left(\boldsymbol{z}_0, \ldots, \boldsymbol{z}_T \mid x\right)\right. \\ \left. + \log r\left(\boldsymbol{z}_0, \ldots, \boldsymbol{z}_{t-1} \mid x, z_T\right)\right],\end{aligned} \tag{3.18}$$

where $r(\cdot)$ are proposed auxiliary inference functions, with a Markov structure, for accounting for the distributions at the intermediate steps of a HMC chain. In (Wolf et al., 2016), authors achieve to guaranteeing the asymptotic convergence to the true posterior by incorporating the acceptance step of the HMC algorithm. In (Caterini et al., 2018), they show how to optimally select the needed reverse kernels and, by making use of Hamiltonian Importance Sampling (HIS) (Neal, 2005), they achieve low-variance unbiased estimators of the ELBO and its gradients using the reparameterization trick. In (Ruiz et al., 2021), an efficient unbiased estimator is proposed for directly approximating the gradients of the true log-likelihood, obtaining VAEs fitted with improved predictive performance. Their estimator is based on an improved version of Iterated Sampling Importance Resampling (ISIR, Andrieu et al. (2010)).

In (Campbell et al., 2021), Hamiltonian Monte Carlo is proposed due to its efficiency and robustness for sampling from high-dimensional targets. The authors plug the HMC sampler once the VAE is pretrained within a first stage using the ELBO. In a second stage, the encoder is optimized for maximizing the ELBO, with the aim at getting a proper initial proposal. The decoder, and the hyperparameters of HMC, are jointly optimized using the target

$$\mathcal{L}_{HMC}(\boldsymbol{x}) = \mathbb{E}_{q_\phi^{(T)}(\boldsymbol{z})}[\log p_\theta(\boldsymbol{x}, \boldsymbol{z}^{(T)})]. \tag{3.19}$$

To avoid overfitting to high density regions, a regularizer is proposed by inflating the variance of the initial proposal with a tunable scaling parameter s. This parameter is tuned by minimizing a well-posed discrepancy measure, named Sliced Stein Kernelized Discrepancy, (SKSD, Gong et al. (2020)) that approximates the mismatch between the true posterior $p(\boldsymbol{z}|\boldsymbol{x})$ and the implicit distribution $q^{(T)}(\boldsymbol{z}|\boldsymbol{x})$. The choice of this discrepancy is motivated by its robust behavior in high dimensional spaces, when compared to the KSD (Liu et al., 2016). Further, it only requires samples from the approximate posterior (provided by HMC) and gradients of the unnormalized true posterior.

$$\mathcal{L}_{SKSD}(\boldsymbol{x}) = \text{SKSD}\left(q_\phi^{(T)}(\boldsymbol{z}|\boldsymbol{x}; \boldsymbol{s}),\, p(\boldsymbol{z}|\boldsymbol{x})\right) = f(\boldsymbol{z}^{(T)}, \nabla_{\boldsymbol{z}} p(\boldsymbol{z}, \boldsymbol{x})), \tag{3.20}$$

A similar strategy is employed in one of the main contributions of this thesis (Peis et al., 2022), presented in Chapter 5. This method is improved and generalized to work with hierarchical densities $\boldsymbol{z}_{1:L}$, solving important issues associated with sampling from autoregressive densities via HMC by using a novel reparameterization technique.

3.6. VAEs for heterogeneous incomplete data

As discussed before, VAEs are successful deep learning methods for generating structured data by encoding in a (ideally meaningful) latent space, and decoding the latent samples into the observational data. Typically, observed data is modeled as a multidimensional vector, hence, the likelihood is measured by multidimensional probability distributions. These distribution assume that every dimension of $\boldsymbol{x}$ belongs to the same data type, which is far from what occurs in real datasets.

Unfortunately, databases often store and organize data that are noisy, **heterogenous**, and **incomplete**. Continuous data, such as real-valued, positive or negative variables, or discrete data, like binary, categorical or ordinal variables, need to be differently handled. For instance, a video-on-demand platform might have incomplete and heterogenous data about its users, such as their watched films, which are marked as favourite, their gender, age, their time using the platform, etc. Likewise, Electronic Health Records of hospitals may have different lab measurements and data from diverse specific studies about their patients. Generative models that can learn the distribution and the hidden structure of such heterogenous and incomplete datasets could help us to understand the data better, fill in missing or corrupted values more robustly, identify outliers and make predictions on new data (Valera and Ghahramani, 2017).

With similar importance, handling **incomplete data** is an essential aspect of machine learning. Any dataset can be affected by missing patterns, caused by a variety of factors such as sensor malfunction, human error, or incomplete responses from survey participants. If left unaddressed, missing data can lead to biased or inaccurate machine learning models, resulting in poor predictions and flawed decision-making (Little and Rubin, 2019). However, by effectively handling missing data, machine learning models can make better use of available information, resulting in more accurate predictions and better decision-making. Properly addressing missing data is critical to ensure the reliability and generalizability of machine learning models, which can have far-reaching implications for a variety of fields (Graham, 2012; Van Buuren, 2018).

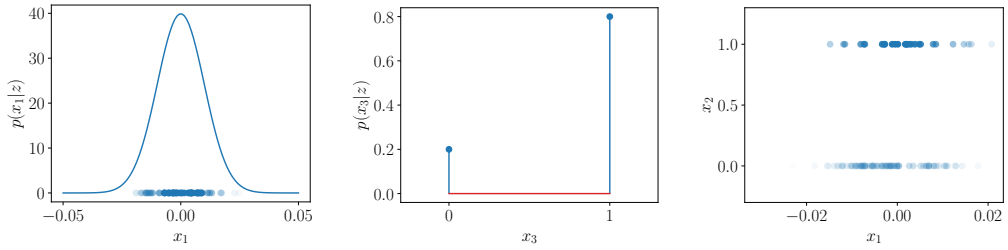
Recent works have attacked the issue of modeling heterogeneous data by proposing several strategies. In the following section, the most typical approach of factorizing over dimensions of the likelihood, is presented. Afterwards, a more robust approach of including a hierarchy of latent variables for getting more balanced likelihoods is explained. This last strategy is the one employed in the second main contribution of the thesis (Peis et al., 2022), presented in Chapter 5.

3.6.1. Likelihood factorization

This first method for adapting VAEs to handling heterogeneous incomplete data is proposed in (Nazabal et al., 2020), and further utilized in other related work (Ma et al., 2018; 2020; Mattei and Frellsen, 2019). First, to account for heterogeneous data, the following factorization of the data likelihood per dimension is proposed

$$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{z})\, p_\theta(\boldsymbol{x}|\boldsymbol{z}) = p(\boldsymbol{z}) \prod_{d=1}^{D} p_{\theta_d}(x_d|\boldsymbol{z}). \tag{3.21}$$

Thanks to this factorization, different data distributions per dimension can be utilized, being parameterized separately by NNs with parameters θ_d. For instance, if $D = 2$ and we have real-valued and binary dimensions, θ_d would be learned to parameterize the corresponding univariate Gaussian and Bernoulli distributions.



(a) $p(x_1|\boldsymbol{z}) = \mathcal{N}(0, 0.01^2)$ (b) $p(x_2|\boldsymbol{z}) = Ber(p = 0.8)$ (c) $(x_1, x_2) \sim p(x_1|\boldsymbol{z}) \cdot p(x_2|\boldsymbol{z})$

Figure 3.5: Univariate factorized likelihoods of an example with $D = 2$ variables, being $p_{\theta_1}(x_1|\boldsymbol{z})$ Gaussian (a) and $p_{\theta_2}(x_2|\boldsymbol{z})$ Bernoulli (b) distributions. Whist in (a) the mode is near 40, in (b) the maximum possible value is 0.8. Samples from the factorized likelihood $p_{\theta_1}(x_1|\boldsymbol{z}) \cdot p_{\theta_2}(x_2|\boldsymbol{z})$ are included in (c). These differences provoke unbalanced optimization when considering the likelihood factorization approach of Nazabal et al. (2020).

Although this factorization allows for easily handling the different data types, one considerable issue needs to be addressed. The values of the pdf from continuous data distributions and the pmf of discrete distributions can be in different ranges. For instance, in the previous bidimensional example (Figure 3.5), if the continuous data is accurately decoded into the probability mass of a peaky Gaussian (small variance σ^2), then $p_{\theta_1}(x_1|\boldsymbol{z}) \gg p_{\theta_2}(x_2|\boldsymbol{z})$, since $p_{\theta_2}(x_2|\boldsymbol{z}) \leq 1$. A graphical interpretation of this unbalance is provided in Figure 3.5. Hence, during the optimization, the objective might focus only on accurately decoding the continuous variables. This problem is successfully addressed by (Ma et al., 2020), as it will be described in the next Section 3.6.2. This latter approach is the one utilized in one of the main contributions, as it will be discussed in Chapter 5.

Regarding the incompleteness, the above factorization of the likelihood allows for splitting the contributions of the observed and unobserved parts

$$p(\boldsymbol{x}|\boldsymbol{z}) = p(\boldsymbol{z}) \prod_{d \in \mathcal{O}} p_{\theta_d}(x_d|\boldsymbol{z}) \prod_{d \in \mathcal{U}} p_{\theta_d}(x_d|\boldsymbol{z}), \tag{3.22}$$

where $\boldsymbol{x}$ is an incomplete datapoint composed by D features, and can be expressed as a joint distribution of two variables: $\boldsymbol{x}_o$, which represents the observed features, and $\boldsymbol{x}_u$, which represents the missing features. The sets indexed by o and u are determined by a missing mechanism that varies for each datapoint. The Missing At Random (MAR) mechanism is typically used, which assumes that the missing indices are independent of the missing feature values.

Within this approach, $\boldsymbol{x}_u$ is modeled as a latent variable, leading to the following marginal likelihood

$$p(\boldsymbol{x}_o) = \log \int p(\boldsymbol{x}_o, \boldsymbol{x}_m, \boldsymbol{z}) d\boldsymbol{z}\, d\boldsymbol{x}_m. \tag{3.23}$$

which, as previously stated, is intractable. Proposing the following variational model

$$q(\boldsymbol{z}, \boldsymbol{x}_m|\boldsymbol{x}_o) = q_\phi(\boldsymbol{z}|\boldsymbol{x}_o) \prod_{d \in \mathcal{U}} p_{\theta_d}(x_d|\boldsymbol{z}), \tag{3.24}$$

which reuses the generative paths $p(x_d|\boldsymbol{z})$ for the unobserved set of features, allowing to

express the ELBO as

$$\mathcal{L}(\boldsymbol{z}) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{z})}\left[\sum_{d\in\mathcal{O}} \log p_\theta(x_d|\boldsymbol{z})\right] - D_{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{z}_O)||p(\boldsymbol{z})), \tag{3.25}$$

or in broader terms, only the observed features contribute to the ELBO for inferring the posterior and being reconstructed.

3.6.2. Two-level hierarchy for modeling mixed-type data

In a more recent method (Ma et al., 2020), an effective solution was proposed for the problem of having unbalanced factorized likelihoods. Authors propose to introduce a hierarchy of two latent variables, $\boldsymbol{z}$ and $\boldsymbol{h}$. The former is a concatenation of D univariate latent variables z_d, and consequently shares dimensionality with $\boldsymbol{x}$. Each z_d is a unidimensional Gaussian latent variable for generating the corresponding x_d by parameterizing its likelihood. The former, $\boldsymbol{h}$, is a multidimensional latent variable, of smaller dimension, that learns to generate the $\boldsymbol{z}$ vectors.

Within this setup, Ma et al. (2020) demonstrated that the model can be trained in a two stage approach, where in an initial stage, the parameters of $p_{\theta_d}(x_d|z_d)$ and $q_{\gamma_d}(z_d|x_d)$ for each dimension are learned using the following ELBO on the observed data

$$\mathcal{L}_d(x_d) = \mathbb{E}_{q_{\gamma_d}(z_d|x_d)}\left[\log \frac{p_{\theta_d}(x_d, z_d)}{q_{\gamma_d}(z_d \mid x_d)}\right], \qquad d \in \mathcal{O}. \tag{3.26}$$

Each $p_{\theta_d}(x_d|z_d)$ and $q_{\gamma_d}(z_d|x_d)$ comprises one of the named *marginal VAEs* for each feature that are trained independently. In the second stage, *the dependency VAE* is trained using the concatenated encodings, $\boldsymbol{z}$, from the learned marginal VAEs, which are all Gaussian, using the second ELBO

$$\mathcal{L}(\boldsymbol{x}) = \mathbb{E}_{q(\boldsymbol{h}|\boldsymbol{z})\prod_d q_{\gamma_d}(z_d|x_d)}\left[\log \frac{p_\theta(\boldsymbol{z}, \boldsymbol{h})}{q_\phi(\boldsymbol{z}|\boldsymbol{h})}\right], \qquad d \in \mathcal{O}. \tag{3.27}$$

Thanks to using this approach, the heterogeneous marginal properties of each dimension are learned by the corresponding *marginal VAE*, whilst the interdependencies between the features of the dataset can be captured by the *dependency VAE* in a balanced manner using Gaussian distributions.

This methodology is used for efficiently handling heterogeneous data in the second contribution of this thesis, the HH-VAEM model (Peis et al., 2022), presented in Chapter 5.

3.7. Hierarchical VAEs

The fundamental idea behind hierarchical models is that real data can often be structured hierarchically. Based on this fact, if a latent variable model is structured as a hierarchy of autoregressive latent variables, it could potentially introduce an inductive bias, limit the variety of models available, and ultimately encourage the flow of information between observable and latent variables, as discussed in previous Section 2.4.8. By using a hierarchical structure, it is enforced that the information flows from high-level representations to more specific factors (as depicted in Figure 3.6), imitating the way information is often organized in the real world. Nevertheless, when articulating

Figure 3.6: Example of information encoded in a hierarchy of latent variables. Samples generated by the VDVAE from (Child, 2020). From left ($l = L$) to right ($l = 1$), samples are obtained by using $\boldsymbol{z}_{l:L}$ sampled from the posterior, and $\boldsymbol{z}_{1:l-1}$ sampled from the prior. More abstract information about the global structure, like the skin tone or hair color, is encoded in the deepest layers of the hierarchy (left), whilst specific details of the face, hair or background are encoded in shallower layers (right).

stochastic dependencies in the hierarchy, other issues require special attention, as it will be discussed.

Mathematically, the aforementioned hierarchical generative model of L latent variables can be expressed using the following joint distribution

$$p(\boldsymbol{x}, \boldsymbol{z}_1, ..., \boldsymbol{z}_L) = p(\boldsymbol{x}|\boldsymbol{z}_1)\, p(\boldsymbol{z}_L) \prod_{l=1}^{L-1} p(\boldsymbol{z}_l|\boldsymbol{z}_{l+1}), \tag{3.28}$$

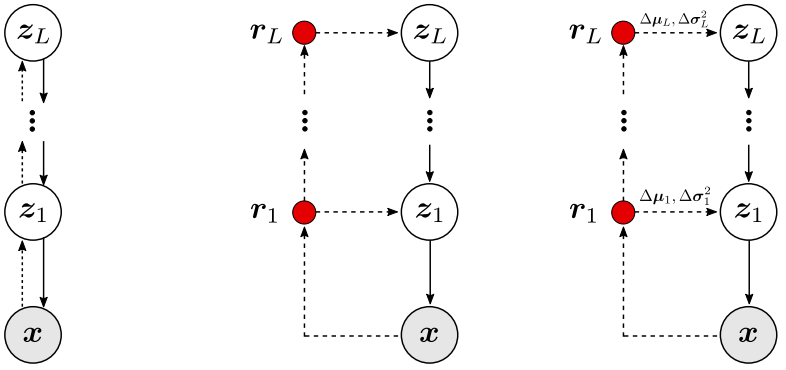
where the parameters are $\theta = \{\theta_x, \theta_{z_1}, ..., \theta_{z_L}\}$ of $p_{\theta_x}(\boldsymbol{x}|\boldsymbol{z}_1)$ and $p_{\phi_{z_l}}(\boldsymbol{z}_l|\boldsymbol{z}_{l+1})$ are omitted for the ease of convenience. For obtaining a sample, firstly $\boldsymbol{z}_L$ (typically referred to as the top or deepest variable) would be sampled from a fixed prior such as a standard normal. Secondly, the Gaussian parameters of each $p(\boldsymbol{z}_l|\boldsymbol{z}_{l+1})$ would be outputs of independent NNs to get the $\boldsymbol{z}_l$ samples. Lastly, the *bottom* latent variable, i.e. the shallowest, would be decoded using a NN into the parameters of the data distribution.

Hierarchical VAEs can be seen as a generalization of diffusion models or flow-based models. Compared with the former, they do not have the restriction of fixing the variational proposal. Regarding the latter, they do not add the limitation of using specific NNs that allow for invertible transformations. The price to pay for these degrees of freedom is a delicate inference procedure, as it will be explained in the next section.

3.7.1. Inference in Hierarchical VAEs

Within a Hierarchical VAE framework, a natural inference proposal would be to infer the variables following the reverse direction of the generation process. If the direction from deeper to shallower layers is denoted by *top-down* , and equivalently, *bottom-up* refers to the opposite direction, this first approach would be expressed mathematically as

$$q(\boldsymbol{z}_{1:L}|\boldsymbol{x}) = q(\boldsymbol{z}_1|\boldsymbol{x}) \prod_{l=2}^{L} q(\boldsymbol{z}_l|\boldsymbol{z}_{l-1}) \tag{3.29}$$



(a) Bottom-up (b) Top-down (c) Top-down residual

Figure 3.7: Variational proposals for the posterior of a Hierarchical VAE. In (a), an intuitive but ill-posed approach would lead to posterior collapse of deepest variables. In (b), this issue is relaxed by sharing the top-down stochastic path. In (c), posterior and prior are tied by learning a residual posterior.

which is also depicted in Figure 3.7a with the dashed lines. Nevertheless, this natural approach includes potential pitfalls to be considered. The ELBO within this setup is

$$\mathcal{L}(\boldsymbol{x}) = \mathbb{E}_{q(\boldsymbol{z}_{1:L}|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z}_1) \\ -D_{\mathrm{KL}}(q(\boldsymbol{z}_1|\boldsymbol{x})||p(\boldsymbol{z}_1|\boldsymbol{z}_2)) - \sum_{l=2}^{L-1} D_{\mathrm{KL}}(q(\boldsymbol{z}_l|\boldsymbol{z}_{l-1})||p(\boldsymbol{z}_l|\boldsymbol{z}_{l+1})) - D_{\mathrm{KL}}(q(\boldsymbol{z}_L|\boldsymbol{z}_{L-1})||p(\boldsymbol{z}_L))]. \tag{3.30}$$

When the weights of the NNs that parameterize the model are randomly initialized, the parameterized Gaussian distributions result to be standard. In case the decoder $p(\boldsymbol{x}|\boldsymbol{z}_1)$ is sufficiently flexible, the optimization can focus on easily minimizing the last KL terms for variables $\boldsymbol{z}_{2:L}$ by making the approximate posterior uninformative, or mathematically $q(\boldsymbol{z}_l|\boldsymbol{z}_{l-1}, \boldsymbol{x}) \approx \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ with $l > 1$. In other terms, the layers on top of the first one would be ignored, acting as noise. This would result on a waste of parameters. The same analysis extends for the cases when the information flows bottom-up, ending in any intermediate layer, and ignoring the rest of layers on top.

Top-down inference

To solve the aforementioned inference issues for Hierarchical VAEs, recent works have proposed an alternative way of performing a better-posed inference consisting on following the top-down generative path, from deeper to shallower variables, also in the variational proposal. Mathematically, the inference model would be given by

$$q(\boldsymbol{z}_{1:L}|\boldsymbol{x}) = q(\boldsymbol{z}_L|\boldsymbol{x}) \prod_{l=1}^{L-1} q(\boldsymbol{z}_l|\boldsymbol{x}, \boldsymbol{z}_{l+1}), \tag{3.31}$$

where a dependency with the top variable $\boldsymbol{z}_{l+1}$ is introduced. This model is depicted in Figure 3.7b. A deterministic bottom-up path would be obtained by NNs that output the variables $\boldsymbol{r}_{1:L}$ from $\boldsymbol{r}_{l-1}$, being $\boldsymbol{r}_0 = \boldsymbol{x}$. These variables, jointly with a sample from the top $\boldsymbol{z}_{l+1}$, would be fed to a NN that outputs the parameters of $q(\boldsymbol{z}_l|\boldsymbol{x}, \boldsymbol{z}_{l+1})$.

This methodology is originally inspired by ResNet VAEs (Kingma et al., 2016) and Ladder VAEs (Sønderby et al., 2016), and later adapted in highly-relevant VAEs like BIVA (Maaløe et al., 2019), NVAE (Vahdat and Kautz, 2020) or VDVAE (Child, 2020).

Of great importance is to highlight in this point of the present thesis that, although this inference method is proven to be highly effective for Hierarchical VAEs, it is also based on a lower bound on the log-evidence, due to the Gaussian approximation proposed by the variational distribution. One of the contributions of this thesis (Peis et al., 2022), presented in Chapter 5 is to develop a novel inference methodology for Hierarchical VAEs based on advanced MCMC-based methods like Hamiltonian Monte Carlo that outperform variational inference.

Model flexibility vs inference bias

As it has been discussed, one of the most brilliant research directions for VAEs is discovering more expressive models, i.e. by proposing flexible priors, employing bigger and more complex neural networks for encoder and decoder, or adding hierarchies of latent variables for introducing useful inductive bias and expressiveness.

Nonetheless, the problem of balancing modeling flexibility and reducing bias in approximate inference is often a complex and interconnected issue, and addressing both simultaneously in a *unified* approach can be exceedingly challenging. The intricate organization of the latent variables gives rise to complex posterior dependencies that are not easily manageable and require special attention. Further, the bias introduced by the variational approximation suffers from the *curse of dimensionality*, i.e. it is more a more considerable when dimensionality increases.

For these reasons, improving the accuracy of approximate inference in these type of models is crucial for leveraging their interesting properties. In Chapter 5, where the second main research contribution of this thesis is presented, an example of a flexible model with accurate is provided: the HH-VAEM. A hierarchical VAE is employed for enforcing a natural flow of information from latents to data, and the bias due to the Gaussian approximate posteriors for applying Variational Inference are overcame by using a robust Hamiltonian Monte Carlo sampler.

3.8. Representation learning in VAEs

As discussed previously in Section 2.4.8, the problem of learning useful latent representations with latent variable models can be challenging. By selecting a proper class of models, i.e. designing the structure of the latent space, inductive bias can be added in a way that meaningful representations can be obtained. In the context of generative models, the Machine Learning area that study this usefulness is named **Representation Learning** (Bengio et al., 2013; Eastwood and Williams, 2018; Higgins et al., 2018). Previously, representation learning was defined as the topic in Deep Learning that studies the identification of hidden representations to automatically capture useful information from the input. Here, this general definition is more specified to the relationships between input data to be learned from, and latent factors that represent generative factors of variation, within unsupervised or semi-supervised settings. Although the field of Representation Learning is far from specific to VAEs and deep generative models (Hyvärinen and Oja, 2000; Yang and Amari, 1997), for the scope of this thesis, the discussion is focused on VAEs.

The *usefulness* of the latent space is strongly related with the concept of *disentanglement*, explained in the next section.

3.8.1. The concept of Disentanglement

In the context of deep generative models, **disentanglement** (Locatello et al., 2019b; Mathieu et al., 2019b) refers to the ability of the model to learn a representation of the input data that separates the underlying **factors of variation**. This means that the model should be able to identify and isolate individual components of the data that are responsible for different aspects of the data, such as shape, color, texture, or pose in image data. Disentanglement is defined by Eastwood and Williams (2018) as the ability of a latent dimension $d \in D$ to predict a genuine generative factor $k \in K$, with each latent dimension being capable of capturing at most one generative factor. In a disentangled representation, each dimension z_i of the learned representation $\boldsymbol{z}$ corresponds to a separate factor of variation, and changing one dimension affects only that factor and not others. For example, in an image of a face, a disentangled representation would have separate dimensions for the identity of the person, their facial expression, or the lighting conditions.

The goal of disentangled representation learning is to create more interpretable and controllable generative models that can be used for tasks such as image editing, style transfer, and data augmentation. Disentangled representations can also be useful in transfer learning, where a model trained on one dataset can be adapted to a related dataset by only changing the dimensions corresponding to the differences between the datasets. Achieving disentanglement in deep generative models is an active area of research, and there are many techniques and approaches being developed to try to improve the disentanglement of learned representations.

As stated before, the discussion will be oriented to study disentanglement in VAEs. Within these models, the motivation resides in reaching independence between the dimensions of the aggregated posterior $q_\phi(\boldsymbol{z}) \approx \sum_{n=1}^{N} q(\boldsymbol{z}|\boldsymbol{x}_n)$. In the existing body of research on disentangled VAEs, a differentiation has been made between unsupervised techniques and semi-supervised methods where the authentic generative factor values are available for a certain subset of data, as studied in (Bouchacourt et al., 2018; Kingma et al., 2014; N et al., 2017). Nonetheless, the attention of this thesis is put on a novel unsupervised setting.

Challenges in disentangled representations

The extended definitions of disentanglement typically assume that $D \geq K$, which is not always the case in real data where a low-dimensional abstraction of a complex process involving multiple factors is often learned instead. Thus, simplistic representations cannot be found for more complex datasets that require richly structured dependencies to encode the necessary information for generating high-dimensional data (Locatello et al., 2019b). Moreover, for datasets with a finite set of data points, it may be unreasonable to assume that the elements of the true generative process can be captured, as the data may not contain enough information to recover them. Even if the data does contain the necessary information, the computation required for model learning to achieve this may not be feasible.

Achieving disentanglement in VAEs is a challenging task due to several factors. One of the main difficulties is the inherent ambiguity of the concept of disentanglement itself, as it lacks a precise mathematical definition. Disentanglement is often related to the ability of a model to separate the underlying factors of variation in the data, but this can

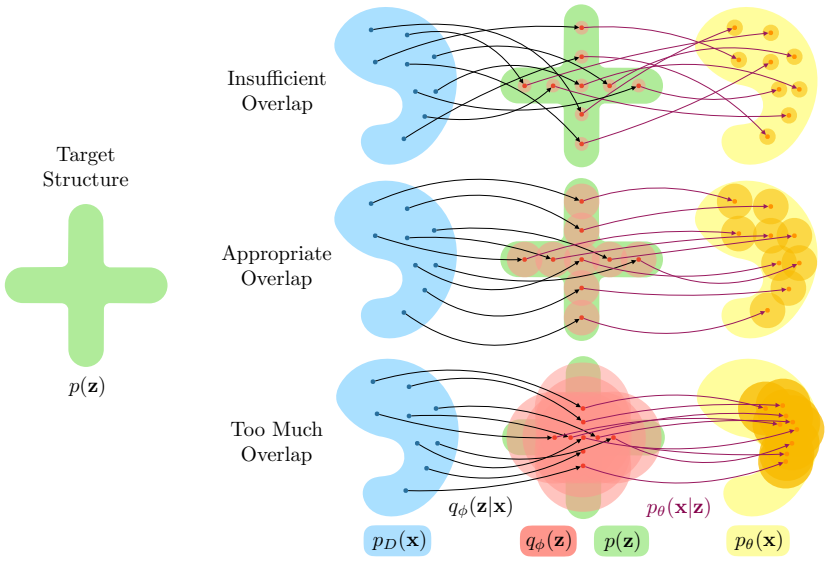


Figure 3.8: Illustration of *decomposition* from (Mathieu et al., 2019b). A cross-shaped structure is utilized as prior for ensuring sparsity and disentanglement, by having two directions of variation. On the top, latent space works as a deterministic Autoencoder. Similar data is encoded in distant latent codes in the latent space. At the bottom, the contrary case occurs. The posterior collapses and becomes uninformative, loosing expressivity. Different latent distributions are decoded into similar samples. In the middle, the ideal behavior is depicted, where the latent space representations cover the data space, are slightly overlapped and encode the generative factors into the two variations.

be subjective and depends on the application. Additionally, the optimization process of VAEs can encourage the model to use different latents for different factors of variation, but it does not guarantee that each latent will capture only one factor, leading to the so-called "collapsed" or "entangled" representations. Moreover, disentanglement requires capturing higher-order interactions between the factors of variation, which is a challenging problem for high-dimensional data. Finally, the evaluation of disentanglement is also a difficult task, as there is no universally agreed-upon metric to assess it, and different metrics may have conflicting goals.

Several recent works have shown efforts in proposing a more generalized, quantifiable definition of disentanglement. For instance, in (Mathieu et al., 2019b), authors propose a *decomposition* framework, depicted in Figure 3.8, to overcome the limitations of disentanglement. In (Locatello et al., 2019a), authors analyze the fairness of disentangled representations, observing that there is a consistent positive correlation in several models between their disentangled results and fairness (non-biased, without group discrimination, etc.) in their predictions. In (Locatello et al., 2019b), the authors argue that the standard definition of disentanglement in VAEs (i.e., each latent dimension corresponds to a single generative factor) is overly restrictive and not necessary for useful representations. They propose an alternative definition of "semantically meaningful representations" which allows for some overlap between the generative factors captured by different latent dimensions, coinciding with Mathieu et al. (2019b).

Disentanglement via semi-supervision

Several approaches have emerged (Bouchacourt et al., 2018; Esmaeili et al., 2019; Johnson et al., 2016b; N et al., 2017; Mathieu et al., 2019b) that recognize the need for richly structured dependencies among latent dimensions. These approaches employ graphical models or propose strategical objectives to overcome the limitations of disentanglement and offer more generalizable interpretations. A remarkable recent work named β-VAE (Higgins et al., 2016) introduces a hyperparameter β that controls the trade-off between reconstruction accuracy and latent channel capacity, showing that it allows for finding interpretable factors of variation in various datasets. A recent model named FactorVAE (Kim and Mnih, 2018) achieves disentanglement by encouraging a factorization across the dimensions of the latent representations.

As stated in (Locatello et al., 2019b), proper disentangled representations might not be possible to be identified without introducing any supervision. Hence, several recent work propose to efficiently incorporate inductive bias by semi-supervising potential generative factors. For instance, in (N et al., 2017), a framework is proposed for learning disentangled representations of data using VAEs with general graphical models in the encoder and decoder, that incorporate labels related to generative factors in some observations. The paper also defines a semi-supervised learning objective and an importance sampling procedure for this framework. In (Bouchacourt et al., 2018), authors propose a Multi-Level VAE (the ML-VAE) for learning two latent variables at the local and global space by grouping samples that share some generative factor, e.g. identity in face images.

This last referenced work from (Bouchacourt et al., 2018) is considered the most related work to one of the main contributions of this thesis, the **UG-VAE** model (Peis et al., 2023), extensively presented in Chapter 4. In contrast to ML-VAE, UG-VAE learns meaningful latent variables at the global level without any kind of supervision. Further, we evidence that certain degree of disentanglement is possible by carefully designing the graphical model.

CHAPTER 4

Unsupervised learning of global factors in VAEs

Since its first proposal by Kingma and Welling (2013), Variational Autoencoders (VAEs) have evolved into a vast amount of variants. To name some representative examples, we can include VAEs with latent mixture models priors (Dilokthanakul et al., 2016), adapted to model time-series (Chung et al., 2015; Bianchi et al., 2019; Fraccaro et al., 2016), trained via deep hierarchical variational families (Ranganath et al., 2016; Tomczak and Welling, 2018), with enhanced, parametric and robust priors (Tomczak and Welling, 2018; Joo et al., 2020; Van Den Oord et al., 2017), that include advanced techniques for gradient estimation (Ruiz et al., 2021; Burda et al., 2015) or that naturally handle heterogeneous data types and missing data (Nazabal et al., 2020; Ma et al., 2019a; 2020; Peis et al., 2022).

The large majority of VAE-like models are designed over the assumption that data is i.i.d., which remains a valid strategy for simplifying the learning and inference processes in generative models with latent variables. A different modelling approach may drop the i.i.d. assumption with the goal of capturing a higher level of dependence between samples. Inferring such kind of higher level dependencies can directly improve current approaches to find interpretable disentangled generative models (Bouchacourt et al., 2018), to perform domain alignment (Heinze-Deml and Meinshausen, 2017; Guerrero-López et al., 2022) or to ensure fairness and unbiased data (Barocas et al., 2017).

The main contribution presented in this chapter is to show that a deep probabilistic VAE non i.i.d. model with both local and global latent variable can capture meaningful and interpretable information among data points in a completely unsupervised fashion. Namely, weak supervision to group the data samples is not required. In the following we refer to our model as **Unsupervised Global VAE** (**UG-VAE**). We combine a clustering inducing mixture model prior in the local space, that helps to separate the fundamental data features that an i.i.d. VAE would separate, with a global latent variable that modulates the properties of such latent clusters depending on the observed samples, capturing fundamental and interpretable data features. We demonstrate such a result using both CelebA (Liu et al., 2015), MNIST (LeCun, 1998) and the 3D FACES dataset (Paysan et al., 2009) in the experimental Section 4.3. Furthermore, we show that the global latent space can explain common features in samples coming from two different databases without requiring any domain label for each sample, establishing a probabilistic unsupervised framework for domain alignment. Up to our knowledge, UG-VAE is the first VAE model in the literature that performs unsupervised domain alignment using global latent variables.

Finally, we demonstrate that, even when the model parameters have been trained using an unsupervised approach, the global latent space in UG-VAE can discriminate groups of samples with non-trivial structures, separating groups of people with black and blond hair in CelebA or series of numbers in MNIST. In other words, if weak supervision

is applied at test time, the posterior distribution of the global latent variable provides with an informative representation of the user defined groups of correlated data.

The chapter is organized as follows: in Section 4.1 we provide with a review on recent related methods for achieving disentanglement by model design or semi-supervision. In Section 4.2, the UG-VAE model is presented, extensively describing its generative and inference models and key components, as well as the final ELBO objective. Section 4.3 provides with all the experiments that evidence our contributions. Finally, in Section 4.4, we include our conclusions based on the presented results. The findings described in this chapter were published in the Pattern Recognition Journal (Peis et al., 2023).

4.1. Related work

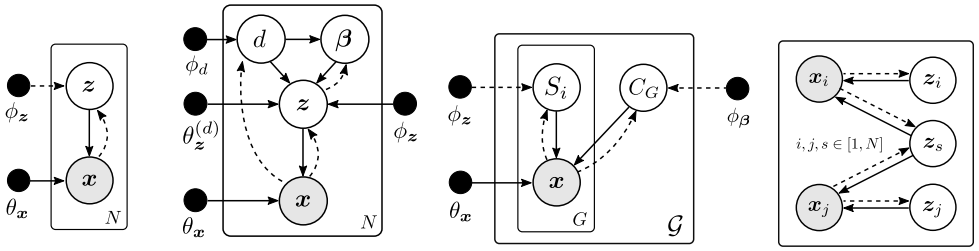
Non i.i.d. deep generative models are getting recent attention but the literature is still scarse. First we find VAE models that implement non-parametric priors: in (Gyawali et al., 2019) the authors make use of a global latent variable that induces a non-parametric Beta process prior, and more efficient variational mechanism for this kind of IBP prior are introduced in (Xu et al., 2019). Second, both (Tang et al., 2019) and (Korshunova et al., 2018) proposed non i.i.d. exchangable models by including correlation information between datapoints via an undirected graph. Third, conditional dependencies with supervised classes are modeled in (Antoran and Miguel, 2019) with the aim at performing natural clustering at the latent space and disentangle class-dependent factors. Finally, some other works rely on simpler generative models (compared to these previous approaches), including global variables with fixed-complexity priors, typically a multi-variate Gaussian distribution, that aim at modelling the correlation between user-specified groups of correlated samples (e.g. images of the same class in MNIST, or faces of the same person). In (Bouchacourt et al., 2018) or (Hosoya, 2019), authors apply weak supervision by grouping image samples by identity, and include in the probabilistic model a global latent variable for each of these groups, along with a local latent variable that models the distribution for each individual sample. In (Liu et al., 2021a), authors use co-supervision for achieving stationary state in learning graphs for multi-view clustering. Below we specify the two most relevant lines of research, in relation to our work.

4.1.1. VAEs with mixture priors

Several previous works have demonstrated that incorporating a mixture in the latent space leads to learn significantly better models. In (Johnson et al., 2016b) authors introduce a latent GMM prior with nonlinear observations, where the means are learned and remain invariant with the data. The GMVAE proposal by (Dilokthanakul et al., 2016) aims at incorporating unsupervised clustering in deep generative models for increasing interpretability. In the VAE with a VAMP prior model (Tomczak and Welling, 2018), the authors define the prior as a mixture with components given by approximated variational posteriors, that are conditioned on learnable pseudo-inputs. This approach leads to an improved performance, avoiding typical local optima difficulties that might be related to irrelevant latent dimensions.

4.1.2. Semi-supervised deep models for grouped data

In contrast to the i.i.d. vanilla VAE model in Figure 4.1 (a), and its augmented version for unsupervised clustering, GMVAE, in Figure 4.1 (b), the graphical model of the



(a) VAE (b) GMVAE (c) ML-VAE (d) NestedVAE

Figure 4.1: Comparison of four deep generative models. Dashed lines represent the graphical model of the associated variational family. The Vanilla VAE (a), the GMVAE (b), and semi-supervised variants for grouped data; ML-VAE (c) and NestedVAE (d).

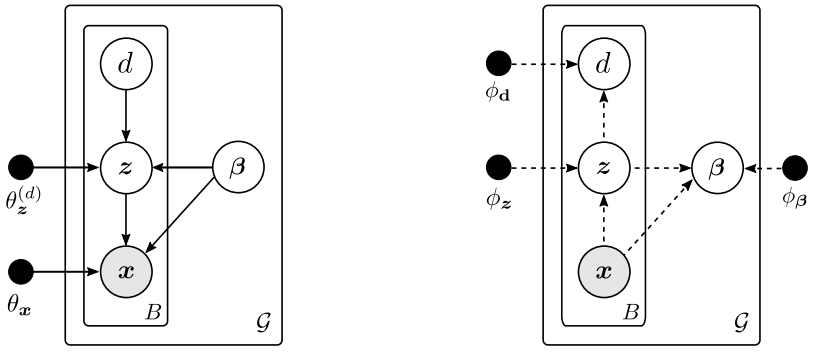
Multi-Level Variational Autoencoder (ML-VAE) in (Bouchacourt et al., 2018) is shown in Figure 4.1 (c), where G denotes the number of groups. ML-VAE includes a local Gaussian variable S_i that encodes style-related information for each sample, and a global Gaussian variable C_G is shared within a group of samples. For instance, they feed their algorithm with batches of face images from the same person, modeling content shared within the group that characterize a person. This approach leads to learning disentangled representations at the group and observations level, in a content-style fashion. Nevertheless, the groups are user-specified, hence resulting in a semi-supervised modelling approach. In (Vowels et al., 2020) authors use weak supervision for pairing samples. They implement two outer VAEs with shared weights for the reconstruction, and a Nested VAE that reconstructs latent representation off one to another, modelling correlations across pairs of samples. The graphical model for Nested VAE is depicted in Figure 4.1 (d). Despite the fact that semi-supervision is proved to improve performance for some deep generative models (Gordon and Hernández-Lobato, 2017; 2020), it requires prior knowledge about the data that we do not assume in this work.

4.2. Unsupervised Global VAE

UG-VAE is a deep generative VAE framework for modeling non-i.i.d. data with global dependencies. It generalizes the ML-VAE graphical model in Figure 4.1 (c), combining the global model with a mixture prior to *i)* remove the group supervision, *ii)* include a clustering-inducing prior in the local space, and *iii)* propose a more structured variational family. The latent discrete variable d is expected to represent the inferred group with no supervision needed.

4.2.1. Generative model

Figure 4.2 represents the generative graphical model of UG-VAE. The global variable $\boldsymbol{\beta} \in \mathbb{R}^g$ modulates the prior, inducing shared properties within a group of B samples $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_B\} \subseteq \mathbb{R}^D$, while the local variable $\boldsymbol{z}$ encodes the local properties for each datapoint. Although we use this notation for the global latent space, we would like to remark that $\boldsymbol{\beta}$ is not a parameter as the β defined in (Higgins et al., 2016). We denote by $\mathcal{G}$ the number of groups we jointly use to amortize the learning of the model



(a) Generative model (b) Inference model

Figure 4.2: Generative (left) and inference (right) of UG-VAE.

parameters. During amortized variational training, groups are simply random data mini-batches from the training dataset, being $\mathcal{G}$ the number of data mini-batches. We could certainly take $B = N$ (the training set size) and hence $\mathcal{G} = 1$, but this leads to a less interpretable global latent space (too much data to correlate with a single global realization), and a slow training process. On the contrary, a small batch size might result in highly dispersed global properties, difficult to capture and again hardly interpretable. The difficulty in choosing a proper value for the batch size limits the potential of learning useful representations, and arises in the lack of agnostic metrics for performing objective validations. Although some useful representation metrics (e.g. (Heusel et al., 2017)) could be used for validating B, we show results in Section 5.4 demonstrating that reasonable batch sizes values that are widely employed in similar works (namely $B = 128$) successfully learn disentangled global representations.

Conditioned to $\boldsymbol{\beta}$, data samples are distributed according to a Gaussian mixture local (one per data) latent variable $\boldsymbol{Z} = \{\boldsymbol{z}_1, ..., \boldsymbol{z}_B\} \subseteq \mathbb{R}^d$, and $\boldsymbol{d} = \{d_1, ..., d_B\} \subseteq \{1, ..., K\}$ are independent discrete categorical variables with uniform prior distributions. This prior, along with the conditional distribution $p(\boldsymbol{z}_i|d_i, \boldsymbol{\beta})$, defines a Gaussian mixture latent space, which helps to infer similarities between samples from different batches (by assigning them to the same cluster), and thus, d_i plays a similar role than the semi-supervision included in (Bouchacourt et al., 2018) by grouping. Our experimental results demonstrate that this level of structure in the local space is crucial to acquire interpretable information at the global space.

The joint distribution for a single group is therefore defined by:

$$p_\theta(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}) = p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta})\, p(\boldsymbol{Z}|\boldsymbol{d}, \boldsymbol{\beta})\, p(\boldsymbol{d})\, p(\boldsymbol{\beta}) \tag{4.1}$$

where the likelihood term of each sample is a Gaussian distribution, whose parameters are obtained from a concatenation of $\boldsymbol{z}_i$ and $\boldsymbol{\beta}$ as input of a decoder network:

$$p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\beta}) = \prod_{i=1}^{B} p(\boldsymbol{x}_i|\boldsymbol{z}_i, \boldsymbol{\beta}) = \prod_{i=1}^{B} \mathcal{N}\left(\boldsymbol{\mu}_{\theta_x}([\boldsymbol{z}_i, \boldsymbol{\beta}]), \boldsymbol{\Sigma}_{\theta_x}([\boldsymbol{z}_i, \boldsymbol{\beta}])\right) \tag{4.2}$$

In contrast with (Johnson et al., 2016b), where the parameters of the clusters are learned but shared by all the observations, in UG-VAE, the parameters of each component are obtained with networks fed with $\boldsymbol{\beta}$. Thus, the prior of each local latent continuous variable

is defined by a mixture of Gaussians, where d_i defines the component and $\boldsymbol{\beta}$ is the input of a NN that outputs its parameters:

$$p(\boldsymbol{Z}|\boldsymbol{d},\boldsymbol{\beta}) = \prod_{i=1}^{B} p(\boldsymbol{z}_i|d_i,\boldsymbol{\beta}) = \prod_{i=1}^{B} \mathcal{N}\left(\boldsymbol{\mu}_{\theta_z}^{(d_i)}(\boldsymbol{\beta}), \boldsymbol{\Sigma}_{\theta_z}^{(d_i)}(\boldsymbol{\beta})\right) \tag{4.3}$$

hence we trained as many NNs as discrete categories. This local space encodes samples in representative clusters to model local factors of variation. The prior of the discrete latent variable is defined as uniform:

$$p(\boldsymbol{d}) = \prod_{i=1}^{B} \text{Cat}(\boldsymbol{\pi}) \quad \pi_k = 1/K \tag{4.4}$$

and the prior over the continuous latent variable β follows an isotropic Gaussian, $p(\boldsymbol{\beta}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

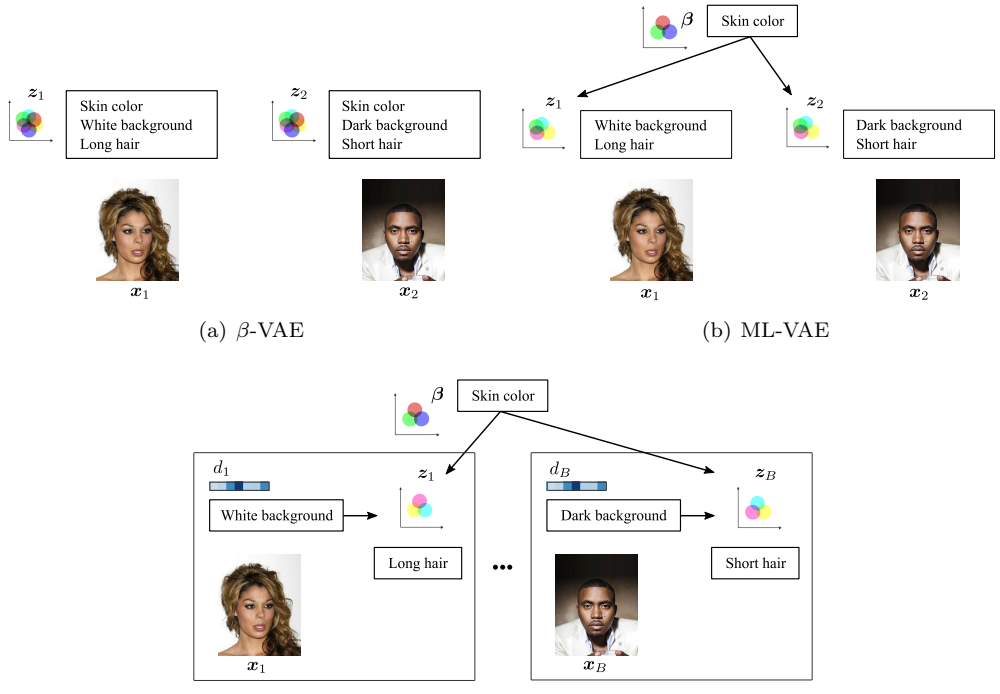


Figure 4.3: Illustration of the inductive bias introduced by the generative model in β-VAE (a), ML-VAE (b) and the proposed UG-VAE (c). β-VAE compresses all features in a single latent space, while ML-VAE uses a global component that needs to be supervised during training. In UG-VAE, we propose a flexible hierarchy of both global and local mixture of components that enables learning group-features in the latent space in an unsupervised manner.

By making use of the presented generative model, we propose a flexible hierarchy of both global and local mixture of components that, while being trained on random-mini

batches, it is able to exploit the augmented degrees of freedom for capturing group-features in the latent space in an unsupervised manner. A graphical representation of how UG-VAE structures the information in the latent space is provided in Figure 4.3. Further, as shown in Section 4.2.2, the posterior approximation results from an individual contribution of each data point that favors group separation across latent spaces.

4.2.2. Inference model

The graphical model of the proposed variational family is shown in Figure 4.2(b):

$$q_\phi(\boldsymbol{Z}, \boldsymbol{d}, \boldsymbol{\beta}|\boldsymbol{X}) = q(\boldsymbol{Z}|\boldsymbol{X})\, q(\boldsymbol{d}|\boldsymbol{Z}) q(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z}) \tag{4.5}$$

where we employ an encoder network that maps the input data into the local latent posterior distribution, which is defined as a Gaussian:

$$q(\boldsymbol{Z}|\boldsymbol{X}) = \prod_{i=1}^{B} q(\boldsymbol{z}_i|\boldsymbol{x}_i) = \prod_{i=1}^{B} \mathcal{N}(\boldsymbol{\mu}_{\phi_z}(\boldsymbol{x}_i), \boldsymbol{\Sigma}_{\phi_z}(\boldsymbol{x}_i)) \tag{4.6}$$

Given the posterior distribution of $\boldsymbol{z}$, the categorical posterior distribution of d_i is parametrized by a NN that takes $\boldsymbol{z}_i$ as input

$$q(\boldsymbol{d}|\boldsymbol{Z}) = \prod_{i=1}^{B} q(d_i|\boldsymbol{z}_i) = \prod_{i=1}^{B} \mathrm{Cat}(\pi_{\phi_d}(\boldsymbol{z}_i)) \tag{4.7}$$

The approximate posterior distribution of the global variable $\boldsymbol{\beta}$ is computed as a product of local contributions per datapoint within a randomly sampled batch. This strategy, as demonstrated by (Bouchacourt et al., 2018), outperforms other approaches like, for example, a mixture of local contributions, as it allows to accumulate group evidence. For each sample, a NN encodes $\boldsymbol{x}_i$ and the Categorical parameters $\pi_{\phi_d}(\boldsymbol{z}_i)$ in a local Gaussian

$$q(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z}) = \mathcal{N}\left(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta\right) = \frac{1}{Z} \prod_{i=1}^{B} \mathcal{N}\left(\boldsymbol{\mu}_{\phi_\beta}([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)]), \boldsymbol{\Sigma}_{\phi_\beta}([\boldsymbol{x}_i, \pi_{\phi_d}(\boldsymbol{z}_i)])\right), \tag{4.8}$$

being Z the normalizing constant. If we denote by $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ the parameters obtained by networks $\boldsymbol{\mu}_{\phi_\beta}$ and $\boldsymbol{\Sigma}_{\phi_\beta}$, respectively, the parameters of the global Gaussian distribution are given, following (Bromiley, 2003), by:

$$\begin{aligned} \boldsymbol{\Lambda}_\beta &= \boldsymbol{\Sigma}_\beta^{-1} = \sum_{i=1}^{B} \boldsymbol{\Lambda}_i \\ \boldsymbol{\mu}_\beta &= (\boldsymbol{\Lambda}_\beta)^{-1} \sum_{i=1}^{B} \boldsymbol{\Lambda}_i \boldsymbol{\mu}_i \end{aligned} \tag{4.9}$$

where $\boldsymbol{\Lambda}_\beta = \boldsymbol{\Sigma}_\beta^{-1}$ is defined as the precision matrix, which we model as a diagonal matrix.

4.2.3. Evidence Lower Bound

Overall, the evidence lower bound reads as follows:

$$\mathcal{L}(\boldsymbol{X}; \theta, \phi) = \mathbb{E}_{q(\boldsymbol{\beta})}\left[\mathcal{L}_i(\boldsymbol{x}_i; \theta, \phi)\right] - \mathbb{E}_{q(\boldsymbol{d})}\left[D_{KL}\left(q(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{Z}) \| p(\boldsymbol{\beta})\right)\right] \tag{4.10}$$

The resulting ELBO is an expansion of the ELBO for a standard GMVAE with a new regularizer for the global variable. As the reader may appreciate, the ELBO for UG-VAE does not include extra hyperparameters to enforce disentanglement, like other previous works as β-VAE, and thus, no extra validation is needed apart from the parameters of the networks architecture, the number of clusters and the latent dimensions. We denote by $\mathcal{L}_i$ each local contribution to the ELBO:

$$\begin{aligned}\mathcal{L}_i(\boldsymbol{x}_i;\theta,\phi) = &\mathbb{E}_{q(\boldsymbol{d_i},\boldsymbol{z}_i)}\left[\log p(\boldsymbol{x}_i|\boldsymbol{z}_i,d_i,\boldsymbol{\beta})\right] \\ &- \mathbb{E}_{q(\boldsymbol{d_i})}\left[D_{KL}\left(q(\boldsymbol{z}_i|\boldsymbol{x}_i)\|p(\boldsymbol{z}_i|d_i,\boldsymbol{\beta})\right)\right] - D_{KL}\left(q(d_i|\boldsymbol{z}_i)\|p(d_i)\right)\end{aligned} \tag{4.11}$$

The first part of (4.10) is an expectation over the global approximate posterior of the so-called local ELBO. This local ELBO differs from the vanilla ELBO proposed by (Kingma and Welling, 2013) in the regularizer for the discrete variable d_i, which is composed by the typical reconstruction term of each sample and two KL regularizers: one for $\boldsymbol{z}_i$, expected over d_i, and the other over d_i. The second part in (4.10) is a regularizer on the global posterior. The expectations over the discrete variable d_i are tractable and thus, analytically marginalized.

In contrast with GMVAE (Figure 4.1 (b)), in UG-VAE, $\boldsymbol{\beta}$ is shared by a group of observations, therefore the parameters of the mixture are the same for all the samples in a batch. In this manner, within each optimization step, the encoder $q(\boldsymbol{\beta}|\mathbf{X},\mathbf{Z})$ only learns from the global information obtained from the product of Gaussian contributions of every observation, with the aim at configuring the mixture to improve the representation of each datapoint in the batch, by means of $p(\mathbf{Z}|\boldsymbol{d},\boldsymbol{\beta})$ and $p(\mathbf{X}|\mathbf{Z},\boldsymbol{\beta})$. Hence, the control of the mixture is performed by using global information. In contrast with ML-VAE (whose encoder $q(C_G|\mathbf{X})$ is also global, but the model does not include a mixture), in UG-VAE, the $\boldsymbol{\beta}$ encoder incorporates information about which component each observation belongs to, as the weights of the mixture inferred by $q(\boldsymbol{d}|\mathbf{Z})$ are used to obtain $q(\boldsymbol{\beta}|\mathbf{X},\mathbf{Z})$. Thus, while each cluster will represent different local features, moving $\boldsymbol{\beta}$ will affect all the clusters. In other words, modifying $\boldsymbol{\beta}$ will have some effect in each local cluster. As the training progresses, the encoder $q(\boldsymbol{\beta}|\mathbf{X},\mathbf{Z})$ learns which information emerging from each batch of data allows to move the cluster in a way that the ELBO increases.

4.3. Experiments

In this section we demonstrate the ability of the UG-VAE model to infer global factors of variation that are common among samples, even when coming from different datasets. In all cases, we have not validated in depth all the networks used, we have merely rely on encoder/decoder networks proposed in state-of-the-art VAE papers such as (Kingma and Welling, 2013), (Bouchacourt et al., 2018) or (Higgins et al., 2016). Our results must be hence regarded as a proof of concept about the flexibility and representation power of UG-VAE, rather than fine-tuned results for each case. Hence there is room for improvement in all cases. Details about network architecture and training parameters are provided in the Appendix A.2.

4.3.1. Unsupervised learning of global factors

Qualitative analysis

In this section we first asses the interpretability of the global disentanglement features inferred by UG-VAE over both CelebA and MNIST. In Figure 4.4 we show samples of

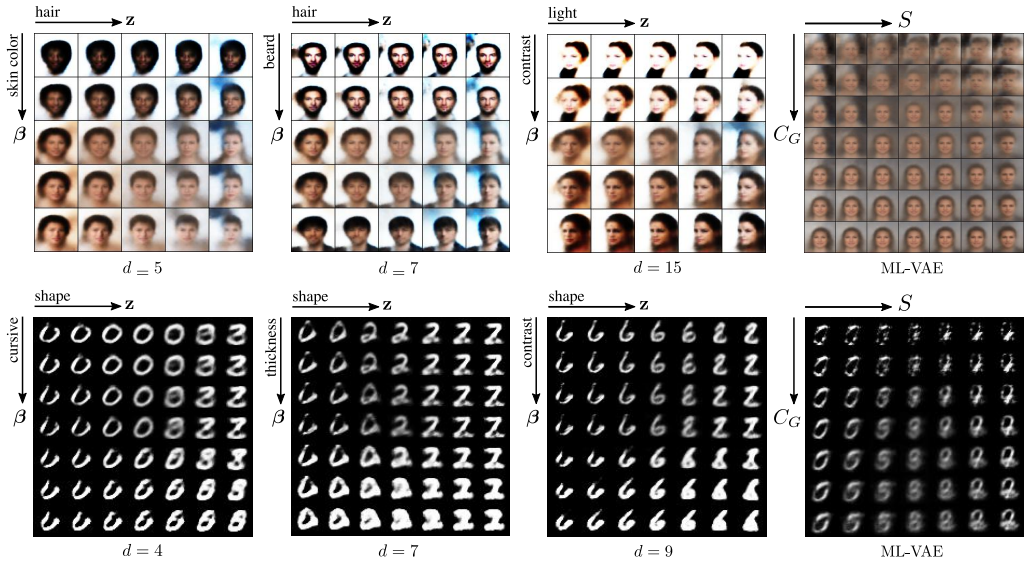


Figure 4.4: Sampling from UG-VAE (first three columns) and ML-VAE (last column) for CelebA (top) and MNIST (bottom). We include samples from 3 local clusters of UG-VAE from a total of $K = 20$ for CelebA and $K = 10$ for MNIST. In CelebA (top), the global latent variable disentangles in skin color, beard and face contrast, while the local latent variable controls hair and light orientation. In MNIST (bottom), $\boldsymbol{\beta}$ controls cursive grade, contrast and thickness of handwriting, while $\boldsymbol{z}$ varies digit shape. In ML-VAE (right column), both spaces are unimodal and the disentanglement is hardly interpretable when we feed the data without semi-supervision.

the generative model as we explore both the global and local latent spaces. We perform a linear interpolation with the aim at exploring the hypersphere centered at the mean of the distribution and with radius σ_i for each dimension i. Instead of finding influential latent factors (Liu et al., 2020) and interpolate them (fixing the rest), we choose to maximize the variation range across every dimension, moving diagonally through the latent space. Rows correspond to an interpolation on the global $\boldsymbol{\beta}$ between $[-1, 1]$ on every dimension ($p(\boldsymbol{\beta})$ follows a standard Gaussian). As the local $p(\boldsymbol{z}|d, \boldsymbol{\beta})$ ((4.3)) depends on d and $\boldsymbol{\beta}$, if we denote $\boldsymbol{\mu}_z = \boldsymbol{\mu}_z^{(d)}(\boldsymbol{\beta})$, the local interpolation goes from $[\mu_{z0} - 3, \mu_{z1} - 3, ...\mu_{zd} - 3]$ to $[\mu_{z0} + 3, \mu_{z1} + 3, ..., \mu_{zd} + 3]$. The range of ± 3 for the local interpolation is determined to cover the variances $\boldsymbol{\Sigma}_z^{(d)}(\boldsymbol{\beta})$ that we observe upon training the model for MNIST and CelebA. The every image in Figure 4.4 correspond to samples from a different cluster (fixed values of d), in order to facilitate the interpretability of the information captured at both local and global levels. By using this set up, we demonstrate that the global information tuned by $\boldsymbol{\beta}$ is different and clearly interpretable inside each cluster. In order to visually remark the advantage of capturing global correlations among samples with UG-VAE, we include in Figure 4.5 an interpolation in the latent space of β-VAE. We explore from $\boldsymbol{z} = [-1, -1, ..., -1]$ to $\boldsymbol{z} = [1, 1, ..., 1]$, given that the prior is an isotropic Gaussian. As the reader may appreciate, only one row is included as β-VAE does not include global space. In this case, moving diagonally through the latent space starts from a blond woman and ends in a brunette woman with the same angle face. Thus, the local space is in charge of encoding both content and style aspects. Although in β-

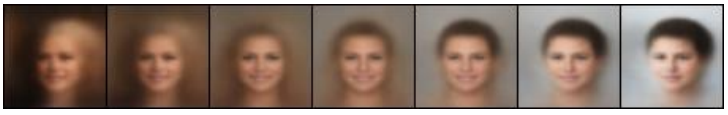


Figure 4.5: Interpolation in the prior latent space of β-VAE with $\beta = 10$, using the same networks architecture than in the local part of UG-VAE. Interpolation consists on 7 steps from $\boldsymbol{z} = [-1, -1, ..., -1]$ to $\boldsymbol{z} = [1, 1, ..., 1]$.

VAE, authors analyze the disentanglement in each dimension of the latent space, we do not study whether each dimension of $\boldsymbol{z}$ represents an interpretable generative factor in UG-VAE or not, as it is out of the scope for this work. The novelty lies on the fact that, apart from the local disentanglement, our model adds an extra point of interpretability through the global space.

The total number of clusters is set to $K = 20$ for CelebA and $K = 10$ for MNIST. Three of these components are presented in Figure 4.4. We can observe that each row (each value of $\boldsymbol{\beta}$) induces a shared generative factor, while $\boldsymbol{z}$ is in charge of variations inside this common feature. For instance, in CelebA (top), features like skin color, presence of beard or face contrast are encoded by the global variable, while local variations like hair style or light direction are controlled by the local variable. In a simple dataset like MNIST (bottom), results show that handwriting global features as cursive style, contrast or thickness are encoded by $\boldsymbol{\beta}$, while the local $\boldsymbol{z}$ defines the shape of the digit. The characterization of whether these generative factors are local/global is based on an interpretation of the effect that varying $\boldsymbol{z}$ and $\boldsymbol{\beta}$ provokes in each image within a batch, and in the whole batch of images, respectively. In Appendix A.1.1, we reproduce the same figures for the all the clusters, in which we can appreciate that there is a significant fraction of clusters with visually interpretable global/local features.

We stress here again the fact that the UG-VAE training is fully unsupervised: data batches during training are completely randomly chosen from the training dataset, with no structured correlation whatsoever. Unlike other approaches for disentanglement, see (Higgins et al., 2016) or (Mathieu et al., 2019b), variational training in UG-VAE does not come with additional ELBO hyperparameters that need to be tuned to find a proper balance among terms in the ELBO.

One of the main contributions in the design of UG-VAE is the fact that, unless we include a clustering mixture prior in the local space controlled by the global variable $\boldsymbol{\beta}$, unsupervised learning of global factors is non-informative. To illustrate such a result, in Figure 4.4 (right most column) we reproduce the same results but for a probabilistic model in which the discrete local variable d is not included. Namely, we use the ML-VAE in Figure 4.2(c) but we trained it with random data batches. In this case, the local space is uni-modal given $\boldsymbol{\beta}$ and we show interpolated values between -1 to 1. Note that the disentanglement effect of variations in both $\boldsymbol{\beta}$ and $\boldsymbol{z}$ is mild and hard to interpret.

Quantitative analysis

It remains a challenge for generative models to obtain a quantitative appropriate metric for evaluating the quality of the generated images. In this work, we employ the FID (Frechet Inception Distance), proposed by Heusel et al. (2017), which summarizes the distance between the Inception feature vectors (Szegedy et al., 2015) for real and generated images in the same domain, with the advantage that it is correlated with the better quality of the generated images. In Table 4.1 we include the score for samples from

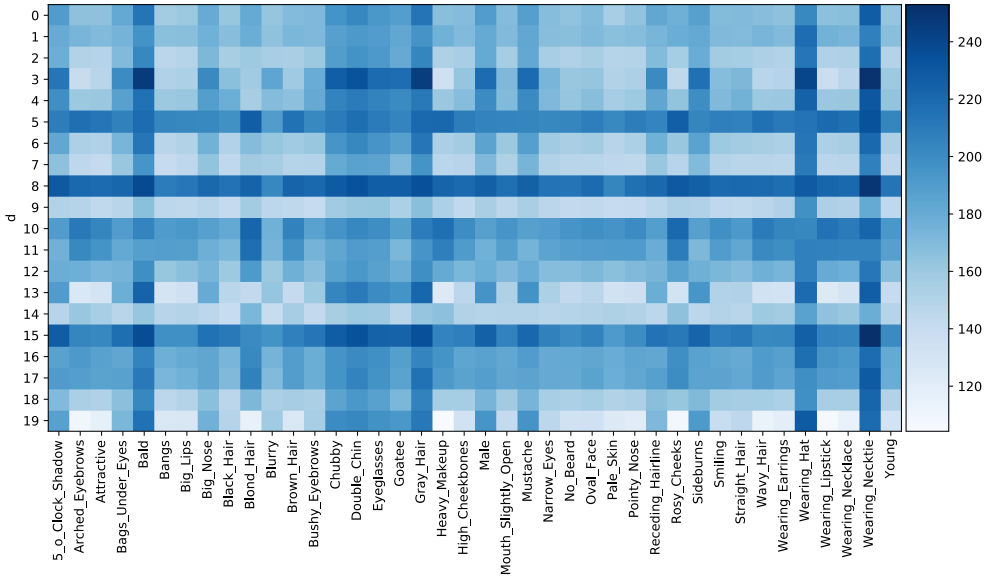


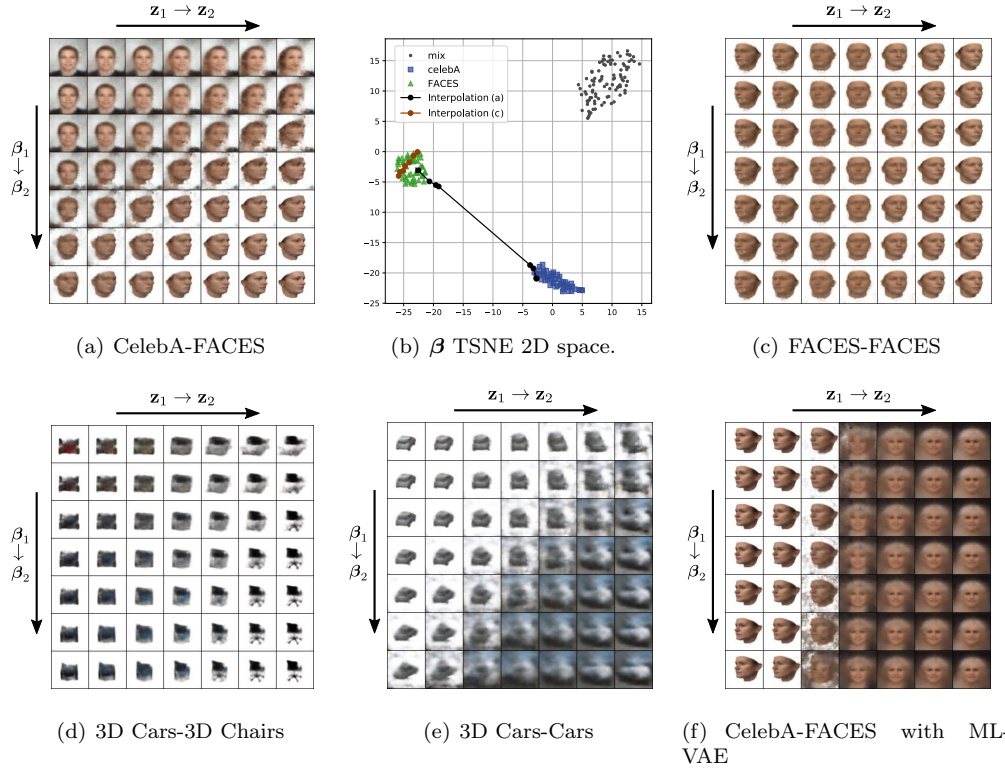
Figure 4.6: FID score between subsets of 1280 images from CelebA with a given attribute and 1280 images generated with UG-VAE from a fixed cluster d.

UG-VAE, ML-VAE and β-VAE. In both CelebA and MNIST, UG-VAE obtain lower distance and thus outperforms the other methods in the quality of the generated samples. We show empirically that the way the information is structured in the latent space of UG-VAE allows an improved generation of images. The reasons are: i) differently from our model and ML-VAE, in β-VAE the global information shared by groups of samples is not captured. ii) UG-VAE latent space is much more expressive than ML-VAE, where the conditional prior $p(\boldsymbol{z}|\boldsymbol{\beta})$ is unimodal. In other words, the prior $p(\boldsymbol{z}|d, \boldsymbol{\beta})$ in UG-VAE is a generalization of ML-VAE ($K = 1$). Therefore, in UG-VAE the latent space is augmented, which increases the representation capacity of the model.

Method	UG-VAE	ML-VAE	β-VAE
CelebA	$\mathbf{162.3 \pm 1.2}$	204.7 ± 2.4	173.5 ± 0.6
MNIST	$\mathbf{63.6 \pm 2.4}$	108.9 ± 4.5	133.2 ± 0.8

Table 4.1: FID score between subsets of 1280 images from the test sets of CelebA and MNIST and 1280 images generated with UG-VAE, ML-VAE and β-VAE. Results are provided as the $mean \pm std$ FID score of 9 repetitions.

In the following quantitative analysis, and with the intention at showing the wide spectrum for factor representation capacity provided by UG-VAE, we compute the FID metric between groups of CelebA images that share a given attribute, and samples generated by UG-VAE from a selected component d, in order to visualize whether the attributes are correlated with some of the components. These results are given in Figure 4.6. As one may appreciate, UG-VAE is able to encode human perceptible factors within each component of the mixture, and images from the same attribute present different FID scores for the set of clusters. Within unimodal models (for instance, ML-VAE or β-VAE), such rich



(a) CelebA-FACES (b) $\boldsymbol{\beta}$ TSNE 2D space. (c) FACES-FACES

(d) 3D Cars-3D Chairs (e) 3D Cars-Cars (f) CelebA-FACES with ML-VAE

Figure 4.7: Interpolation in local (columns) and global (rows) posterior spaces, fusing several datasets, using UG-VAE from (a) to (e). In (a) the interpolation goes between the posteriors of a sample from CelebA dataset and a sample from FACES dataset. In (b) we plot the t-SNE map of the samples from each dataset. In (c) the interpolation goes between samples from the same dataset. In (d) and (e) we include interpolations from 3D Cars to Chairs, and for 3D Cars to Cars Dataset, respectively. In (f) we reproduce the interpolation using the latent space of ML-VAE.

representation is not possible. We are able to obtain a set of basis faces that are tuned by the global variable. Incorporating the mixture allows β to control the distribution of clusters for representing groups that can be compared to the semi-supervision applied in ML-VAE by grouping.

4.3.2. Domain alignment

In this section, we evaluate the UG-VAE performance in an unsupervised domain alignment setup. During training, the model is fed with data batches that include random samples coming from two different datasets. In particular, we train our model with a mixed dataset between CelebA and 3D FACES (Paysan et al., 2009), a dataset of 3D scanned faces, with a proportion of 50% samples from each dataset inside each batch.

Upon training with random batches, in Figure 4.7, we perform the following experiment using domain supervision to create test data batches. We create two batches containing only images from CelebA and 3D FACES. Let $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ be the mean global posterior computed using (4.8) associated for each batch. For two particular images in

these two batches, let $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ be the mean local posterior of these two images, computed using (4.3). Figure 4.7 (a) shows samples of the UG-VAE model when we linearly interpolate between $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ (rows) and between $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ (columns)[1]. Certainly $\boldsymbol{\beta}$ is capturing the domain knowledge. For fixed $\mathbf{z}$, e.g. $\boldsymbol{z}_1$ in the first column, the interpolation between $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ is transferring the CelebA image into the 3D FACES domain (note that background is turning white, and the image is rotated to get a 3D effect). Alternatively, for fixed $\boldsymbol{\beta}$, e.g. $\boldsymbol{\beta}_1$ in the first row, interpolating between $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ modifies the first image into one that keeps the domain but resembles features of the image in the second domain, as face rotation.

In Figure 4.7(b) we show the 2D t-SNE plot of the posterior distribution of $\boldsymbol{\beta}$ for batches that are random mixtures between datasets (grey points), batches that contain only CelebA faces (blue squares), and batches that contain only 3D faces (green triangles). We also add the corresponding points of the $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ interpolation in Figure 4.7(a). In Figure 4.7(c), we reproduce the experiment in (a) but interpolating between two images and values of $\boldsymbol{\beta}$ that correspond to the same domain (brown interpolation line in Figure 4.7(b)). As expected, the interpolation of $\boldsymbol{\beta}$ in this case does not change the domain, which suggests that the domain structure in the global space is smooth, and that the interpolation along the local space $\mathbf{z}$ modifies image features to translate one image into the other. In Figure 4.7(d) and (e) experiments with more datasets are included. When mixing the 3DCars dataset (Fidler et al., 2012) with the 3D Chairs dataset (Aubry et al., 2014), in Figure 4.7(d), we find that certain correlations between cars and chairs are captured. Interpolating between a racing car and an office desk chair leads to a white car in the first domain (top right) and in a couch (bottom left). In Figure 4.7 (e), when using the 3D Cars along with the Cars Dataset (Krause et al., 2013), rotations in the cars are induced.

Finally, in 4.7(f) we show that, as expected, the rich structured captured by UG-VAE is lost when we do not include the clustering effect in the local space, i.e. if we use ML-VAE with unsupervised random data batches, and all the transition between domains is performed within the local space.

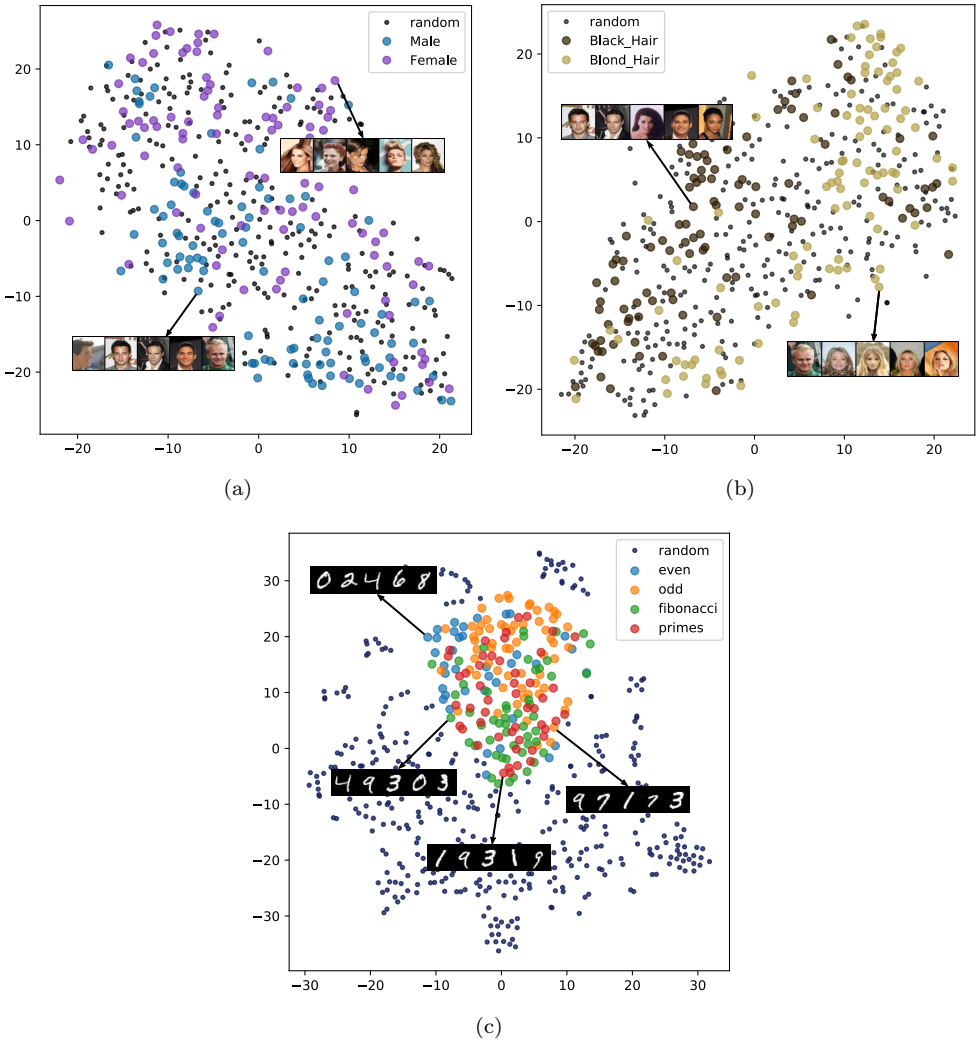
4.3.3. UG-VAE representation of structured non-trivial data batches

In the previous subsection, we showed that the UG-VAE global space is able to separate certain structure in the data batches (e.g. data domain) even though during training batches did not present such an explicit correlation. Using UG-VAE trained over CelebA with unsupervised random batches of 128 images as a running example, in this section we want to further demonstrate this result.

In Figure 4.8 we show the t-SNE 2D projection of structured batches using the posterior $\boldsymbol{\beta}$ distribution in (4.8) over CelebA and MNIST test images. In Figure 4.8(a), we display the distribution of batches containing only men and women, while in Figure 4.8(b) the distribution of batches containing people with black or blond hair. In both cases we show the distribution of randomly constructed batches as the ones in the training set. To some extend, in both cases we obtain separable distributions among the different kinds of batches. A quantitative evaluation can be found in Table 4.2. We have employed samples from the $\boldsymbol{\beta}$ distribution to train a supervised classifier that discriminates between different types of batches. When random batches are not taken as a class, the separability is evident. When random batches are included, it is expected that the classifier struggles

[1]Note that since both $\boldsymbol{\beta}$ and $\boldsymbol{z}$ are deterministically interpolated, the discrete variable d plays no role to sample from the model.

to differentiate between a batch that contains 90% of male images and a batch that only contain male images, hence the drop in accuracy for the multi-case problem.



(a) (b)

(c)

Figure 4.8: 2D t-SNE projection of the UG-VAE β posterior distribution of structured batches of 128 CelebA images. UG-VAE is trained with completely random batches of 128 train images.

An extension with similar results when using structured grouped batches from MNIST dataset for testing our model is exposed in Figure 4.8(c). In this experiment, the groups are digits that belong to certain mathematical series, including even numbers, odd numbers, Fibonacci series and prime numbers. We prove that UG-VAE is able to discriminate among their global posterior representations.

Table 4.2: Batch classification accuracy using samples of the posterior $\boldsymbol{\beta}$ distribution.

Batch categories	**Classifier**	**Train accuracy**	**Test accuracy**
Black (0) vs blond (1)	Linear SVM	1.0	0.95
	RBF SVM	1.0	0.98
Black (0) vs blond (1) vs random (2)	Linear SVM	0.91	0.54
	RBF SVM	0.85	0.56
Male (0) vs female (1)	Linear SVM	1.0	0.85
	RBF SVM	1.0	0.85
Male (0) vs female (1) vs random (2)	Linear SVM	0.84	0.66
	RBF SVM	0.89	0.63

4.4. Conclusion

In this Chapter we have presented UG-VAE, an unsupervised deep generative model able to capture both local and global factors from batches of data samples. Unlike similar approaches in the literature, by combining a structured clustering prior in the local latent space with a Gaussian global prior and a structured variational family, we have demonstrated that interpretable group features can be inferred from the global space in a completely unsupervised fashion. Model training does not require artificial manipulation of the ELBO to force latent interpretability, which makes UG-VAE stand out w.r.t. most of the current disentanglement approaches using VAEs.

The ability of UG-VAE to infer diverse features from the training set is further demonstrated in a domain alignment setup, where we show that the global space allows interpolation between domains, and also by showing that images in correlated batches of data, related by non-trivial features such as hair color or gender in CelebA, define identifiable structures in the posterior global space.

The code is publicly available at https://github.com/ipeis/UG-VAE. The package includes the UG-VAE model, and all the experiments of this paper for reproducibility purposes.

CHAPTER 5

Hierarchical VAEs and Hamiltonian Monte Carlo

Many real-world unsupervised learning tasks require dealing with complicated datasets with mixed types (real, positive-valued, continuous, or discrete) and missing values. For this purpose, variational autoencoders (Kingma and Welling, 2013; Rezende et al., 2014; Kingma and Welling, 2019) stand out in the recent literature as robust generative models that efficiently handle high-dimensional data. However, in their naive configuration, every data dimension is assumed to have similar statistical properties (i.e., homogeneity), and all dimensions are considered to be completely observed. Both assumptions won't hold in many real-world scenarios. Recent works have adapted VAEs to handle incomplete (Collier et al., 2020; Ma et al., 2018; Garnelo et al., 2018; Mattei and Frellsen, 2019) and mixed-type data (Nazabal et al., 2020; Ma et al., 2020; Gong et al., 2021), and demonstrated improved performance in downstream tasks such as missing data imputation and active information acquisition. Despite these advances, existing approaches are far from optimal as they are based on restrictive design choices: 1), only one layer of latent variables are considered; 2), Gaussian posterior approximations are usually adopted. These will lead to limited flexibility and additional bias, especially under real-world settings with complex mixed-type incomplete data.

In the literature, the issue of model flexibility and inference bias are often addressed separately. For example, approximate inference bias can be reduced by using Monte Carlo sampling (Salimans et al., 2015; Thin et al., 2021). More specifically, Hamiltonian Monte Carlo (HMC) (Duane et al., 1987; Betancourt and Girolami, 2015) stands out among MCMC methods in machine learning due to its superior efficiency for exploring the target density. In the context of VAE, HMC has also been combined with stochastic variational inference (Caterini et al., 2018) for improving the training of VAEs. On the other hand, the flexibility of VAEs can be improved by considering hierarchical VAEs with multiple layers of hidden variables (Sønderby et al., 2016; Maaløe et al., 2019; Vahdat and Kautz, 2020; Child, 2020). By using a hierarchical structure in the latent space, they enforce the information to flow from high-level representations to more specific observable factors, imitating the way information is often organized in the real world.

However, the issue of modeling flexibility and approximate inference bias are often heavily intertwined, and addressing them simultaneously in a *joint* manner is highly non-trivial. The hierarchical organization of the latent variables creates complicated posterior dependencies that are not straightforward to deal with and require special consideration. To improve Gaussian approximate inference, most works opt by defining shared paths between the recognition and generative networks. While this makes hierarchical VAEs practical, the bias introduced by the Gaussian approximations is still present. To the best of our knowledge, none of the aforementioned hierarchical VAEs has been previously combined with Monte Carlo algorithms for improving over standard Gaussian approximate inference.

To overcome such limitations, we focus on training new hierarchical VAE models for heterogeneous mixed-type data with HMC. Our models can be used for missing data imputation and for supervised learning with missing data. We also present a sampling-based framework that allows our models to perform accurate sequential active information acquisition.

This chapter is organized as follows: in Section 5.1 we revise alternative methods for dealing with incomplete data using VAEs, recent hierarchical VAEs, Hamiltonian Monte Carlo, and the task of Active Feature Acquisition. The HH-VAEM model, all its components, its optimization algorithm and computational cost are described in Section 5.2. Our proposed sampling-based active learning algorithm is presented in Section 5.3. The findings described in this chapter were presented as a paper in the main track of the NeurIPS22 conference (Peis et al., 2022).

5.1. Related work

5.1.1. VAEs for mixed-type incomplete data

Variational Autoencoders (Kingma and Welling, 2013; Rezende et al., 2014) are deep generative models that make use of encoder and decoder networks for mapping data into a latent Gaussian distribution, and reconstructing the latent codes into the original observational space, respectively. The parameters of these networks are trained using amortized Variational Inference (Zhang et al., 2018; Cremer et al., 2018) optimizing a lower bound (ELBO) on the log evidence: $\mathbb{E}_{q_{\boldsymbol{z}}(\boldsymbol{z}|\boldsymbol{x})} \log \frac{p_\theta(\boldsymbol{x},\boldsymbol{z})}{q_\psi(\boldsymbol{z}|\boldsymbol{x})}$, where the generative model $p_\theta(\boldsymbol{x},\boldsymbol{z})$ can be expressed in terms of the likelihood $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ and the prior $p(\boldsymbol{z})$. In mixed-type data, the vector $\boldsymbol{x}$ is composed by data from different types: real, positive real, categorical, binary, etc. A naive approach is to consider a factorized decoder using different likelihood contributions $p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \prod_d p_\theta(x_d|\boldsymbol{z})$ (Nazabal et al., 2020; Barrejón et al., 2021). Nonetheless, the problem of handling unbalanced likelihoods leads to the domination of some dimensions during the optimization process. In (Ma et al., 2020), authors propose a solution using a set of *marginal* VAEs that encode each feature into a Gaussian uni-dimensional space, and a *dependency* VAE that captures the inter-dimensional dependencies more effectively using balanced Gaussian likelihoods.

By marginalizing each dimension of the decoder, incomplete data can be easily handled by dividing the vector $\boldsymbol{x}$ into the observed $\boldsymbol{x}_O$ and unobserved $\boldsymbol{x}_U$ parts. This methodology is completely valid when using the missing-at-random (MAR) assumption (Little and Rubin, 2019), i.e. assuming the missing mechanism is independent of the missing values. In this work, the same assumption is adopted. As proposed in (Nazabal et al., 2020) and (Mattei and Frellsen, 2019), the ELBO objective is transformed into a lower bound on the observed data, and the unobserved data is replaced with zeros.

5.1.2. Hierarchical VAEs

Hierarchical models have been successfully employed in deep generative modeling, (Bengio et al., 2009; Salakhutdinov and Hinton, 2009; Salakhutdinov, 2015). In VAEs, defining a hierarchical latent space for VAEs can be straightforward. Nevertheless, potential pitfalls require special attention. Concretely, if the decoder is powerful enough, the model tends to uniquely use the shallowest layers, ignoring the deepest ones and falling into the well-known problem of *posterior collapse* (Wang and Cunningham, 2020; Razavi et al., 2019a; Maaløe et al., 2019). In the last few years, several works have investigated

possible hierarchical structures for VAEs. In (Sønderby et al., 2016), a bottom-up deterministic path is used along with a top-down inference path that shares parameters with the top-down structure of the generative model. In (Maaløe et al., 2019), the authors use a bidirectional stochastic inference path. More recently, (Vahdat and Kautz, 2020) or (Child, 2020) have adapted these architectures to complex datasets and high quality images. Possibly motivated by the residual connections in (Kingma et al., 2016), all these works use a shared path between recognition and generative models that helps in tying the divergences between approximations and priors in the ELBO.

5.1.3. Hamiltonian Monte Carlo

HMC (Duane et al., 1987; Neal et al., 2011; Betancourt, 2017) is a particularly effective MCMC algorithm for sampling from a target distribution $p(\boldsymbol{z}) = \frac{1}{\mathcal{Z}} p^*(\boldsymbol{z})$ where $\mathcal{Z}$ is the normalization constant and $\boldsymbol{z}$ is a d-dimensional vector. By augmenting this model to $p(\boldsymbol{r}, \boldsymbol{z}) = \mathcal{N}(\boldsymbol{r}; \boldsymbol{0}, \boldsymbol{M}) p(\boldsymbol{z})$, and denoting $\boldsymbol{r}$ as the *momentum* variable with diagonal covariance matrix $\boldsymbol{M}$, with the same dimensionality as $\boldsymbol{z}$, HMC samples are obtained from the distribution by simulating the time-evolution of a fictitious physical system.

The algorithm starts by firstly sampling $\boldsymbol{z}$ and $\boldsymbol{r}$ from an initial proposal and the momentum distribution, respectively. Chains with length T are built by recurrently proposing and accepting new states. To propose a new state, the Hamiltonian dynamics are simulated using a symplectic integrator, Leapfrog being the most common choice. The following updates are repeated for $l = 1 : LF$ steps:

$$\begin{aligned} \boldsymbol{r}_{l+\frac{1}{2}} &= \boldsymbol{r}_l + \frac{1}{2} \boldsymbol{\phi} \odot \nabla_{z_l} \log p^*(\boldsymbol{z}_l) \,, \\ \boldsymbol{z}_{l+1} &= \boldsymbol{z}_l + \boldsymbol{r}_{l+\frac{1}{2}} \odot \boldsymbol{\phi} \odot \frac{1}{\boldsymbol{M}} \,, \\ \boldsymbol{r}_{l+1} &= \boldsymbol{r}_{l+\frac{1}{2}} + \frac{1}{2} \boldsymbol{\phi} \odot \nabla_{z_{l+1}} \log p^*(\boldsymbol{z}_{l+1}), \end{aligned} \tag{5.1}$$

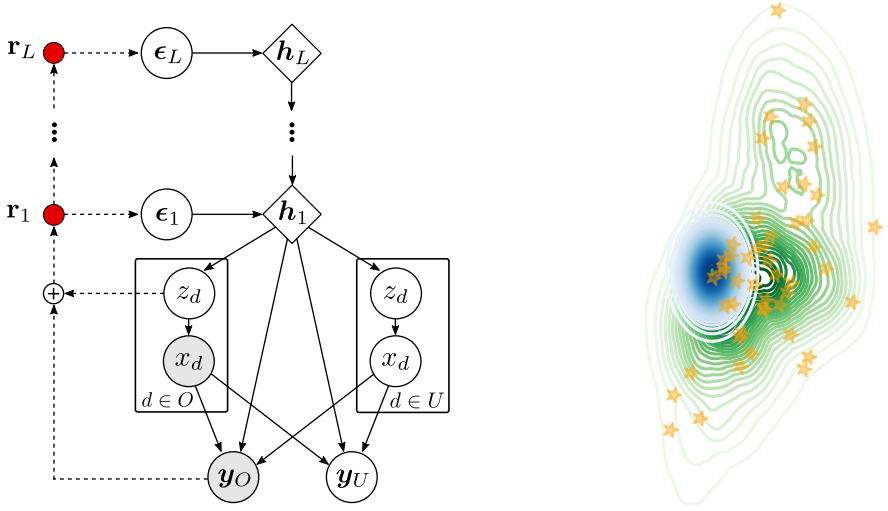
where $\odot$ refers to the Hadamard product, and ϕ is the *step size* hyperparameter. Although it is typically defined as a scalar for simplicity, a d-dimensional vector can be considered to apply different step sizes per dimension, or further, as considered in this work, a $T \times d$ matrix to apply different steps per each proposal of the chain. The new proposal $(\boldsymbol{z}', \boldsymbol{r}')$ is accepted with probability $min\left[1, \exp(-H(\boldsymbol{z}', \boldsymbol{r}') + H(\boldsymbol{z}, \boldsymbol{r}))\right]$, where

$$H(\boldsymbol{z}, \boldsymbol{r}) = -\log p^*(\boldsymbol{z}) + \frac{1}{2} \boldsymbol{r}^T \boldsymbol{M}^{-1} \boldsymbol{r}. \tag{5.2}$$

For the consecutive T proposals, a new momentum $\boldsymbol{r}$ is resampled and the updates of (5.1) are repeated for LF steps to update the state if $(\boldsymbol{z}', \boldsymbol{r}')$ is accepted.

5.1.4. Active Feature Acquisition

Among all the Active Learning techniques, Active Feature Acquisition (Melville et al., 2004; Saar-Tsechansky et al., 2009; Thahir et al., 2012; Huang et al., 2018) is of special interest in cost-sensitive applications for modeling a trade-off between the improvement of predictions and the cost of acquiring new data at the feature level. Several works in the recent literature have studied methods for performing the task of sequentially acquiring high-value information by selecting features that maximize an information theoretical reward function and enhance the accuracy of the predictions. This task is denoted by SAIA (Sequential Active Information Acquisition). In (Ma et al., 2018), an efficient



(a) The HH-VAEM model

(b) HMC sampling

Figure 5.1: The HH-VAEM model (a). Illustrative example (b): samples $\boldsymbol{\epsilon}^{(T)}$ obtained with HMC (orange) following the true posterior $p(\boldsymbol{\epsilon}|\boldsymbol{x}_O, \boldsymbol{y}_O)$ (green) using the Gaussian distribution given by the encoder $q^{(0)}(\boldsymbol{\epsilon}|\boldsymbol{x}_O, \boldsymbol{y}_O)$ (blue) as the initial proposal, with latent dimension $M = 2$.

method is proposed for approximating a non-tractable reward by using the encoder of a VAE that handles missing data. In (Ma et al., 2020) they extend this method for handling mixed-type data. Both works estimate the reward by relying on Gaussian approximations given by the encoder networks.

5.2. Hamiltonian Hierarchical VAE for Mixed-type incomplete data

The HH-VAEM model (Figure 5.1 (a)) is a Hierarchical VAE for mixed-type, incomplete data that incorporates HMC with automatic hyper-parameter optimization for sampling from the posterior of the latent variables. In a first stage, the mixed-type data is encoded into marginal Gaussian posterior distributions as given by univariate VAEs fitted to each data dimension. In a second stage, a hierarchical structure captures the dependencies among the standarized, homogeneous dimensions with well-balanced Gaussian likelihoods. The model is trained using samples from the posterior of the hierarchical latent variables by means of HMC, whilst the HMC hyper-parameters are automatically tuned. A more detailed description is provided in the following subsections.

5.2.1. Notation

The model generates both data $\boldsymbol{x} \in \mathbb{R}^D$ and output $\boldsymbol{y} \in \mathbb{R}^P$, where each of these variables is divided into observed parts $\boldsymbol{x}_O$, $\boldsymbol{y}_O$ and unobserved parts $\boldsymbol{x}_U$, $\boldsymbol{y}_U$. Each dimension of $\boldsymbol{x}$ is denoted by x_d. A training set is composed of N observations as tuples $(\boldsymbol{x}_O^{(n)}, \boldsymbol{y}_O^{(n)})$. For ease of notation, we omit the observation index n, and the objectives

are presented for a single observation. The *dependency* latent space is composed of L latent variables $[\boldsymbol{\epsilon}_1, ..., \boldsymbol{\epsilon}_L]$, with $\boldsymbol{\epsilon}_l \in \mathbb{R}^{m_l}$. The dimension of the joint latent distribution is $\sum_l m_l = M$. In the marginal VAEs, the latent variables z_d are unidimensional.

5.2.2. Handling heterogeneous incomplete data

Following the strategy proposed by (Ma et al., 2020), we perform a two-staged approach for handling heterogeneous data. The marginal distribution of each feature $p_{\theta_d}(x_d)$ is modeled by a one-dimensional VAE. First, the D marginal VAEs are trained independently by maximizing the marginal ELBO over observed points

$$\mathcal{L}_d(x_d; \{\theta_d, \gamma_d\}) = \mathbb{1}(x_d \in \boldsymbol{x}_O)\mathbb{E}_{q_{\gamma_d}(z_d|x_d)} \log \frac{p_{\theta_d}(x_d, z_d)}{q_{\gamma_d}(z_d|x_d)}, \tag{5.3}$$

where $\mathbb{1}(x_d \in \boldsymbol{x}_O)$ is an indicator function that activates the ELBO when the feature is observed. Under the missing-at-random assumption, which is the one considered in this work, (5.3) leads to a lower bound on observed data likelihoods. Second, a *dependency* VAE encodes a vector $\boldsymbol{z}$ with the concatenated samples from the marginal posteriors $q_{\gamma_d}(z_d|x_d)$ into the global latent variable $\boldsymbol{h}$, using zero-filling for the unobserved variables. By using this approach, $\boldsymbol{z}$ is now homogeneous and can then be easily modeled using a standard Gaussian decoder. By contrast, other works (Nazabal et al., 2020; Eduardo et al., 2020) directly operate with different decoding likelihoods per data type. This approach often leads to having very different magnitudes in the ELBO and may reduce learning efficiency. The ELBO for the second stage dependency VAE is

$$\mathcal{L}(\boldsymbol{x}_O, \boldsymbol{y}_O; \{\theta, \psi\}) = \mathbb{E}_{q_\psi}\left[\log \frac{p_\theta(\boldsymbol{z}_O, \boldsymbol{y}_O, \boldsymbol{\epsilon})}{q_\psi(\boldsymbol{\epsilon}|\boldsymbol{z}_O, \boldsymbol{x}_O, \boldsymbol{y}_O)}\right] \tag{5.4}$$

where $\boldsymbol{\epsilon} = \{\boldsymbol{\epsilon}_1, ..., \boldsymbol{\epsilon}_L\}$ is a set of reparameterized hierarchical latent variables. Further details on the design of the hierarchical dependency VAE are provided below.

5.2.3. Predictive enhancement

The combination of generative and discriminative models is an effective well-studied strategy for dealing with predictive models under missing data (Tresp et al., 1993; Ghahramani and Jordan, 1995). In (Ghahramani and Jordan, 1995), they model $p(\boldsymbol{x})$ for imputing missing data using a Gaussian Mixture Model. In a deep learning context, recent supervised VAE models have revisited this combination (Śmieja et al., 2018; Ipsen et al., 2020) or used factorisations of type $p(\boldsymbol{z})p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{y}|\boldsymbol{z})$ (Joy et al., 2021) to learn meaningful representations. In (Li et al., 2019) the authors propose a deep generative model with factorisation $p(\boldsymbol{z})p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{z})$ for detecting adversarial attacks.

With the aim at reinforcing the prediction of the variable of interest, we turn into a supervised model by including a separate predictor for $p_{\theta_y}(\boldsymbol{y}|\hat{\boldsymbol{x}}, \boldsymbol{h})$, apart from the decoder $p_{\theta_z}(\boldsymbol{z}|\boldsymbol{h})$. The vector $\hat{\boldsymbol{x}} = (x_i \in \boldsymbol{x}_O, \hat{x}_j \in \boldsymbol{x}_U)$ includes the observed part and imputation of the missing variables $\hat{x}_j$ by decoding the latent $\boldsymbol{h}$ into $\boldsymbol{z}$ using $p(\boldsymbol{z}|\boldsymbol{h_1})$, and each dimension $z_j \in \boldsymbol{z}$ into $\hat{x}_j$ using $p(x_j|z_j)$. The predictor parameters θ_y are optimized along with the decoder parameters θ_z.

5.2.4. Hierarchical reparameterized latent space

A hierarchical structure over the latent space $\boldsymbol{h} = \{\boldsymbol{h}_1, ..., \boldsymbol{h}_L\}$ enriches the prior assumptions and permits a flexible generation of data in a more natural fashion. Nevertheless, as stated in (Betancourt, 2017; Betancourt and Girolami, 2015), HMC can be

pathological when used for sampling from hierarchical densities, where the magnitude of autoregressive variations increase with the depth. For approximating the Hamiltonian dynamics, inside each Leapfrog integrator step, gradients $\nabla_{\boldsymbol{h}_{1:L}} \log p^*(\boldsymbol{h}_{1:L})$ are required. Due to the strong curvature regions, huge norm of high-order derivatives are backpropagated and might eventually explode, ending in overflow issues (Figure 35 in (Betancourt and Girolami, 2015)).

If we were to run our HMC method over the hierarchical variables without any reparameterization (Figure 5.4a), by the time the states reached the aforementioned problematic regions, the integrator would diverge and we would experience the aforementioned overflow problems. By rejecting these problematic states, chains would get stuck close to the proposal and the hierarchical density would not be properly explored (concluding that HMC would not improve the Gaussian proposal). To give an example, in (Vahdat and Kautz, 2020), the AR path $p(\boldsymbol{h}_l|\boldsymbol{h}_{<l})$ would provoke instabilities inside the HMC integrator due to huge gradients in $\nabla_{\boldsymbol{h}_{1:L}} \log p^*(\boldsymbol{h}_{1:L})$.

We successfully solved this issue by introducing a hierarchical reparameterization technique. The representation at each layer is reparameterized from variable $\boldsymbol{\epsilon}_l$ with standard Gaussian prior $p(\boldsymbol{\epsilon}_l)$

$$\boldsymbol{h}_l = f_{\mu_l}(\boldsymbol{h}_{l+1}) + f_{\sigma_l}(\boldsymbol{h}_{l+1}) \cdot \boldsymbol{\epsilon}_l, \tag{5.5}$$

where the functions $f_{\mu_l}(\boldsymbol{h}_{l+1})$ and $f_{\sigma_l}(\boldsymbol{h}_{l+1})$ are applied by NNs with parameters $\theta_l = \{\theta_{\mu_l}, \theta_{\sigma_l}\}$. The result is equivalent as learning the mean and covariance of autoregressive variables (see Figure 5.4 for illustrative details). However, thanks to this trick, we relax the dependencies among the latent variables, resulting in a smoother joint posterior density $p(\boldsymbol{\epsilon}|\boldsymbol{x}_O, \boldsymbol{y}_O)$. Performing the inference over $\boldsymbol{\epsilon} = \{\boldsymbol{\epsilon}_1, ..., \boldsymbol{\epsilon}_L\}$ leads to a better posed basis for running our HMC optimization, detailed in Section 5.2.5, and avoids the necessity of employing more advanced HMC samplers like (Girolami and Calderhead, 2011; Betancourt and Girolami, 2015). We include further details on the pathological behavior and demonstration of the effectiveness of our solution in Appendix B.1.2. Provided the promising results we obtain in Section 5.4, we propose our reparameterization trick as a novel contribution for solving the pathological behavior of HMC combined with hierarchical VAEs.

For the sake of simplicity, we name the generative parameters as $\theta = \{\theta_z, \theta_y, \theta_1, ..., \theta_L\}$. The dependency ELBO under this hierarchical reparameterized model becomes

$$\begin{aligned} \mathcal{L}_{VI}(\boldsymbol{x}_O, \boldsymbol{y}_O; \{\theta, \psi\}) = \mathbb{E}_{q_\psi}\left[\log \frac{p_\theta(\boldsymbol{z}_O, \boldsymbol{y}_O, \boldsymbol{\epsilon})}{q_\psi(\boldsymbol{\epsilon}|\boldsymbol{z}_O, \boldsymbol{x}_O, \boldsymbol{y}_O)}\right] = \\ \mathbb{E}_{q_\psi}\left[\log p_\theta(\boldsymbol{z}_O|\boldsymbol{h}_1) + \log p_\theta(\boldsymbol{y}_O|\hat{\boldsymbol{x}}, \boldsymbol{h}_1)\right] - \sum_{l=1}^{L} D_{\mathrm{KL}}\left(q_\psi(\boldsymbol{\epsilon}_l|\boldsymbol{x}_O, \boldsymbol{y}_O)||p(\boldsymbol{\epsilon}_l)\right). \end{aligned} \tag{5.6}$$

We name $\boldsymbol{r}_l$ the hidden representation at each layer, and defining $\boldsymbol{r}_0 = \{\boldsymbol{x}_O, \boldsymbol{y}_O\}$, we employ NNs with parameters ψ_{r_l} for computing $\boldsymbol{r}_l = f_r(\boldsymbol{r}_{l-1})$. These vectors are mapped into the parameters of the variational posterior $q_{\psi_l}(\boldsymbol{\epsilon_l}|\boldsymbol{x}_O, \boldsymbol{y}_O)$, using NNs for computing the mean as $g_{\mu_l}(\boldsymbol{r}_l)$ and the covariance as $g_{\sigma_l}(\boldsymbol{r}_l)$, with parameters ψ_{μ_l} and ψ_{σ_l}. With compactness purposes, we will denote the encoder parameters as $\psi = \{\psi_1, ..., \psi_L\}$, with $\psi_l = \{\psi_{r_l}, \psi_{\mu_l}, \psi_{\sigma_l}\}$.

Balancing the KLs

Following (Vahdat and Kautz, 2020), we define a short initial warming stage (10% of the total training steps) during the optimization where the KLs for the different layers are

balanced according to their magnitude and the corresponding latent dimension, preventing the model for falling into posterior collapse by ignoring deepest layers. A factor is applied to each KL, following

$$\gamma_l = \frac{d_l \, \mathbb{E}_{x\sim B}\left[\mathrm{KL}(q(\boldsymbol{\epsilon}_l|\boldsymbol{x})||p(\boldsymbol{\epsilon}))\right]}{\sum_{i=1}^{L} d_i \, \mathbb{E}_{x\sim B}\left[\mathrm{KL}(q(\boldsymbol{\epsilon}_i|\boldsymbol{x})||p(\boldsymbol{\epsilon}))\right]}. \tag{5.7}$$

The factors penalises the fact that a layer might be ignored by making the KL smaller when its magnitude is small compared to the rest layers.

5.2.5. HMC over the hierarchical density

In recent works, HMC has been combined with deep generative models for improving the inference of the latent variables by obtaining better samples from the posterior (Hoffman, 2017; Caterini et al., 2018). In this work, we propose to transcend these previous approaches and build a generalized method for sampling from complicated, hierarchical latent structures composed by several layers. Inspired by (Campbell et al., 2021) and their method for sampling from the posterior within a vanilla VAE framework while tuning the HMC hyperparameters, we follow a procedure for training the dependency model where i) during a pre-training stage, the encoder and decoder are optimized using standard VI and the ELBO from (5.6), and ii) using the pre-trained encoder for starting from a good proposal (Hoffman, 2017), HMC samples are obtained to follow the true posterior and jointly optimize the generative model and the HMC hyperparameters. In Figure 5.1 (b) we include an illustrative example.

We denote by $q_\phi^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{z}_O, \boldsymbol{x}_O, \boldsymbol{y}_O)$ the implicit distribution for the posterior after T HMC steps. The hyper-parameters of HMC are named ϕ, a $T \times d$ matrix containing the step sizes for each dimension at each step of the chain. Within this perspective, the hyper-parameters can be optimized using variational inference by maximizing the ELBO

$$\mathbb{E}_{q_\phi^{(T)}(\boldsymbol{\epsilon})}[\log p(\boldsymbol{z}_O, \boldsymbol{y}_O, \boldsymbol{\epsilon})] + H[q_\phi^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{x}_O, \boldsymbol{y}_O)], \tag{5.8}$$

where the first part is the HMC objective, and can be easily estimated via Monte Carlo

$$\mathcal{L}_{HMC}(\boldsymbol{z}_O, \boldsymbol{y}_O; \{\theta, \psi, \phi\}) = \mathbb{E}_{q_\phi^{(T)}(\boldsymbol{\epsilon})}[\log p_\theta(\boldsymbol{z}_O|\boldsymbol{h}_1) + \log p_\theta(\boldsymbol{y}_O|\hat{\boldsymbol{x}}, \boldsymbol{h}_1) + \sum_{l=1}^{L} p(\boldsymbol{\epsilon}_l^{(T)})]. \tag{5.9}$$

Nevertheless, the entropy term $H[q_\phi^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{x}_O, \boldsymbol{y}_O)]$ in Equation (5.8) is not tractable since we are not able to explicitly evaluate the distribution $q_\phi^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{x}_O, \boldsymbol{y}_O)$. Although optimizing the first term might result in a well-posed algorithm, this would bring consequences that must be considered. Namely, without a proper regularization term, and in case the initial proposal $q_\phi^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{x}_O, \boldsymbol{y}_O)$ is concentrated in high density regions, the chains would barely move from the initial state and only these regions with high density would be explored (see Section 5.4.6 and Appendix B.1.2 for illustrative details). To cope with this problem, we define an inflation parameter $\boldsymbol{s}$ to increase the variance of the proposal $q_\phi(\boldsymbol{\epsilon}|\boldsymbol{z}_O, \boldsymbol{x}_O, \boldsymbol{y}_O)$ given by the Gaussian encoder, ending in the proposal $\prod_{l=1}^{L} \mathcal{N}(g_{\mu_l}(\boldsymbol{r}_l),\ s_l \cdot g_{\sigma_l}(\boldsymbol{r}_l))$. Whilst in (Campbell et al., 2021) the authors define this parameter as a scalar factor applied to all the latent dimensions, in our work, we extend this to apply a different inflation factor at each latent level $\boldsymbol{s} = \{s_1, ..., s_L\}$. In order to tune these inflations we ensure a wider

coverage of the density by minimizing the Sliced Kernelized Stein Discrepancy (SKSD) (Gong et al., 2020)

$$\mathcal{L}_{SKSD}(\boldsymbol{x}_O, \boldsymbol{y}_O; \boldsymbol{s}) = \text{SKSD}\left(q_\phi^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{z}_O, \boldsymbol{x}_O, \boldsymbol{y}_O; \boldsymbol{s}),\, p(\boldsymbol{\epsilon}|\boldsymbol{z}_O, \boldsymbol{x}_O, \boldsymbol{y}_O)\right), \tag{5.10}$$

which fits perfectly our requirements, since only requires samples from HMC and gradients $\nabla_\epsilon \log p(\boldsymbol{z}_O, \boldsymbol{y}_O, \boldsymbol{\epsilon})$ for measuring a discrepancy between the implicit and the true posterior. Further, the SKSD performs better than other discrepancies like (Liu et al., 2016) in high dimensional latent spaces.

We provide in Section 5.4.6 a toy demonstration on the efficacy of the HMC optimization, and in Appendix B.1.2 a demonstration of the optimization convergence.

5.2.6. HH-VAEM optimization

The optimization of the HH-VAEM is divided into three stages. In a first stage, we train one independent *marginal* VAE per dimension. In a second stage, the *dependency* VAE is trained, using as inputs the concatenation of the encoded dimensions $\boldsymbol{z} = \{z_1, ..., z_D\}$ and the target $\boldsymbol{y}$. Finally, in a third stage, the HMC hyperparameters, the decoder and the predictor are tuned using the HMC objective, the inflation parameter is trained using the SKSD discrepancy, and the encoder is trained used the ELBO. The pseudocode for HH-VAEM training is shown in Algorithm 1.

5.2.7. Computational cost

In a VAE, the data is fed to the encoder with aim at obtaining the variational parameters for sampling from the Gaussian approximated posterior. In our method, the data is similarly encoded to obtain the initial Gaussian proposal $q^{(0)}$, and the samples from this distribution are updated for T cycles to obtain the implicit $q^{(T)}$ using HMC. Within each of these iterations, L leapfrog steps (5.1) are executed. For each of these steps, the computation of the gradients $\nabla_{\boldsymbol{\epsilon}_l} \log p(\boldsymbol{z}, \boldsymbol{y}, \boldsymbol{\epsilon}_l)$ and $\nabla_{\boldsymbol{\epsilon}_{l+1}} \log p(\boldsymbol{z}, \boldsymbol{y}, \boldsymbol{\epsilon}_{l+1})$ is required. To obtain these gradients, we need to *i)* compute the parameters of the likelihood $p(\boldsymbol{z}, \boldsymbol{y}|\boldsymbol{\epsilon})$ that are given by the decoder ($p(\boldsymbol{z}|\boldsymbol{h}_1)$) and predictor ($p(\boldsymbol{y}|\hat{\boldsymbol{x}}, \boldsymbol{h}_1)$), *ii)* evaluate the likelihood and *iii)* perform the automatic differentiation. Thus, for running our method, an additional cost from both decoding and performing differentiation a total of $2TL$ times is introduced.

By jointly optimizing the HMC hyperparameters we are able to achieve faster convergence with smaller lengths. To reduce the computational cost, we optimize the hyperparameters in a final training stage, since convergence is rapidly achieved (as demonstrated empirically in Section 5.4.6). At test, samples from the Gaussian $q^{(0)}$ (faster and cheaper), or from HMC $q^{(T)}$ (slower and better) can be used to fit computational constraints.

Algorithm 1 Training algorithm for HH-VAEM

Input: data $\left(\boldsymbol{x}_O^{(1:N)}, \boldsymbol{y}_O^{(1:N)}\right)$, steps: T_d, T_{VI}, T_{HMC}
Parameters: γ, θ, ψ, ϕ, s
STAGE 1: MARGINAL VAEs
for $d = 1$ **to** D **do**
  Initialize marginal VAE $\{\theta_d, \gamma_d\}$
  for $t = 1$ **to** T_d **do**
    $\gamma_d^{t+1}, \theta_d^{t+1} \leftarrow \text{Adam}_{\gamma_d^t, \theta_d^t}(\mathcal{L}_d)$
  end for
end for
STAGE 2: DEPENDENCY VAE
for $t = 1$ **to** T_{VAE} **do**
  $\theta^{t+1}, \psi^{t+1} \leftarrow \text{Adam}_{\theta^t, \psi^t}(\mathcal{L}_{VI})$
end for
STAGE 3: JOINTLY OPTIMIZING VAE + HMC
for $t = 1$ **to** T_{HMC} **do**
  $\psi^{t+1} \leftarrow \text{Adam}_{\psi^t}(\mathcal{L}_{VI})$
  $\theta^{t+1}, \phi^{t+1} \leftarrow \text{Adam}_{\theta^t, \phi^t}(\mathcal{L}_{HMC})$
  $s^{t+1} \leftarrow \text{Adam}_{s^t}(\mathcal{L}_{SKSD})$
end for

5.3. Sampling-based Active Learning

Considering that the input data are tuples of observed and missing features $\{\boldsymbol{x}_O, \boldsymbol{x}_U\}$, our Active Learning framework follows (Ma et al., 2018; 2020) and determines which feature $\boldsymbol{x}_i \in \boldsymbol{x}_U$ will enhance the prediction of the target $\boldsymbol{y}$ the most for a particular $\boldsymbol{x}_O$. Concretely, in a Sequential Active Information Acquisition (SAIA) scenario, this decision is taken sequentially to optimally increase knowledge and accurately predict $\boldsymbol{y}$. From a information theoretical perspective, this task can be performed recurrently by maximizing a reward function R at each step d. This reward might represent abstract quantities of interest like the cost or benefit of acquiring $\boldsymbol{x}_i$ (depending on the sign). In Bayesian experimental analysis, R is the expected gain of information (Lindley, 1956). Following (Bernardo, 1979), we can define it as

$$R(i, \boldsymbol{x}_O) = \mathbb{E}_{p(x_i|\boldsymbol{x}_O)} D_{\text{KL}} \left(p(\boldsymbol{y}|x_i, \boldsymbol{x}_O) p(\boldsymbol{y}|\boldsymbol{x}_O) \right), \tag{5.11}$$

where i is the index of each unobserved feature. Intuitively, this quantity can be interpreted as the expected change in the predictive distribution when $\boldsymbol{x}_i$ is observed. The reward needs to be estimated via Monte Carlo by sampling from $p(x_i|\boldsymbol{x}_O)$. With a robust generative model that handles missing data like HH-VAEM, these samples are easily obtained: first, using HMC, we sample $\boldsymbol{\epsilon}^{(T)}$ from $p(\boldsymbol{\epsilon}|\boldsymbol{x}_O)$. Second, we decode these samples to obtain x_i from $p(\boldsymbol{z}|\boldsymbol{h}_1)$ and $p(x_i|z_d)$. Nonetheless, the reward defined in (5.11) is intractable since both $p(\boldsymbol{y}|x_i)$ and $p(\boldsymbol{y}|x_i, \boldsymbol{x}_O)$ are intractable: computing them requires to integrate out the latent variables. This motivates the authors of (Ma et al., 2018; 2020) to present a transformation of the reward for being computed in the latent space using the encoder network. Although they prove that this transformation effectively provides a good estimation in several datasets, we demonstrate that for low dimensional targets (commonly one or two dimensions), an approximation using histograms is more effective.

	Bank	Insurance	Avocado	Naval	Yatch	Diabetes	Concrete	Wine	Energy	Boston
VAEM	2.84 ± 0.07	1.81 ± 0.03	1.89 ± 0.01	0.55 ± 0.05	3.15 ± 0.28	2.78 ± 0.16	2.45 ± 0.26	3.01 ± 0.61	2.09 ± 0.10	2.01 ± 0.23
MIWAEM	2.74 ± 0.05	1.88 ± 0.04	1.92 ± 0.04	0.57 ± 0.03	2.66 ± 0.11	2.55 ± 0.09	2.34 ± 0.51	2.76 ± 0.48	2.06 ± 0.14	1.94 ± 0.23
H-VAEM	2.82 ± 0.06	1.80 ± 0.04	1.89 ± 0.01	0.48 ± 0.06	3.06 ± 0.31	2.74 ± 0.09	2.42 ± 0.21	2.85 ± 0.56	1.72 ± 0.11	1.89 ± 0.24
HMC-VAEM	2.69 ± 0.05	1.77 ± 0.06	1.89 ± 0.02	0.49 ± 0.07	$\mathbf{2.21 \pm 0.24}$	2.72 ± 0.20	2.28 ± 0.29	2.83 ± 0.46	1.73 ± 0.05	1.83 ± 0.16
HH-VAEM	$\mathbf{2.63 \pm 0.04}$	$\mathbf{1.75 \pm 0.03}$	$\mathbf{1.88 \pm 0.05}$	$\mathbf{0.40 \pm 0.05}$	2.47 ± 0.27	$\mathbf{2.54 \pm 0.13}$	$\mathbf{2.28 \pm 0.09}$	$\mathbf{1.90 \pm 0.17}$	$\mathbf{1.71 \pm 0.04}$	$\mathbf{1.83 \pm 0.11}$

Table 5.1: Test NLL of the unobserved features for our model and baselines.

	Bank	Insurance	Avocado	Naval	Yatch	Diabetes	Concrete	Wine	Energy	Boston
VAEM	0.56 ± 0.06	1.20 ± 0.03	1.18 ± 0.02	2.69 ± 0.01	0.61 ± 0.02	1.59 ± 0.19	1.07 ± 0.09	0.28 ± 0.09	0.61 ± 0.14	0.85 ± 0.21
MIWAEM	0.51 ± 0.03	1.15 ± 0.03	1.15 ± 0.03	2.70 ± 0.01	0.60 ± 0.03	$\mathbf{1.36 \pm 0.10}$	0.95 ± 0.22	0.28 ± 0.13	0.54 ± 0.12	0.80 ± 0.21
H-VAEM	0.50 ± 0.03	1.06 ± 0.02	1.18 ± 0.02	2.68 ± 0.01	0.60 ± 0.02	1.71 ± 0.14	1.02 ± 0.09	0.26 ± 0.11	0.46 ± 0.14	0.90 ± 0.22
HMC-VAEM	0.52 ± 0.02	1.00 ± 0.03	1.12 ± 0.03	2.71 ± 0.01	$\mathbf{0.52 \pm 0.15}$	1.55 ± 0.29	0.95 ± 0.26	0.28 ± 0.09	0.41 ± 0.07	0.71 ± 0.13
HH-VAEM	$\mathbf{0.49 \pm 0.03}$	$\mathbf{0.93 \pm 0.06}$	$\mathbf{1.10 \pm 0.01}$	$\mathbf{2.62 \pm 0.01}$	0.56 ± 0.02	1.38 ± 0.18	$\mathbf{0.95 \pm 0.08}$	$\mathbf{0.20 \pm 0.04}$	$\mathbf{0.32 \pm 0.05}$	$\mathbf{0.55 \pm 0.04}$

Table 5.2: Test NLL of the predicted target for our model and baselines.

Concretely, the reward in (5.11) can be rewritten as

$$D_{\mathrm{KL}}\left[p(\boldsymbol{y}, x_i|\boldsymbol{x}_O)||p(\boldsymbol{y}|\boldsymbol{x}_O)p(x_i|\boldsymbol{x}_O)\right] = \mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O), \tag{5.12}$$

While a set of advanced non-parametric estimators of the mutual information are available (Kraskov et al., 2004; Ross, 2014), many are not easilly adapted for parallelization. We demonstrate that the simplest one,

$$\hat{I}(\boldsymbol{y}; \boldsymbol{x}_i\ |\boldsymbol{x}_O) \approx \sum_{ij} p(i,j) \log \frac{p(i,j)}{p_x(i)p_y(j)}, \tag{5.13}$$

based on binning the x_i and $\boldsymbol{y}$ domains, is effective and easy to parallelize. In the equation, $p_x(i) = n_x(i)/N$, $p_y(j) = n_y(j)/N$ and $p(i,j) = n(i,j)/N$ are the relative frequencies that approximate $p(x)$, $p(y)$ and $p(x,y)$ for each bin. Thus, $n_x(i)$, $n_y(j)$ and $n(i,j)$ are the number of samples inside each interval. The number of bins defines the width of uniformly distributed intervals over x_d and $\boldsymbol{y}$ supports. Since this estimator is sampling-based, under certain conditions (namely, if all densities exist as proper functions), (5.13) indeed converges to $\mathcal{I}(\boldsymbol{y}; x_i\ |\boldsymbol{x}_O)$ if we first let the number of samples $N \to \infty$ (Kraskov et al., 2004).

5.4. Experiments

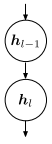
The evaluation of the HH-VAEM model is organized into three quantitative experiments and one qualitative experiment. The ablation study includes the validation of our proposed HMC-based, hierarchical model with respect to the Gaussian and one-layered alternatives. Namely, the comparison is performed with the following baseline models:

- *VAEM*: 1 layer, Gaussian-based VAEM (Ma et al., 2020) (without including the Partial VAE).
- *MIWAEM*: 1 layer, Gaussian-based, importance weighted IWAEM (VAEM + IWAE (Mattei and Frellsen, 2019)).
- *H-VAEM*: 2 layers, Gaussian-based VAEM.
- *HMC-VAEM*: 1 layer, HMC-based (with our optimization method) VAEM.

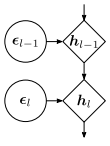
For all the models, we manually introduce missing data in the training set by randomly setting per data point a feature as missing with a probability sampled uniformly in the

interval $[0.01, 0.99]$ within each batch. Both the input data $\boldsymbol{x}$ and the target $\boldsymbol{y}$ can be missing. For the test set, a fixed probability of 0.5 leads to about half of the input data being observed, whilst the target is completely unobserved.

For the quantitative experiments, a total of 10 UCI datasets (Dua et al., 2017) that include mixed-type data are employed for the evaluation. We include both MNIST (LeCun, 1998) and Fashion-MNIST (Xiao et al., 2017) datasets for evaluating our model in higher dimensional observational and latent spaces and bigger architectures (3 layered convolutional nets for encoder/decoder). For the qualitative results, we evaluate our model in the image inpainting task on MNIST and CelebA (Liu et al., 2015). For the three image datasets, the marginal VAEs are not included and the dependency VAE is fed directly with the Bernoulli-distributed pixels. We name this model HH-VAE, and similarly, the baselines are renamed as VAE, MIWAE, HMC-VAE and H-VAE. Extended experiments and validations are provided in the Appendix. The source code for reproducing our work is available at https://github.com/ipeis/HH-VAEM.



(a) Autoregressive



(b) Reparameterization

Figure 5.4: AR (a) vs reparameterized (b).

	VAE	MIWAE	H-VAE	HMC-VAE	**HH-VAE**
MNIST	0.124 ± 0.001	0.121 ± 0.001	0.119 ± 0.001	0.101 ± 0.004	$\mathbf{0.094 \pm 0.003}$
F-MNIST	0.162 ± 0.002	0.160 ± 0.002	0.156 ± 0.002	0.150 ± 0.002	$\mathbf{0.144 \pm 0.002}$

Table 5.3: Test NLL of the unobserved features for the MNIST datasets.

	VAE	MIWAE	H-VAE	HMC-VAE	**HH-VAE**
MNIST	0.153 ± 0.009	0.151 ± 0.007	0.146 ± 0.006	0.067 ± 0.007	$\mathbf{0.056 \pm 0.019}$
F-MNIST	0.501 ± 0.012	0.496 ± 0.008	0.494 ± 0.007	0.357 ± 0.060	$\mathbf{0.337 \pm 0.069}$

Table 5.4: Test NLL of the predicted target for the MNIST datasets.

	VAE	MIWAE	H-VAE	HMC-VAE	**HH-VAE**
MNIST	0.953 ± 0.004	0.953 ± 0.003	0.953 ± 0.003	0.978 ± 0.003	$\mathbf{0.981 \pm 0.005}$
F-MNIST	0.824 ± 0.005	0.824 ± 0.004	0.824 ± 0.004	0.869 ± 0.015	$\mathbf{0.876 \pm 0.017}$

Table 5.5: Test accuracy of the predicted digits for the MNIST datasets.

5.4.1. Experimental setup

Apart from the MNIST, Fashion-MNIST and CelebA datasets, a total of 10 UCI datasets have been employed in this work including: Bank Marketing, Insurance Company Benchmark, Avocado sales, Naval Propulsion Plants, Yatch Hydrodynamics, Diabetes, Boston Housing, Wines, Energy efficiency and Bank Marketing.

The networks for the encoder of the model with the MNIST datasets are 2 layered Deep CNNs with {16, 32, 32} output channels, kernel size 5, stride 2 and padding 2. For the experiments with CelebA, we use 5 layered Deep CNNs with {32, 32, 64, 64, 512} output channels, kernel size 4, stride 2 and padding 1, followed by batch norm layers. They are followed by MLPs with 512 hidden units for obtaining the variational parameters for each layer. The decoder that obtains $p_\theta(\boldsymbol{x}|\boldsymbol{h}_1)$ is the symmetric CNN. All the NNs employed in the models trained with UCI datasets are one single layer MLPs with 256 hidden units. The noise variance for Gaussian likelihoods is set up to 0.1.

We employ learning rates of 1×10^{-3} for the models with MLP networks and 2×10^{-4} for the convolutional models. For the inflation parameter $\boldsymbol{s}$, we increase to 1×10^{-2} for a faster convergence. A batch size of 100 is used for all the models except for

Yatch and Wine dataset, where we use 20 samples per batch. The number of training steps is 2×10^4 for Boston, Energy, Wine, Yatch, Concrete, Diabetes and Yatch, and 5×10^4 for Naval, Avocado, Bank, and Insurance. For MNIST and Fashion-MNIST, we have 1×10^5 training steps. For CelebA, we use $1,5 \times 10^5$ training steps. For the marginal VAEs stage, we employ 1×10^3 training steps. The dimension of the latent variables is $[d_1 = 10, d_2 = 5]$ for Boston, Energy, Wine, Naval Avocado, Bank and Insurance $[d_1 = 4, d_2 = 2]$ for Concrete, Yatch and Diabetes, $[d_1 = 20, d_2 = 10]$ for MNIST and Fashion-MNIST, and $[d_1 = 32, d_2 = 16]$ for CelebA.

We use $LF = 5$ Leapfrog steps in all cases, chains of $T = 10$ for Boston, Energy, Wine, Naval Avocado, Bank, Insurance, MNIST, Fashion-MNIST and CelebA, and $T = 5$ for Concrete, Yatch and Diabetes. The SKSD function is estimated using 30 HMC samples.

For the MNIST and CelebA datasets, the use of Nvidia P100 GPU with Pascal architecture sped up the training with the CNN-based architecture. For the UCI datasets, due to the use of small networks, the differences when using CPU or GPU are almost imperceptible.

5.4.2. Mixed type conditional data imputation

In order to evaluate the performance of the model in terms of data imputation, we opt by computing the negative log likelihood of the unobserved features. We make use of the Monte Carlo approximation

$$\log p(\boldsymbol{x}_U|\boldsymbol{x}_O) \approx \log \mathbb{E}_{\boldsymbol{\epsilon}\sim q^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{x}_O)}\left[p(\boldsymbol{x}_U|\boldsymbol{\epsilon})\right] \approx \log \frac{1}{k}\sum_{i}^{k} p(\boldsymbol{x}_U|\boldsymbol{\epsilon}_i), \tag{5.14}$$

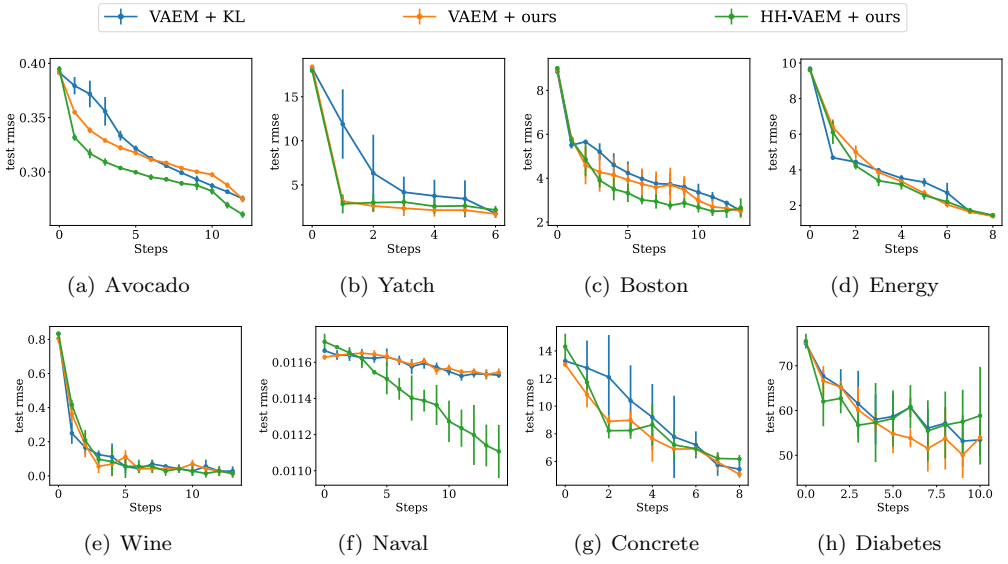
which is averaged over features in order to compare the imputation performance with the baselines. Additionally, we include in Section B.1.5 similar results averaging each of the considered likelihoods. Results on the 10 UCI datasets and the MNIST datasets are included in Tables 5.1 and 5.3, showing that for most of the datasets, incremental improvement is obtained: VAEM < H-VAEM < HMC-VAEM < HH-VAEM. Extended results with the imputation error, included in Appendix B.1.3, corroborate this.

5.4.3. Target prediction

For this experiment, we compute the negative log likelihood of the target under the predictive distribution using the same procedure as in the previous section. Results included in Tables 5.2 and 5.4 show the same incremental improvement in the prediction task.

5.4.4. Sequential active information acquisition (SAIA)

In this experiment, our HH-VAEM model and our acquisition method are evaluated in a SAIA task. Starting by predicting from completely unobserved inputs, at each step, the missing feature that maximizes the reward is acquired. Figure 5.5 shows the error curves for the UCI datasets. Blue lines correspond to the Gaussian-based reward proposed by (Ma et al., 2020). Orange lines are our sampling-based reward in a VAEM framework. Green lines correspond to HH-VAEM with our reward. In most of the cases, our model and acquisition method obtains lower errors and faster discovery of information.



(a) Avocado (b) Yatch (c) Boston (d) Energy

(e) Wine (f) Naval (g) Concrete (h) Diabetes

Figure 5.5: SAIA curves. Horizontal axis shows number of discovered features. Vertical axis is RMSE.

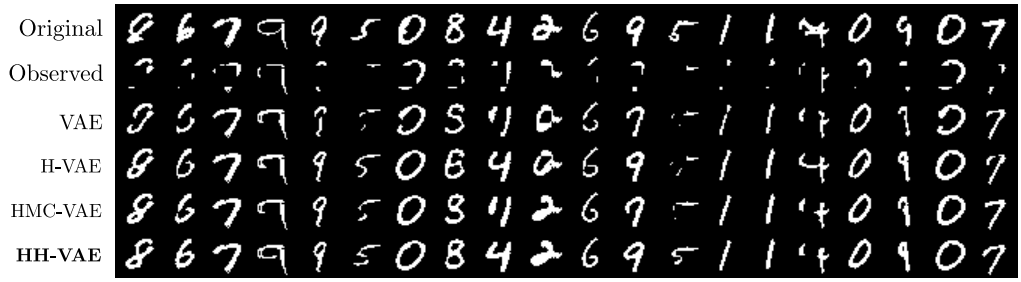
5.4.5. Conditional image inpainting

We include in this experiment qualitative results when comparing our model with the baselines on the image conditional inpainting task. We include results for MNIST and CelebA in Figure 5.6 (a) and (b), respectively, that show the superiority of our method. First, the hierarchical Gaussian model (fourth row) considerably improves the one-layered Gaussian alternative. Second, the HMC-based methods (two last rows) vastly improve the Gaussian methods. Third, in some specific cases, an extra improvement is added by HH-VAE with respect to the one-layered HMC-based model (columns 2, 4, 8, 9, 11, 12 and 13 in Figure 5.6 (a) or columns 1, 2, 5 and 12 in (b)).

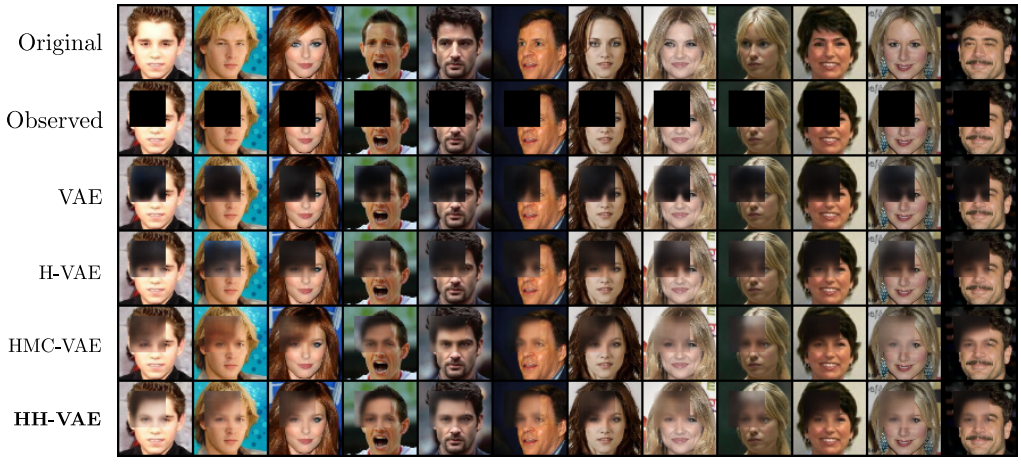
5.4.6. Efficacy of HMC optimization

We include in this experiment results that demonstrate the efficacy of training the HMC hyperparameters using our proposed gradient-based strategy on 2D densities. In Figure 5.7, each row correspond to a different example: first row is a *wave* density, second row is a *dual moon* density. In the first column, the initial set up is showed, including the density contour (dark blue), the initial Gaussian proposal contour (light blue) and samples from HMC (green). In both cases, due to the tightness of the proposal, chains do not properly explore the density and get stuck close the initial state. In the second column, results after training HMC hyperparameters and the inflation parameter are included. Again, in both cases, and more specially in the first one, the inflation of the horizontal variance of the proposal successfully increases to better cover the surface, and final HMC samples vastly improve the exploration. In the third column, the approximations of the HMC objective, the SKSD discrepancy and the inflation parameters over the optimization steps are included.

In Figure 5.8 (a), the mean acceptance rate of the HMC sampler over the training steps and the step sizes (b) are included. The steps are initialized from $U(0.05, 0.2)$. After 2×10^3 steps, the mean acceptance rate converges to a value closer to $\bar{p}_a = 0.65$, which is



(a) MNIST



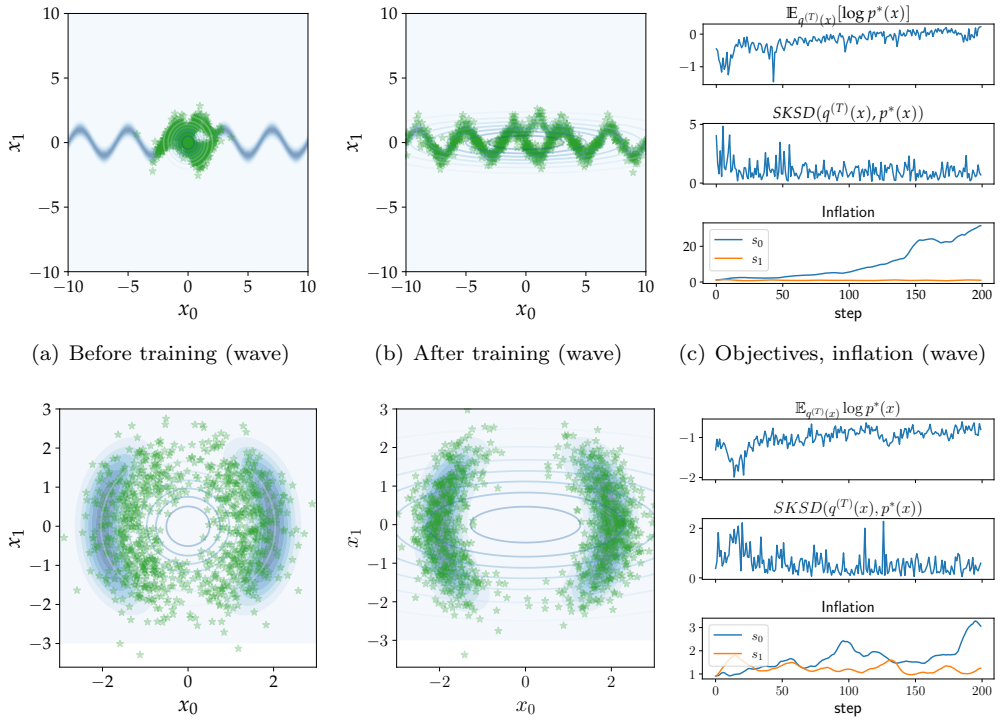
(b) CelebA

Figure 5.6: Image conditional inpainting on MNIST (a) and CelebA (b). First row: original images from the test set. Second row: input to the model, $\boldsymbol{x}_O$, with a black square missing mask $\boldsymbol{x}_U$ manually introduced. Third, fourth, fifth and sixth rows: imputed $\hat{\boldsymbol{x}}_U$ for VAE, H-VAE, HMC-VAE and HH-VAE, our model, where $\hat{x}_U$ is decoded from samples of the approximate posterior (Gaussian for VAE and H-VAE, or HMC-based for HMC-VAE and HH-VAE).

defined as the optimal desired acceptance probability (Neal et al., 2011). This empirical result provides evidence that, apart from reducing the computational cost, reducing the HMC training step to this value is sufficient for achieving convergence.

5.5. Conclusion

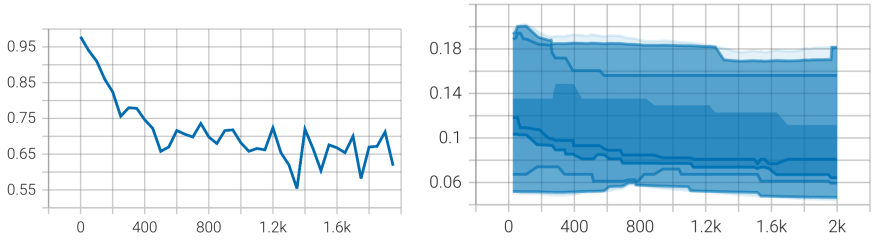
We presented HH-VAEM, to our knowledge, the first hierarchical VAE for mixed-type incomplete data that uses HMC with automatic hyper-parameter tuning for improved inference. We provide both quantitative and qualitative experiments that demonstrate its superiority with respect the baselines in the tasks of missing data imputation and supervised learning, placing HH-VAEM as a robust model for real-world datasets. Further, we have developed a novel sampling-based technique for dynamic feature selection that outperforms the Gaussian-based alternatives and results in an efficient method for active learning in deep generative models.



(a) Before training (wave) (b) After training (wave) (c) Objectives, inflation (wave)

(d) Before training (dual moon) (e) After training (dual moon) (f) Objectives, inflation (dual moon)

Figure 5.7: Toy example showing the efficacy of training HMC hyperpameters using our method on two densities: wave (top row) with $T = 5$ and dual moon (bottom row) with $T = 10$. Left column illustrates the non desired behavior when chains hardly explore the density and stuck in a small region close to the mass of the tight Gaussian initial proposal (light blue contour ellipses). Right column shows how optimizing the step sizes and the inflation parameter leads to a vast improvement of the exploration. More specifically, (b) justifies scaling each dimension of the target, since the inflation is bigger on the horizontal axis.



(a) Mean acceptance rate $\bar{p}_a$ (b) Step sizes

Figure 5.8: Evolution of mean acceptance rate $\bar{p}_a$ (a) and step sizes (b) over the training steps.

CHAPTER 6

Conclusions and Future Work

The preceding chapters have meticulously examined the two domains of study that have been taken into account in this research. Chapters 2 and 3 laid the groundwork for the contributions by reviewing the literature on Deep Generative Models from a broader perspective and Variational Autoencoders from a more specific standpoint. In Chapter 4, the initial primary contribution was put forth, consisting on the development of a novel method for learning global meaningful representations in a fully unsupervised setting. In Chapter 5, the second principal contribution was presented, which involved the creation of a model with heightened expressiveness and superior inference performance, resulting in more accurate acquisition of unobserved data. This final chapter summarizes the key findings of the doctoral thesis, offering a comprehensive overview of the outcomes and an extensive presentation of the principal points for future research development.

6.1. Summary of models and contributions

Variational Autoencoders are widely used and have achieved impressive results in recent years. However, there is still room for methodological improvement. In this doctoral thesis, two critical issues were successfully addressed. One of them is the problem of learning meaningful or *disentangled* representations, which has been previously tackled by incorporating additional information into the data with some semi-supervision or adding new factors to the ELBO to assess the level of disentanglement. However, the former unrealistically assumes the availability of additional information, and the latter requires additional modifications of the objective, which departs from the pure theoretical definitions of the ELBO. In contrast, the UG-VAE presented in this thesis demonstrates that model design is a powerful technique for achieving meaningful latent representations of the observed data. By designing a global encoder that allows for permutations, batches of images can be compressed into a global latent representation. This global concept is shared by a group of samples and induces the probabilities of a mixture model for the prior of the local latent space, affecting every single sample differently. The experimental results demonstrate that the model can effectively perform inference, allowing to split global-local concepts within the set of two latent variables. The results provide evidence of promising improvements in generation quality, diversity, interpretability of the learned representations, domain alignment, and expressivity, highlighting the significance of this contribution.

Overall, the principal contributions presented in Chapter 4 are:

- To our knowledge, we propose the first deep generative model for generating groups of samples with shared properties learned in a fully-unsupervised fashion, named UG-VAE.

- We demonstrate that the information captured in the structured latent space of UG-VAE is highly interpretable in comparison with other related methods, leading to an improved disentanglement in both local and global spaces.

- We demonstrate that, by simply training UG-VAE with minibatches of samples from several datasets, the structured latent space aligns them and captures common interpretable properties without any label or supervision.

- We demonstrate that, although the training unsupervised, the global space is able to effectively separate the global posterior of different groups when weak supervision is included at test time for grouping observations with a given label or attribute.

The second part of the thesis focuses on solving different tasks using the same family of models, namely missing data imputation and data acquisition. Both tasks are highly dependent on the accuracy of the model's posterior approximation, as predictive distributions are decoded from latent samples drawn from the learned posterior. To address this, a powerful inference technique is presented that combines HMC and VAEs. Additionally, to enable the model to learn from more complex datasets, a hierarchical structure of latent variables is used, but this comes with the drawback of increasing the complexity of the approximate inference in such a complicated posterior density. However, by proposing an efficient reparameterization method and jointly tuning its hyperparameters, HMC is successfully integrated into the Hierarchical VAE. The results, both qualitative and quantitative, demonstrate the superiority of the hierarchical approach and HMC-based inference in achieving superior performance in both imputation and acquisition tasks. In addition, a new contribution is introduced that leverages the advantages of HH-VAEM in sampling by proposing a sampling-based scheme for approximating the reward function of acquiring unobserved features, leading to a considerable improvement over alternatives as demonstrated in the SAIA experiments.

To conclude, the main contributions presented in Chapter 5 are as follows:

- We present **HH-VAEM**, a deep hierarchical model for handling mixed-type incomplete data that uses HMC with automatic hyper-parameter tuning for outperforming amortized variational inference by generating low bias samples from the true posterior.

- We propose a sampling-based strategy for missing feature acquisition that benefits from the improved inference of HH-VAEM. By using histograms to estimate the mutual information, this strategy achieves lower bias than other Gaussian-based alternatives.

- We exhaustively evaluate HH-VAEM in the tasks of 1) missing data imputation, 2) supervised learning with missing data and 3) information acquisition with our sampling-based strategy. In all cases we report significant gains with respect to baselines.

6.2. Future research

For both two contributions, the following potential lines of future research can be considered. They are classified into technical or applied research in different sections.

6.2.1. Technical research

Technical research often entails developing innovative modeling strategies, which may involve combining insights from the presented studies in chapters 4 and 5 or pursuing the natural progression of each individual contribution.

Global factors in sequential data

One of the research projects that preceeded the present doctoral thesis was focused in developing deep neural models for sequential data (Peis et al., 2019). This line could leverage the findings of this doctoral thesis by considering entire sequences as local datapoints. Finding interpretable global patterns shared among sequences could help in better describing sequencial data from multiple sources, like Electronical Health Records from a patient population, audio recordings from different speakers, or stock prices from different companies, to list some examples. A comparative migh be the way Latent Dirichlet Allocation (LDA, (Blei et al., 2003)) finds global topics in a text corpus within a unsupervised setting. In the UG-VAE setting, both local discrete "topics", and global continuous generative factors could be captured. Recent works like (Li and Mandt, 2018) have shown promising results in disentangling sequences. Their model splits into static-dynamic variations, or latent time-dependent features (dynamics) from features which are preserved over time (content). Others build models for learning disentangled representations in music data (Yang et al., 2019; Luo et al., 2020), video (Vowels et al., 2021; Albarracin and Rivera, 2022), sequence-to-sequence (Yang et al., 2022), audio synthesis (Melechovsky et al., 2023), or sequential recommenders.

Hierarchical latent spaces and global factors

In chapter 5, Hierarchical VAEs were considered as a robust model that allows for increased flexibility of the prior, inducing a flow of information from more abstract or general concepts, to more specific features. However, all this presented insights were modeled independently for each datapoint, i.e. they only represent local variations. If findings from UG-VAE (Peis et al., 2023) were combined with this hierarchical setting, it might be possible to find global properties at different levels of abstraction. Mathematically, this disentanglement could be achieved by designing two latent hierarchies $\boldsymbol{\beta} = \{\boldsymbol{\beta}_1, ...\boldsymbol{\beta}_{L_\beta}\}$ at the global level and $\boldsymbol{z} = \{\boldsymbol{z}_1, ..., \boldsymbol{z}_{L_z}\}$ at the local level. By doing this, both the specific and abstract properties could be organized hierarchically. For instance, these two levels of abstraction could refer to style-content in models like (Jing et al., 2019; Bouchacourt et al., 2018; Esmaeili et al., 2019), or different domains in domain alignment (Ilse et al., 2020; Liu et al., 2021b; Heinze-Deml and Meinshausen, 2021).

Domain alignment with global factors

As empirically demonstrated in section 4.3.2, the UG-VAE model allows for efficiently handling domain alignment in an unsupervised setting. Other wors have proposed similar design strategies for modeling data from different domains, like (Ilse et al., 2020), where three independent latent variables are combined to account for domain, class and residual variation factors. More examples can be found in (Nguyen et al., 2021; Liu et al., 2021c).

MCMC for enhancing disentanglement

In this thesis, MCMC methods for obtaining more accurate samples from the true posterior in VAEs have been shown to be successful in the considered tasks, mainly: missing data imputation, reconstruction and information acquisition (Peis et al., 2022). Another research question arises when merging the two contributions: does the approximate inference influence the quality of the disentanglement? Would MCMC methods improve the interpretability of the learned latent space? For addressing these questions, the HMC algorithm can be plugged to UG-VAE to analyze the latent space.

6.2.2. Applied research

Global factors in clinical data

The use of UG-VAE presents a promising avenue for learning meaningful representations of clinical data, with potential applications in the development of diagnostic tools, disease stratification, and personalized treatments. Such applications could have a substantial impact, as the discovery of global factors of variation in clinical populations can be crucial for understanding disease progression and identifying effective interventions. For example, in a previous contribution from the author of this thesis (Peis et al., 2020), a Hierarchical Gaussian Process was used to model two levels of variation - patient-level and population-level - in activity-related measures from depressed inpatients, with the goal of predicting the discharge date. While this approach successfully differentiated the two levels of variation, the use of UG-VAE could capture more complex patterns in a similar way to previous studies (Couronné et al., 2021; Cetin et al., 2023). Furthermore, by incorporating semi-supervision of the patient anonymized identification, as demonstrated in (Bouchacourt et al., 2018), more robust disentanglement could be achieved. Overall, the potential for UG-VAE to uncover hidden patterns in clinical data suggests that further investigation is warranted.

Global factors in recommender systems

Recent studies have shown that Deep Learning methods have the potential to improve the performance of Recommender Systems by learning representations of users and items that capture complex patterns in user-item interactions (Deng et al., 2022; Antognini and Faltings, 2021; Ma et al., 2019b). Within this framework, UG-VAE could be used to model dynamic user preferences over time, which could improve the accuracy of personalized recommendations. Additionally, incorporating side information such as user demographics or item features could further enhance the ability of UG-VAE to capture disentangled user-item interactions. Another potential direction is to explore the interpretability of UG-VAE representations, as this could help users better understand the reasons behind recommendations and improve trust in the system, or providers to reasonate on user preferences. Finally, there is also potential for combining UG-VAE with other machine learning techniques such as reinforcement learning or active learning (like the proposed acquisition method in Peis et al. (2022)) to optimize recommendation strategies.

Efficient clinical data acquisition

As briefly discussed in chapter 5, Active Learning (Lindley, 1956; MacKay, 1992) is a type of Machine Learning in which an algorithm actively selects the most informative data

points to learn from, instead of passively learning from a fixed dataset. This is achieved by selecting data points for which the algorithm is uncertain or has low confidence in its predictions, and requesting additional labels or annotations for those samples. This approach can help reduce the amount of labeled data needed to train a model, and improve its accuracy. Active Learning is particularly useful when labeling large datasets is expensive or time-consuming, and has applications in areas such as clinical applications, natural language processing, computer vision, and bioinformatics.

Specifically, Active Learning is of vital importance in clinical applications. First, medical datasets can be large and complex, and labeling them can be time-consuming, expensive, and potentially error-prone. Active Learning can help reduce the amount of labeled data needed, making it more feasible to train accurate models with limited resources (Budd et al., 2021; Bucklin et al., 2021; Kholghi et al., 2016). Second, accurate modeling of medical data can have significant implications for patient health outcomes. Active Learning can help ensure that models are trained on the most informative data points, leading to more accurate predictions and better treatment recommendations. Finally, Active Learning can also enable the efficient exploration of new treatments and interventions by providing an effective means of identifying the most promising candidates for further investigation.

The applications and advantages of Active Learning can be effectively leveraged by the method of (Peis et al., 2022) presented in chapter 5 due to its robustness, efficiency, and low cost. For instance, in clinical decision-making with incomplete data, the algorithm could identify which unobserved features are worth discovering. This could result in the selection of inexpensive variables, such as simple pervasive tests, that could improve the success of the decision-making process more effectively than complex, expensive, or invasive tests.

APPENDIX A

UG-VAE: ADDITIONAL DETAILS

A.1. Experimental extension

A.1.1. Extended results for Section 4.1: Unsupervised learning of global factors

With the aim at evaluating whether a fraction of the clusters inferred by UG-VAE encode visually interpretable global/local features, in Figure **??** we include the results for CelebA for $K = 20$ clusters. We observe that a considerate proportion of the clusters captures disentangled generative factors. Moreover, considering the heterogeneity and variety in the generative factors of celebA faces (up to 40 different attributes), increasing the number of clusters might lead to capture more representative faces, and thus, generative global factors modulated by $\boldsymbol{\beta}$. In Figure A.2, we appreciate that, apart from skin color, beard or image contrast, other generative factors controlled by the global variable are hair style (remarkable for components 9, 16 , 17 or 18), sex (components 4 and 14), or background color (components 4, 16 and 17). In order to compare these results with a model trained on a small number of clusters, we include Figure A.1 with samples from UG-VAE with $K = 4$. In this case, the model compresses the information of the whole dataset in only four modes, and thus, the variation of the samples within each cluster is higher.

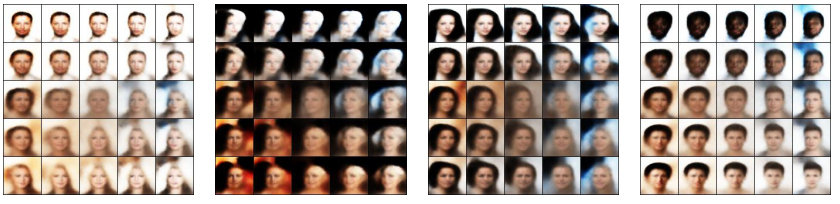
(a) $d = 0$ (b) $d = 1$ (c) $d = 2$ (d) $d = 3$

Figure A.1: Sampling from each cluster of UG-VAE for CelebA when $K = 4$.

A.1.2. Extended results for Section 4.2: Domain Alignment

We include here the results of a interpolation in both the local space obtained when the number of components is $K = 1$, i. e., using the ML-VAE approach. As showed in Figure A.3, when training ML-VAE with randomly grouped data, global space is not capable of capturing correlations between datasets, and the local space is in charge of encoding the transition from celebA to 3D FACES, which is performed within each row.

(a) $d = 0$ (b) $d = 1$ (c) $d = 2$ (d) $d = 3$



(e) $d = 4$ (f) $d = 5$ (g) $d = 6$ (h) $d = 7$

(i) $d = 8$ (j) $d = 9$ (k) $d = 10$ (l) $d = 11$

(m) $d = 12$ (n) $d = 13$ (ñ) $d = 14$ (o) $d = 15$

(p) $d = 16$ (q) $d = 17$ (r) $d = 185$ (s) $d = 185$

Figure A.2: Sampling from UG-VAE for CelebA. We include samples from each of the K = 20 clusters.

Figure A.3: ML-VAE interpolation in local (columns) and global (rows) posterior spaces, fusing celebA and FACES datasets

With the aim at reinforcing the robustness of UG-VAE in domain alignment, we include in Figure A.4 the results of evaluating GMVAE with two clusters ($K = 2$) in a similar setup that in section 4.2. As GMVAE does not have global variables, the interpolation only applies for the latent encodings in $\mathbf{z}$. Note that the interpolation is merely a gradual overlap between the two images. Namely, the model is not able to correlate the features of both images, regardless of their domain. On the other hand, with UG-VAE, by keeping fixed the global variable and interpolating in the local one, we maintain the domain but we translate the features of one image into the other. This analysis corroborates that the model finds this type of correlations in a clearly separated way.

Figure A.4: Interpolation in the latent space of GMVAE with $K = 2$ for performing domain alignment, using the same network architecture than in the local part of UG-VAE. We interpolate between the encodings of images from CelebA and FACES dataset.

A.2. Networks architecture

In this section we detail the architectures and parameters used for training the models exposed in the main paper. An extended overview is included in Table A.1.

Table A.1: Architecture, parameters and hyperparameters for all the models trained for the experiments presented in the paper.

Dataset	Architecture				Params	Hyperparams
	Pre-encoder	Local encoder	Global encoder	Decoder		
CelebA	**h**: 5 CNN layers Filters: 32, 32, 64, 64, 256 Stride: all 4 Padding: All 1 ReLU activation Batch normalization	ϕ_z: Linear layer: $256 \to 2d$ First half $\boldsymbol{\mu}_z$ Second half diag($\boldsymbol{\Sigma}_z$) ϕ_d: Linear layers: $d \to 256 \to K$ Tanh activation Softmax output	ϕ_B: Linear layer: $256 + K \to 2g$ First half $\boldsymbol{\mu}_B$ Second half diag($\boldsymbol{\Sigma}_B$)	θ_z: Linear layers: $g \to 256 \to 2d$ First half $\boldsymbol{\mu}_z$ Second half diag($\boldsymbol{\Sigma}_z$) θ_x: Linear layer: $d + g \to 256$ 5 transpose CNN layers Filters: 64, 64, 32, 32, 3 Stride: 1, 4, 4, 4, 4 Padding: 0, 1, 1, 1, 1 ReLU activation Sigmoid output	d=20 $g = 50$ $K = 20$	$\sigma_x = 0.2$ $B = 128$
MNIST	**h**: Linear layer: $28 * 28 \to 256$ ReLU activation	ϕ_z: Linear layer: $256 \to 2d$ First half $\boldsymbol{\mu}_z$ Second half diag($\boldsymbol{\Sigma}_z$) ϕ_d: Linear layers: $d \to 256 \to K$ Tanh activation Softmax output	ϕ_B: Linear layer: $256 + K \to 2g$ First half $\boldsymbol{\mu}_B$ Second half diag($\boldsymbol{\Sigma}_B$)	θ_z: Linear layers: $g \to 256 \to 2d$ First half $\boldsymbol{\mu}_z$ Second half diag($\boldsymbol{\Sigma}_z$) θ_x: Linear layers: $d + g \to 256 \to 28 * 28$ ReLU activation Sigmoid output	$d = 10$ $g = 20$ $K = 10$	$\sigma_x = 0.2$ $B = 128$
CelebA + 3D FACES	Same than for CelebA				$d = 40$ $g = 40$ $K = 40$	$\sigma_x = 0.2$ $B = 128$
3D Cars-3D Chairs	Same than for CelebA				$d = 20$ $g = 20$ $K = 20$	$\sigma_x = 0.2$ $B = 128$
3D Cars-Cars	Same than for CelebA				$d = 20$ $g = 50$ $K = 20$	$\sigma_x = 0.2$ $B = 128$

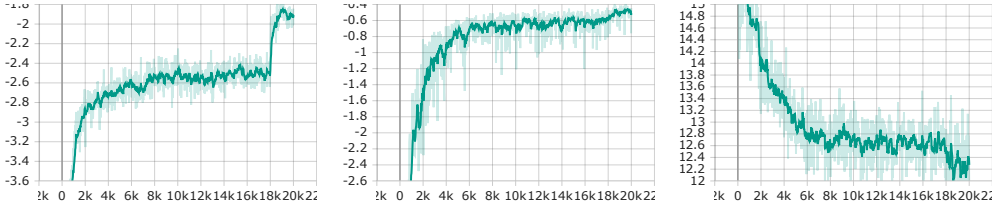
APPENDIX B

HH-VAEM: ADDITIONAL DETAILS

B.1. Extended experiments

B.1.1. Efficiently incorporating HMC

We face the computational cost of running HMC by defining a small percentage of training steps for the last stage in Algorithm 1. A 10% of the total training steps for T_{HMC} is sufficient for obtaining the convergence. In Figure B.1 we include the validation metrics obtained during the optimization of Yatch dataset, where $T_{HMC} = 2 \times 10^3$.



(a) $\log p(\boldsymbol{x}_U|\boldsymbol{x}_O)$ (b) $\log p(\boldsymbol{y}|\boldsymbol{x}_O)$ (c) RMSE

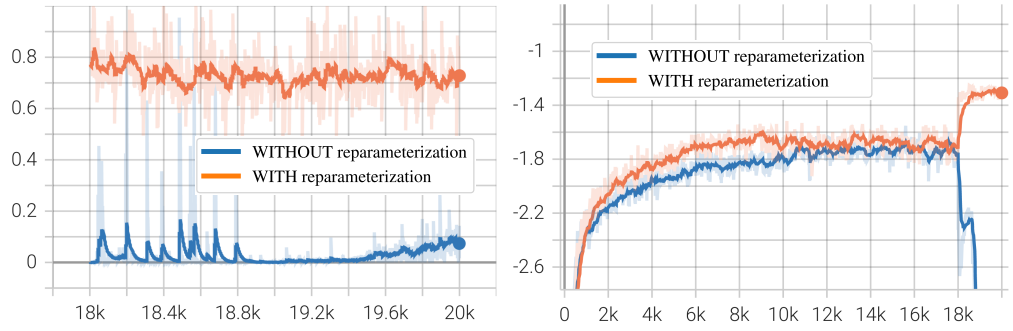
Figure B.1: Validation curves during optimization for Naval dataset.

B.1.2. Reparameterization trick for solving ill-posed HMC with hierarchical densities

We provide in this section strong empirical demonstration on the efficacy of our proposed reparameterization trick. As stated in Section 5.2.4, naïve implementations of HMC are ill-posed when combined with hierarchical densities. The autoregressive correlations lead to non-smooth densities with huge peaks. The large gradients $\nabla_{\boldsymbol{h}_{1:L}} \log p^*(\boldsymbol{h}_{1:L})$ evaluated on these regions inside the Leapfrog steps of 5.1 make huge modifications of the proposed states. If we denote these diverged states by $\boldsymbol{h}_{1:L}^{(ill)}$, evaluating the objective $\log p^*(\boldsymbol{h}_{1:L}^{(ill)})$ lead to overflow issues. This undesired behavior is what we call *divergence of the Leapfrog integrator*, and is also illustrated in Figure 35 of (Betancourt, 2017).

In order to avoid the aforementioned overflow issues, we can directly reject these problematic states. Nevertheless, by doing this, HMC will not properly explore the density by the time the states fall into the problematic regions, leading to extremely low acceptance rates. To demonstrate this, we include in Figure B.2 (a) with blue line the evolution of the acceptance rates when optimizing HMC with the HH-VAEM variant without reparameterization (as illustrated in Figure 5.4 (a)). The acceptance rate is extremely low as expected, which is a clear indicator of a poorly mixing sampler.

On the contrary, by using our proposed reparameterization trick (Figure 5.4 (b)), we are able to make HMC work properly and tune the step sizes, getting closer to the ideal



(a) Mean acceptance rate $\bar{p}_a$

(b) $\log p(\boldsymbol{x}_U|\boldsymbol{x}_O)$

Figure B.2: Demonstration of the efficacy of our reparameterization trick. Our model is showed with orange lines, and the same model without the reparameterization with blue lines. In (a), the mean acceptance rate of the HMC proposals over the optimization steps is included. We demonstrate that without the reparameterization, HMC is ill-posed and by rejecting the proposals that make the integrator diverge, the acceptance rate is extremely low, thus not properly exploring the density. In (b), the imputation log likelihood metric is showed for the whole optimization. With the reparameterization trick, we successfully solve the pathological issues, leading to a considerable increase of the metric.

acceptance rate $\bar{p}_a = 0.65$ (Neal et al., 2011). Evaluations are applied to the same validation split of the Boston dataset. The step sizes of HMC are initialized from $U(0.05, 0.2)$ in both cases.

Further, we also include in Figure B.2 (b) the evolution of the imputation log likelihood metric of (5.14) during the whole optimization for both approaches. First $1,8 \times 10^4$ steps correspond to the pretraining stage using only the ELBO. When introducing HMC without the reparameterization, due to the low acceptance rate, the states hardly move from the initial proposal, or move away from the density, and the joint optimization fail. We demonstrate again the effectiveness of the reparameterization trick by observing an increase in this metric.

B.1.3. Deterministic imputation metrics

We include in Table B.1 results on the RMSE obtained with other discriminative validated predictors, using mean imputation under the same missing rates. Additionally, we include here the missForest in the baselines, a wide-spread method for missing data imputation using a Random Forest approach (Stekhoven and Bühlmann, 2012). For classification tasks, the error rate is considered. In almost all cases, HH-VAEM outperforms the baselines.

	Bank	Insurance	Avocado	Naval	Yatch	Diabetes	Concrete	Wine	Energy	Boston
missForest	0.64 ± 0.01	0.61 ± 0.06	0.59 ± 0.02	0.30 ± 0.01	$\mathbf{0.86 \pm 0.12}$	0.76 ± 0.08	0.76 ± 0.07	0.77 ± 0.11	0.64 ± 0.08	0.59 ± 0.06
VAEM	0.57 ± 0.01	0.40 ± 0.01	0.59 ± 0.00	0.33 ± 0.01	0.93 ± 0.04	0.79 ± 0.02	0.74 ± 0.04	0.64 ± 0.03	0.75 ± 0.02	0.65 ± 0.02
MIWAEM	0.56 ± 0.00	0.39 ± 0.00	0.59 ± 0.01	0.34 ± 0.01	0.94 ± 0.04	0.75 ± 0.01	0.71 ± 0.03	0.63 ± 0.02	0.75 ± 0.02	0.63 ± 0.01
H-VAEM	0.56 ± 0.01	0.39 ± 0.00	0.58 ± 0.01	0.32 ± 0.01	0.92 ± 0.05	0.77 ± 0.01	0.71 ± 0.02	0.60 ± 0.04	0.55 ± 0.03	0.59 ± 0.01
HMC-VAE	0.55 ± 0.01	$\mathbf{0.38 \pm 0.00}$	0.58 ± 0.01	0.30 ± 0.02	0.91 ± 0.05	0.76 ± 0.03	0.71 ± 0.02	0.60 ± 0.02	0.55 ± 0.02	0.57 ± 0.02
HH-VAEM	$\mathbf{0.54 \pm 0.01}$	$\mathbf{0.38 \pm 0.00}$	$\mathbf{0.57 \pm 0.00}$	$\mathbf{0.29 \pm 0.02}$	0.90 ± 0.00	$\mathbf{0.75 \pm 0.01}$	$\mathbf{0.70 \pm 0.01}$	$\mathbf{0.59 \pm 0.04}$	$\mathbf{0.54 \pm 0.01}$	$\mathbf{0.56 \pm 0.03}$

Table B.1: Test RMSE of the unobserved features for our model and baselines.

B.1.4. Likelihood of the observed features

We include in this section results on the negative log likelihood of the observed features

$$\log p(\boldsymbol{x}_O) \approx \log \mathbb{E}_{\boldsymbol{\epsilon} \sim q^{(T)}(\boldsymbol{\epsilon}|\boldsymbol{x}_O)} \left[p(\boldsymbol{x}_O|\boldsymbol{\epsilon}) \right] \approx \log \frac{1}{k} \sum_{i}^{k} p(\boldsymbol{x}_O|\boldsymbol{\epsilon}_i). \tag{B.1}$$

Results are included in Table B.2. In almost all the cases, we confirm the incremental superiority when adding each part of our proposed design.

	bank	insurance	avocado	naval	yatch	diabetes	concrete	wine	energy	boston
VAEM	0.51 ± 0.05	0.99 ± 0.05	0.44 ± 0.01	0.21 ± 0.01	0.62 ± 0.13	0.92 ± 0.12	0.63 ± 0.18	0.73 ± 0.18	1.86 ± 0.09	0.56 ± 0.11
MIWAEM	0.63 ± 0.02	1.06 ± 0.03	0.60 ± 0.03	0.33 ± 0.01	0.75 ± 0.07	1.05 ± 0.06	0.76 ± 0.09	0.80 ± 0.06	1.77 ± 0.15	0.67 ± 0.03
H-VAEM	0.40 ± 0.04	0.93 ± 0.04	0.42 ± 0.05	0.19 ± 0.07	0.58 ± 0.09	0.70 ± 0.13	0.53 ± 0.18	0.71 ± 0.15	0.38 ± 0.02	0.49 ± 0.07
HMC-VAE	0.37 ± 0.07	$\mathbf{0.92 \pm 0.04}$	0.39 ± 0.06	0.18 ± 0.05	0.54 ± 0.10	$\mathbf{0.68 \pm 0.07}$	0.49 ± 0.22	$\mathbf{0.55 \pm 0.07}$	0.40 ± 0.06	0.41 ± 0.04
HH-VAEM	$\mathbf{0.33 \pm 0.03}$	0.95 ± 0.05	$\mathbf{0.36 \pm 0.01}$	$\mathbf{0.17 \pm 0.04}$	$\mathbf{0.45 \pm 0.04}$	0.68 ± 0.16	$\mathbf{0.40 \pm 0.16}$	0.64 ± 0.17	$\mathbf{0.37 \pm 0.06}$	$\mathbf{0.41 \pm 0.04}$

Table B.2: Test NLL of the observed features for our model and baselines.

B.1.5. Heterogeneous likelihoods

In experiment 5.4.2 we reported an average likelihood across heterogeneous variables. Although this quantity is not a valid joint likelihood probability, we employed this average to provide a fair comparison on models that have been trained on the same heterogeneous likelihoods. In this section, we show the comparison on averaging separately the three considered marginal likelihoods (Gaussian, Bernoulli and Categorical) for two of the biggest datasets considered on Tables B.3-B.5. Again, we show the incremental superiority when adding the different design choices of our model.

	Bank	Avocado
VAEM	0.36 ± 0.29	0.26 ± 0.08
MIWAEM	0.33 ± 0.27	0.32 ± 0.06
H-VAEM	0.26 ± 0.22	0.29 ± 0.07
HMC-VAEM	0.25 ± 0.21	0.25 ± 0.08
HH-VAEM	$\mathbf{0.20 \pm 0.22}$	$\mathbf{0.22 \pm 0.07}$

Table B.3: Average test Gaussian NLL of the observed features.

	Bank	Avocado
VAEM	0.13 ± 0.00	0.06 ± 0.00
MIWAEM	0.15 ± 0.00	0.09 ± 0.00
H-VAEM	0.11 ± 0.00	0.07 ± 0.00
HMC-VAEM	0.08 ± 0.00	0.05 ± 0.00
HH-VAEM	$\mathbf{0.07 \pm 0.00}$	$\mathbf{0.04 \pm 0.00}$

Table B.4: Average test Bernoulli NLL of the observed features.

	Bank	Avocado
VAEM	0.24 ± 0.16	0.33 ± 0.00
MIWAEM	0.26 ± 0.17	0.36 ± 0.00
H-VAEM	0.23 ± 0.16	0.32 ± 0.00
HMC-VAEM	0.22 ± 0.15	$\mathbf{0.30 \pm 0.00}$
HH-VAEM	$\mathbf{0.21 \pm 0.15}$	$\mathbf{0.30 \pm 0.00}$

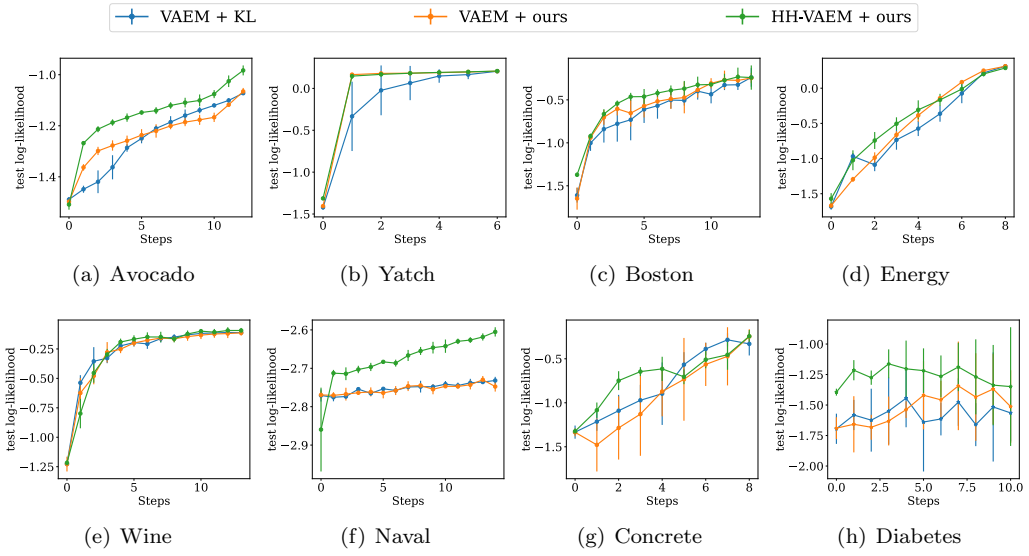
Table B.5: Average test Cat. NLL of the observed features.

B.1.6. SAIA log-likelihoods

In order to extend the results provided in Section 5.4.4, we include here the log-likelihoods curves when dynamically selecting features using the same procedure (Figure B.3).

B.1.7. Training times

Table B.6 shows the average training time in minutes for each model in the experiments for Tables 5.1 and 5.2. The ratio between training times for our method and the Gaussian baselines is approximately between 5 and 10.



(a) Avocado (b) Yatch (c) Boston (d) Energy

(e) Wine (f) Naval (g) Concrete (h) Diabetes

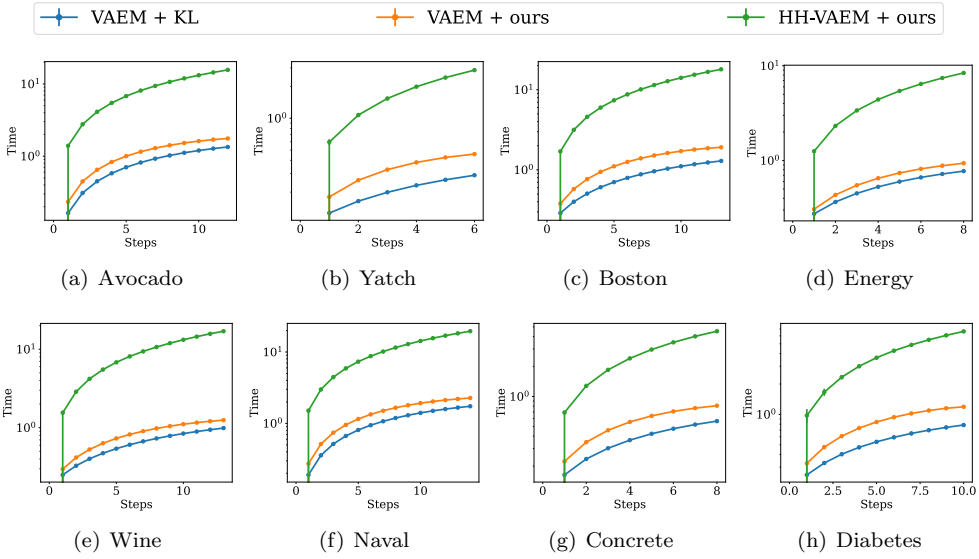
Figure B.3: SAIA log-likelihood curves. Horizontal axis shows acquisition steps (number of discovered features). Vertical axis is the log-likelihood of the target $\log p(\boldsymbol{y}|\boldsymbol{x}_O)$.

	Bank	Avocado	Yatch	Diabetes	Concrete	Wine	Energy	Boston
VAEM	29.92 ± 0.39	21.49 ± 1.64	5.89 ± 0.01	8.79 ± 0.30	7.70 ± 0.58	7.92 ± 0.11	8.18 ± 0.09	10.01 ± 0.34
MIWAEM	63.33 ± 6.20	37.17 ± 2.28	8.21 ± 0.24	13.29 ± 0.33	11.51 ± 0.52	11.81 ± 0.13	16.71 ± 0.16	15.01 ± 0.13
H-VAEM	41.44 ± 0.38	33.83 ± 0.81	15.84 ± 0.11	13.38 ± 0.47	11.80 ± 0.38	10.61 ± 0.54	12.71 ± 1.20	13.74 ± 1.43
HMC-VAEM	281.88 ± 9.14	356.22 ± 5.94	23.50 ± 1.60	50.42 ± 1.18	42.70 ± 3.14	63.46 ± 1.97	90.87 ± 7.05	103.72 ± 7.24
HH-VAEM	316.81 ± 9.49	388.28 ± 6.47	27.08 ± 0.12	68.29 ± 4.06	65.78 ± 0.43	79.97 ± 5.53	140.33 ± 7.36	129.30 ± 4.69

Table B.6: Training times (in minutes) of our model and baselines.

B.1.8. SAIA times

The times for obtaining the SAIA metric curves in Section 5.4.4 are included in Figure B.4. Although the performance is improved with HH-VAEM, it requires considerably longer time than the baselines to evaluate the reward, due to the HMC algorithm for sampling from the better approximated posterior. Future work might be oriented in proposing ways to measure and reduce this gap.



(a) Avocado (b) Yatch (c) Boston (d) Energy

(e) Wine (f) Naval (g) Concrete (h) Diabetes

Figure B.4: SAIA time curves. Horizontal axis shows acquisition steps (number of discovered features). Vertical axis is the elapsed time.

D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A Learning Algorithm for Boltzmann Machines. *Cognitive science*, 9(1):147–169, 1985.

J. F. H. Albarracin and A. R. Rivera. Video Reenactment as Inductive Bias for Content-Motion Disentanglement. *IEEE Transactions on Image Processing*, 31:2365–2374, 2022.

A. Alemi, B. Poole, I. Fischer, J. Dillon, R. A. Saurous, and K. Murphy. Fixing a Broken ELBO. In *International conference on machine learning*, pages 159–168. PMLR, 2018.

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov Chain Monte Carlo Methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

D. Antognini and B. Faltings. Fast Multi-Step Critiquing for VAE-based Recommender Systems. In *Proceedings of the 15th ACM Conference on Recommender Systems*, pages 209–219, 2021.

J. Antoran and A. Miguel. Disentangling and Learning Robust Representations with Natural Clustering. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 694–699. IEEE, 2019.

M. Aubry, D. Maturana, A. A. Efros, B. C. Russell, and J. Sivic. Seeing 3D chairs: Exemplar Part-Based 2D-3dD Alignment Using a Large Dataset of CAD Models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3762–3769, 2014.

J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. van den Berg. Structured Denoising Diffusion Models in Discrete State-Spaces. *Advances in Neural Information Processing Systems*, 34: 17981–17993, 2021.

D. Bahdanau, K. Cho, and Y. Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473*, 2014.

S. Barocas, M. Hardt, and A. Narayanan. Fairness in Machine Learning. *NIPS Tutorial*, 1, 2017.

D. Barrejón, P. M. Olmos, and A. Artés-Rodríguez. Medical data wrangling with sequential Variational Autoencoders. *arXiv preprint arXiv:2103.07206*, 2021.

Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Y. Bengio et al. Learning Deep Architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

R. v. d. Berg, L. Hasenclever, J. M. Tomczak, and M. Welling. Sylvester Normalizing Flows for Variational Inference. *arXiv preprint arXiv:1803.05649*, 2018.

J. M. Bernardo. Expected Information as Expected Utility. *the Annals of Statistics*, pages 686–690, 1979.

J. M. Bernardo and A. F. Smith. *Bayesian Theory*, volume 405. John Wiley & Sons, 2009.

M. Betancourt. A conceptual introduction to Hamiltonian Monte Carlo. *arXiv preprint arXiv:1701.02434*, 2017.

M. Betancourt and M. Girolami. Hamiltonian Monte Carlo for Hierarchical Models. *Current trends in Bayesian methodology with applications*, 79(30):2–4, 2015.

R. Bhagwatkar, K. Fitter, S. Bachu, A. Kulkarni, and S. Chiddarwar. Paying Attention to Video Generation. In *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*, pages 139–154. PMLR, 2021.

F. M. Bianchi, L. Livi, K. Ø. Mikalsen, M. Kampffmeyer, and R. Jenssen. Learning Representations of Multivariate Time Series with Missing Data. *Pattern Recognition*, 96:106973, 2019.

C. M. Bishop and N. M. Nasrabadi. *Pattern Recognition and Machine Learning*, volume 4. Springer, 2006.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Drichlet Allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

D. Bouchacourt, R. Tomioka, and S. Nowozin. Multi-Level Variational Autoencoder: Learning Disentangled Representations from Grouped Observations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio. Generating Sentences from a Continuous Space. *arXiv preprint arXiv:1511.06349*, 2015.

A. Brock, J. Donahue, and K. Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *International Conference on Learning Representations*, pages 1–18, 2018.

P. Bromiley. Products and Convolutions of Gaussian Probability Density Functions. *Tina-Vision Memo*, 3(4):1, 2003.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language Models Are Few-Shot Learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

B. A. Bucklin, N. L. Asdigian, J. L. Hawkins, and U. Klein. Making It Stick: Use of Active Learning Strategies in Continuing Medical Education. *BMC medical education*, 21:1–9, 2021.

S. Budd, E. C. Robinson, and B. Kainz. A Survey on Active Learning and Human-in-the-loop Deep Learning for Medical Image Analysis. *Medical Image Analysis*, 71:102062, 2021.

Y. Burda, R. Grosse, and R. Salakhutdinov. Importance Weighted Autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner. Understanding Disentangling in *beta*-VAE. *arXiv preprint arXiv:1804.03599*, 2018.

R. Cai, G. Yang, H. Averbuch-Elor, Z. Hao, S. Belongie, N. Snavely, and B. Hariharan. Learning Gradient Fields for Shape Generation. In *European Conference on Computer Vision*, pages 364–381. Springer, 2020.

A. Campbell, W. Chen, V. Stimper, J. M. Hernandez-Lobato, and Y. Zhang. A Gradient-Based Strategy for Hamiltonian Monte Carlo Hyperparameter optimization. In *International Conference on Machine Learning*, pages 1238–1248. PMLR, 2021.

A. L. Caterini, A. Doucet, and D. Sejdinovic. Hamiltonian Variational Auto-Encoder. *arXiv preprint arXiv:1805.11328*, 2018.

I. Cetin, M. Stephens, O. Camara, and M. A. G. Ballester. Attri-VAE: Attribute-Based Interpretable Representations of Medical Images with Variational Autoencoders. *Computerized Medical Imaging and Graphics*, 104:102158, 2023.

K. Chauhan, P. Shenoy, M. Gupta, D. Sridharan, et al. Robust Outlier Detection by De-Biasing VAE Likelihoods. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9881–9890, 2022.

N. Chen, Y. Zhang, H. Zen, R. J. Weiss, M. Norouzi, and W. Chan. WaveGrad: Estimating Gradients for Waveform Generation. *arXiv preprint arXiv:2009.00713*, 2020.

X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel. Variational Lossy Autoencoder. In *International Conference on Learning Representations*, 2017a. URL https://openreview.net/forum?id=BysvGP5ee.

Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng. Dual Path Networks. *Advances in neural information processing systems*, 30, 2017b.

R. Child. Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images. *arXiv preprint arXiv:2011.10650*, 2020.

J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv preprint arXiv:1412.3555*, 2014.

J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio. A Recurrent Latent Variable Model for Sequential Data. *Advances in neural information processing systems*, 28, 2015.

M. Collier, A. Nazabal, and C. K. Williams. VAEs in the Presence of Missing Data. *arXiv preprint arXiv:2006.05301*, 2020.

R. Couronné, P. Vernhet, and S. Durrleman. Longitudinal Self-Supervision to Disentangle Inter-Patient Variability from Disease Progression. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part II 24*, pages 231–241. Springer, 2021.

C. Cremer, X. Li, and D. Duvenaud. Inference Suboptimality in Variational Autoencoders. In *International Conference on Machine Learning*, pages 1078–1086. PMLR, 2018.

T. R. Davidson, L. Falorsi, N. De Cao, T. Kipf, and J. M. Tomczak. Hyperspherical Variational Auto-Encoders. *arXiv preprint arXiv:1804.00891*, 2018.

T. R. Davidson, J. M. Tomczak, and E. Gavves. Increasing Expressivity of a Hyperspherical VAE, 2019.

L. Deng, D. Lian, C. Wu, and E. Chen. Graph Convolution Network based Recommender Systems: Learning Guarantee and Item Mixture Powered Strategy. In *Advances in Neural Information Processing Systems*, volume 35, pages 3900–3912, 2022.

T. Denouden, R. Salay, K. Czarnecki, V. Abdelzad, B. Phan, and S. Vernekar. Improving Reconstruction Autoencoder Out-Of-Distribution Detection with Mahalanobis Distance. *arXiv preprint arXiv:1812.02765*, 2018.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*, 2018.

P. Dhariwal and A. Nichol. Diffusion Models Beat GANs on Image Synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever. Jukebox: A Generative Model for Music. *arXiv preprint arXiv:2005.00341*, 2020.

A. B. Dieng, Y. Kim, A. M. Rush, and D. M. Blei. Avoiding Latent Variable Collapse with Generative Skip Models. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2397–2405. PMLR, 2019.

N. Dilokthanakul, P. A. Mediano, M. Garnelo, M. C. Lee, H. Salimbeni, K. Arulkumaran, and M. Shanahan. Deep Unsupervised Clustering with Gaussian Mixture Variational Autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

L. Dinh, D. Krueger, and Y. Bengio. NICE: Non-linear Independent Components Estimation. *arXiv preprint arXiv:1410.8516*, 2014.

L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density Estimation Using Real NVP. *arXiv preprint arXiv:1605.08803*, 2016.

J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial Feature Learning. In *International Conference on Learning Representations*, 2017.

D. Dua, C. Graff, et al. UCI Machine Learning Repository. 2017. URL http://archive.ics.uci.edu/ml.

S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222, 1987.

R. O. Duda, P. E. Hart, et al. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.

D. Duvenaud, J. Kelly, K. Swersky, M. Hashemi, M. Norouzi, and W. Grathwohl. No MCMC for me: Amortized Samplers for Fast and Stable Training of Energy-Based Models. 2021.

C. Eastwood and C. K. Williams. A Framework for the Quantitative Evaluation of Disentangled Representations. In *International Conference on Learning Representations*, 2018.

S. Eduardo, A. Nazábal, C. K. Williams, and C. Sutton. Robust Variational Autoencoders for outlier detection and repair of mixed-type data. In *International Conference on Artificial Intelligence and Statistics*, pages 4056–4066. PMLR, 2020.

J. L. Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

B. Esmaeili, H. Wu, S. Jain, A. Bozkurt, N. Siddharth, B. Paige, D. H. Brooks, J. Dy, and J.-W. Meent. Structured Disentangled Representations. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2525–2534, 2019.

S. Fidler, S. Dickinson, and R. Urtasun. 3D Object Detection and Viewpoint Estimation with a Deformable 3D Cuboid Model. In *Advances in neural information processing systems*, pages 611–619, 2012.

M. Fraccaro, S. K. Sønderby, U. Paquet, and O. Winther. Sequential Neural Models with Stochastic Layers. *Advances in neural information processing systems*, 29, 2016.

M. Garnelo, D. Rosenbaum, C. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. W. Teh, D. Rezende, and S. A. Eslami. Conditional Neural Processes. In *International Conference on Machine Learning*, pages 1704–1713. PMLR, 2018.

I. Gatopoulos and J. M. Tomczak. Self-Supervised Variational Auto-Encoders. *Entropy*, 23(6): 747, 2021.

L. A. Gatys, A. S. Ecker, and M. Bethge. A nNeural Algorithm of Artistic Style. *arXiv preprint arXiv:1508.06576*, 2015.

Z. Ghahramani and M. I. Jordan. Learning from Incomplete Data. 1995.

E. Giné. The Lévy-Lindeberg central limit theorem in $l_p, 0 < p < 1$. *Proceedings of the American Mathematical Society*, 88(1):147–153, 1983.

M. Girolami and B. Calderhead. Riemann Manifold Langevin and Hamiltonian Monte Carlo Methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2): 123–214, 2011.

W. Gong, Y. Li, and J. M. Hernández-Lobato. Sliced Kernelized Stein Discrepancy. *arXiv preprint arXiv:2006.16531*, 2020.

Y. Gong, H. Hajimirsadeghi, J. He, T. Durand, and G. Mori. Variational Selective Autoencoder: Learning from Partially-Observed Heterogeneous Data. In *International Conference on Artificial Intelligence and Statistics*, pages 2377–2385. PMLR, 2021.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT press, 2016.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative Adversarial Networks. *Communications of the ACM*, 63(11):139–144, 2020.

J. Gordon and J. M. Hernández-Lobato. Bayesian Semisupervised Learning with Deep Generative Models. *arXiv preprint arXiv:1706.09751*, 2017.

J. Gordon and J. M. Hernández-Lobato. Combining Deep Generative and Discriminative Models for Bayesian Semi-Supervised Learning. *Pattern Recognition*, 100:107156, 2020.

J. W. Graham. *Missing Data: Analysis and Design*. Springer Science & Business Media, 2012.

A. Graves. Generating Sequences with Recurrent Neural Networks. *arXiv preprint arXiv:1308.0850*, 2013.

A. Graves and N. Jaitly. Towards End-to-End Speech Recognition with Recurrent Neural Networks. In *International conference on machine learning*, pages 1764–1772. PMLR, 2014.

A. Graves, A.-r. Mohamed, and G. Hinton. Speech Recognition with Deep Recurrent Neural Networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee, 2013.

U. Grenander and M. I. Miller. Representations of Knowledge in Complex Systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(4):549–581, 1994.

A. Guerrero-López, C. Sevilla-Salcedo, V. Gómez-Verdejo, and P. M. Olmos. Multi-View Hierarchical Variational AutoEncoders with Factor Analysis Latent Space. *arXiv preprint arXiv:2207.09185*, 2022.

P. Gyawali, Z. Li, C. Knight, S. Ghimire, B. M. Horacek, J. Sapp, and L. Wang. Improving Disentangled Representation Learning with the Beta Bernoulli Process. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 1078–1083. IEEE, 2019.

D. Ha, A. Dai, and Q. V. Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

J. He, A. Lehrmann, J. Marino, G. Mori, and L. Sigal. Probabilistic Video Generation using Holistic Attribute Control. In *Proceedings of the European conference on computer vision (ECCV)*, pages 452–467, 2018.

J. He, D. Spokoyny, G. Neubig, and T. Berg-Kirkpatrick. Lagging Inference Networks and Posterior Collapse in Variational Autoencoders. *arXiv preprint arXiv:1901.05534*, 2019.

K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016a.

K. He, X. Zhang, S. Ren, and J. Sun. Identity Mappings in Deep Residual Networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016b.

C. Heinze-Deml and N. Meinshausen. Conditional Variance Penalties and Domain Shift Robustness. *arXiv preprint arXiv:1710.11469*, 2017.

C. Heinze-Deml and N. Meinshausen. Conditional Variance Penalties and Domain Shift Robustness. *Machine Learning*, 110(2):303–348, 2021.

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. *Advances in neural information processing systems*, 30, 2017.

I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. 2016.

I. Higgins, D. Amos, D. Pfau, S. Racaniere, L. Matthey, D. Rezende, and A. Lerchner. Towards a Definition of Disentangled Representations. *arXiv preprint arXiv:1812.02230*, 2018.

G. Hinton, N. Srivastava, and K. Swersky. Neural Networks for Machine Learning Lecture 6a Overview of Mini-Batch Gradient Descent. *Cited on*, 14(8):2, 2012.

J. Ho, A. Jain, and P. Abbeel. Denoising Diffusion Probabilistic Models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

M. D. Hoffman. Learning deep latent Gaussian models with Markov Chain Monte Carlo. In *International conference on machine learning*, pages 1510–1519. PMLR, 2017.

E. Hoogeboom, V. Garcia Satorras, J. Tomczak, and M. Welling. The Convolution Exponential and Generalized Sylvester Flows. *Advances in Neural Information Processing Systems*, 33: 18249–18260, 2020.

E. Hoogeboom, D. Nielsen, P. Jaini, P. Forré, and M. Welling. Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 12454–12465. Curran Associates, Inc., 2021.

H. Hosoya. Group-Based Learning of Disentangled Representations with Generalizability for Novel Contents. In *IJCAI*, pages 2506–2513, 2019.

C.-W. Huang, J. H. Lim, and A. C. Courville. A Variational Perspective on Diffusion-based Generative Models and Score Matching. *Advances in Neural Information Processing Systems*, 34:22863–22876, 2021.

S.-J. Huang, M. Xu, M.-K. Xie, M. Sugiyama, G. Niu, and S. Chen. Active Feature Acquisition with Supervised Matrix Completion. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1571–1579, 2018.

A. Hyvärinen and P. Dayan. Estimation of Non-Normalized Statistical Models by Score Matching. *Journal of Machine Learning Research*, 6(4), 2005.

A. Hyvärinen and E. Oja. Independent Component Analysis: Algorithms and Applications. *Neural networks*, 13(4-5):411–430, 2000.

M. Ilse, J. M. Tomczak, C. Louizos, and M. Welling. DIVA: Domain Invariant Variational Autoencoders. In *Medical Imaging with Deep Learning*, pages 322–348. PMLR, 2020.

N. Ipsen, P.-A. Mattei, and J. Frellsen. How to Deal with Missing Data in Supervised Deep Learning? In *Artemiss-ICML Workshop on the Art of Learning with Missing Values*, 2020.

H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148(34):13, 2001.

A. Jalal, M. Arvinte, G. Daras, E. Price, A. G. Dimakis, and J. Tamir. Robust Compressed Sensing MRI with Deep Generative Priors. *Advances in Neural Information Processing Systems*, 34:14938–14954, 2021.

Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song. Neural Style Transfer: A Review. *IEEE transactions on visualization and computer graphics*, 26(11):3365–3385, 2019.

J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 694–711. Springer, 2016a.

M. J. Johnson, D. K. Duvenaud, A. Wiltschko, R. P. Adams, and S. R. Datta. Composing Graphical Models with Neural Networks for Structured Representations and Fast Inference. *Advances in neural information processing systems*, 29:2946–2954, 2016b.

W. Joo, W. Lee, S. Park, and I.-C. Moon. Dirichlet Variational Autoencoder. *Pattern Recognition*, 107:107514, 2020.

M. I. Jordan. *Attractor Dynamics and Parallelism in a Connectionist Sequential Machine*, page 112–127. IEEE Press, 1990. ISBN 0818620153.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An Introduction to Variational Methods for Graphical Models. *Machine learning*, 37:183–233, 1999.

T. Joy, S. Schmon, P. Torr, S. Narayanaswamy, and T. Rainforth. Capturing Label Characteristics in VAEs. In *Proceedings of the ICLR Conference 2021*. OpenReview, 2021.

J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, et al. Highly Accurate Protein Structure Prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

T. Karras, S. Laine, and T. Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

M. Kholghi, L. Sitbon, G. Zuccon, and A. Nguyen. Active Learning: A Step Towards Automating Medical Concept Extraction. *Journal of the American Medical Informatics Association*, 23 (2):289–296, 2016.

H. Kim and A. Mnih. Disentangling by Factorising. In *International Conference on Machine Learning*, pages 2649–2658. PMLR, 2018.

D. Kingma, T. Salimans, B. Poole, and J. Ho. Variational Diffusion Models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014.

D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

D. P. Kingma and M. Welling. An Introduction to Variational Autoencoders. *arXiv preprint arXiv:1906.02691*, 2019.

D. P. Kingma, S. Mohamed, D. Jimenez Rezende, and M. Welling. Semi-Supervised Learning with Deep Generative Models. *Advances in neural information processing systems*, 27, 2014.

D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved Variational Inference with Inverse Autoregressive Flow. *Advances in neural information processing systems*, 29:4743–4751, 2016.

A. Klushyn, N. Chen, R. Kurle, B. Cseke, and P. van der Smagt. Learning Hierarchical Priors in VAEs. *Advances in neural information processing systems*, 32, 2019.

Z. Kong and W. Ping. On Fast Sampling of Diffusion Probabilistic Models. *arXiv preprint arXiv:2106.00132*, 2021.

Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro. DiffWave: A Versatile Diffusion Model for Audio Synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

I. Korshunova, J. Degrave, F. Huszár, Y. Gal, A. Gretton, and J. Dambre. Bruno: A Deep Recurrent Model for Exchangeable Data. In *Advances in Neural Information Processing Systems*, pages 7190–7198, 2018.

B. Koyuncu, P. Sanchez-Martin, I. Peis, P. M. Olmos, and I. Valera. Variational Mixture of HyperGenerators for Learning Distributions Over Functions. *arXiv preprint arXiv:2302.06223*, 2023.

A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating Mutual Information. *Physical review E*, 69(6):066138, 2004.

J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3D Object Representations for Fine-Grained Categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet Classification with Deep Convolutional Neural Networks. *Communications of the ACM*, 60(6):84–90, 2017.

K. Lang. The development of the time-delay neural network architecture for speech recognition. *Technical Report CMU-CS-88-152*, 1988.

K. J. Lang, A. H. Waibel, and G. E. Hinton. A time-delay neural network architecture for isolated word recognition. *Neural networks*, 3(1):23–43, 1990.

Y. LeCun. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural computation*, 1(4): 541–551, 1989.

Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang. A Tutorial on Energy-Based Learning. *Predicting structured data*, 1(0), 2006.

Y. Li and S. Mandt. Disentangled Sequential Autoencoder. *arXiv preprint arXiv:1803.02991*, 2018.

Y. Li, J. Bradshaw, and Y. Sharma. Are Generative Classifiers More Robust to Adversarial Attacks? In *International Conference on Machine Learning*, pages 3804–3814. PMLR, 2019.

D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 2018 world wide web conference*, pages 689–698, 2018.

A. Liapounoff. Sur une proposition de la théorie des probabilités. *News of the Russian Academy of Sciences, Mathematical series*, 13(4):359–386, 1900.

D. V. Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, pages 986–1005, 1956.

R. J. Little and D. B. Rubin. *Statistical Analysis with Missing Data*, volume 793. John Wiley & Sons, 2019.

C. Liu, Z. Liao, Y. Ma, and K. Zhan. Stationary Diffusion State Neural Estimation for Multiview Clustering. *arXiv preprint arXiv:2112.01334*, 2021a.

Q. Liu, J. Lee, and M. Jordan. A Kernelized Stein Discrepancy for goodness-of-fit tests. In *International conference on machine learning*, pages 276–284. PMLR, 2016.

S. Liu, J. Liu, Q. Zhao, X. Cao, H. Li, D. Meng, H. Meng, and S. Liu. Discovering Influential Factors in Variational Autoencoders. *Pattern Recognition*, 100:107166, 2020.

X. Liu, S. Li, Y. Ge, P. Ye, J. You, and J. Lu. Recursively Conditional Gaussian for Ordinal Unsupervised Domain Adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 764–773, 2021b.

X. Liu, S. Thermos, A. O'Neil, and S. A. Tsaftaris. Semi-Supervised Meta-Learning with Disentanglement for Domain-Generalised Medical Image Segmentation. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part II 24*, pages 307–317. Springer, 2021c.

Z. Liu, P. Luo, X. Wang, and X. Tang. Deep Learning Face Attributes in the Wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

F. Locatello, G. Abbati, T. Rainforth, S. Bauer, B. Schölkopf, and O. Bachem. On the Fairness of Disentangled Representations. *Advances in neural information processing systems*, 32, 2019a.

F. Locatello, S. Bauer, M. Lucic, G. Raetsch, S. Gelly, B. Schölkopf, and O. Bachem. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. In *international conference on machine learning*, pages 4114–4124, 2019b.

Y.-J. Luo, K. W. Cheuk, T. Nakano, M. Goto, and D. Herremans. Unsupervised Disentanglement of Pitch and Timbre for Isolated Musical Instrument Sounds. In *ISMIR*, pages 700–707, 2020.

C. Ma, S. Tschiatschek, K. Palla, J. M. Hernández-Lobato, S. Nowozin, and C. Zhang. EdDI: Efficient Dynamic Discovery of High-Value Information with Partial VAE. *arXiv preprint arXiv:1809.11142*, 2018.

C. Ma, S. Tschiatschek, K. Palla, J. M. Hernandez-Lobato, S. Nowozin, and C. Zhang. EDDI: Efficient Dynamic Discovery of High-Value Information with Partial VAE. In *International Conference on Machine Learning*, pages 4234–4243. PMLR, 2019a.

C. Ma, S. Tschiatschek, J. M. Hernández-Lobato, R. Turner, and C. Zhang. VAEM: a Deep Generative Model for Heterogeneous Mixed Type Data. *arXiv preprint arXiv:2006.11941*, 2020.

J. Ma, C. Zhou, P. Cui, H. Yang, and W. Zhu. Learning Disentangled Representations for Recommendation. *Advances in neural information processing systems*, 32, 2019b.

L. Maaløe, M. Fraccaro, V. Liévin, and O. Winther. BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. *arXiv preprint arXiv:1902.02102*, 2019.

D. J. MacKay. Information-based Objective Functions for Active Data Selection. *Neural computation*, 4(4):590–604, 1992.

E. Mathieu, C. Le Lan, C. J. Maddison, R. Tomioka, and Y. W. Teh. Continuous Hierarchical Representations with Poincaré Variational Auto-Encoders. *Advances in neural information processing systems*, 32, 2019a.

E. Mathieu, T. Rainforth, N. Siddharth, and Y. W. Teh. Disentangling Disentanglement in Variational Autoencoders. In *International Conference on Machine Learning*, pages 4402–4412, 2019b.

P.-A. Mattei and J. Frellsen. MIWAE: Deep Generative Modelling and Imputation of Incomplete Data Sets. In *International Conference on Machine Learning*, pages 4413–4423. PMLR, 2019.

J. Melechovsky, A. Mehrish, D. Herremans, and B. Sisman. Learning Accent Representation with Multi-Level VAE Towards Controllable Speech Synthesis. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 928–935. IEEE, 2023.

P. Melville, M. Saar-Tsechansky, F. Provost, and R. Mooney. Active Feature-Value Acquisition for Classifier Induction. In *Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 483–486. IEEE, 2004.

N. Metropolis and S. Ulam. The Monte Carlo method. *Journal of the American statistical association*, 44(247):335–341, 1949.

G. Mittal, J. Engel, C. Hawthorne, and I. Simon. Symbolic Music Generation with Diffusion Models. *arXiv preprint arXiv:2103.16091*, 2021.

T. Miyato and M. Koyama. cGANs with Projection Discriminator. In *International Conference on Learning Representations*, 2018.

F. Moreno-Pino, P. M. Olmos, and A. Artés-Rodríguez. Deep Autoregressive Models with Spectral Attention. *Pattern Recognition*, 133:109014, 2023.

K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT press, 2012.

S. N, B. Paige, J.-W. van de Meent, A. Desmaison, N. Goodman, P. Kohli, F. Wood, and P. Torr. Learning Disentangled Representations with Semi-Supervised Deep Generative Models. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/9cb9ed4f35cf7c2f295cc2bc6f732a84-Paper.pdf.

V. Nair and G. E. Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

A. Nazabal, P. M. Olmos, Z. Ghahramani, and I. Valera. Handling Incomplete Heterogeneous Data Using VAEs. *Pattern Recognition*, page 107501, 2020.

R. M. Neal. Hamiltonian Importance Sampling. In *talk presented at the Banff International Research Station (BIRS) workshop on Mathematical Issues in Molecular Dynamics*, 2005.

R. M. Neal et al. MCMC using Hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

A. T. Nguyen, T. Tran, Y. Gal, and A. G. Baydin. Domain Invariant Representation Learning with Domain Density Transformations. *Advances in Neural Information Processing Systems*, 34:5264–5275, 2021.

A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. *arXiv preprint arXiv:1609.03499*, 2016.

OpenAI. GPT-4 Technical Report, 2023.

G. Papamakarios, E. T. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan. Normalizing Flows for Probabilistic Modeling and Inference. *J. Mach. Learn. Res.*, 22(57): 1–64, 2021.

G. Parisi. Correlation functions and computer simulations. *Nuclear Physics B*, 180(3):378–384, 1981.

P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter. A 3D Face Model for Pose and Illumination Invariant Face Recognition. In *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 296–301. Ieee, 2009.

I. Peis, P. M. Olmos, C. Vera-Varela, M. L. Barrigón, P. Courtet, E. Baca-Garcia, and A. Artes-Rodriguez. Deep Sequential Models for Suicidal Ideation from Multiple Source Data. *IEEE journal of biomedical and health informatics*, 23(6):2286–2293, 2019.

I. Peis, J.-D. López-Moríñigo, M. M. Pérez-Rodríguez, M.-L. Barrigón, M. Ruiz-Gómez, A. Artés-Rodríguez, and E. Baca-García. Actigraphic recording of motor activity in depressed inpatients: a novel computational approach to prediction of clinical course and hospital discharge. *Scientific reports*, 10(1):17286, 2020.

I. Peis, C. Ma, and J. M. Hernández-Lobato. Missing Data Imputation and Acquisition with Deep Hierarchical Models and Hamiltonian Monte Carlo. In *Advances in Neural Information Processing Systems*, volume 35, pages 35839–35851, 2022.

I. Peis, P. M. Olmos, and A. Artés-Rodríguez. Unsupervised learning of global factors in deep generative models. *Pattern Recognition*, 134:109130, 2023.

C. Pelletier, G. I. Webb, and F. Petitjean. Temporal Convolutional Neural Network for the Classification of Satellite Image Time Series. *Remote Sensing*, 11(5):523, 2019.

M. Phuong, M. Welling, N. Kushman, R. Tomioka, and S. Nowozin. The Mutual Autoencoder: Controlling Information in Latent Code Representations, 2018. In *URL https://openreview.net/forum*.

V. Popov, I. Vovk, V. Gogoryan, T. Sadekova, and M. Kudinov. Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech. In *International Conference on Machine Learning*, pages 8599–8608. PMLR, 2021.

A. Radford, L. Metz, and S. Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv preprint arXiv:1511.06434*, 2015.

A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving Language Understanding by Generative Pre-Training. 2018.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language Models Are Unsupervised Multitask Learners. *OpenAI blog*, 1(8):9, 2019.

R. Ranganath, D. Tran, and D. Blei. Hierarchical Variational Models. In *International Conference on Machine Learning*, pages 324–333, 2016.

A. Razavi, A. v. d. Oord, B. Poole, and O. Vinyals. Preventing posterior collapse with Delta-VAEs. *arXiv preprint arXiv:1901.03416*, 2019a.

A. Razavi, A. Van den Oord, and O. Vinyals. Generating Diverse High-Fidelity Images with VQ-VAE-2. *Advances in neural information processing systems*, 32, 2019b.

S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Advances in neural information processing systems*, 28, 2015.

D. Rezende and S. Mohamed. Variational Inference with Normalizing Flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015a.

D. Rezende and S. Mohamed. Variational Inference with Normalizing Flows. In *International Conference on Machine Learning*, pages 1530–1538. PMLR, 2015b.

D. J. Rezende and F. Viola. Taming VAEs. *arXiv preprint arXiv:1810.00597*, 2018.

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck. A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music. In *International conference on machine learning*, pages 4364–4373. PMLR, 2018.

O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

B. C. Ross. Mutual Information Between Discrete and Continuous Data Sets. *PloS one*, 9(2): e87357, 2014.

F. J. Ruiz, M. K. Titsias, T. Cemgil, and A. Doucet. Unbiased Gradient Estimation for Variational Auto-Encoders using Coupled Markov Chains. In *Uncertainty in Artificial Intelligence*, pages 707–717. PMLR, 2021.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning Internal Representations by Error Propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

M. Saar-Tsechansky, P. Melville, and F. Provost. Active Feature-Value Acquisition. *Management Science*, 55(4):664–684, 2009.

C. Saharia, J. Ho, W. Chan, T. Salimans, D. J. Fleet, and M. Norouzi. Image Super-resolution via Iterative Refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

R. Salakhutdinov. Learning Deep Generative Models. *Annual Review of Statistics and Its Application*, 2:361–385, 2015.

R. Salakhutdinov and G. Hinton. Deep Boltzmann machines. In *Artificial intelligence and statistics*, pages 448–455. PMLR, 2009.

T. Salimans, D. Kingma, and M. Welling. Markov Chain Monte Carlo and Variational Inference: Bridging the gap. In *International Conference on Machine Learning*, pages 1218–1226. PMLR, 2015.

T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma. PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications. *arXiv preprint arXiv:1701.05517*, 2017.

D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski. DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

J. Schmidhuber. Making the World Differentiable: On Using Fully Recurrent Self-supervised Neural Networks for Dynamic Reinforcement Learning and Planning in Non-Stationary Environments. *Institut für Informatik, Technische Universität München. Technical Report FKI-126*, 90, 1990.

J. Serrà, D. Álvarez, V. Gómez, O. Slizovskaia, J. F. Núñez, and J. Luque. Input Complexity and Out-Of-Distribution Detection with Likelihood-Based Generative Models. *arXiv preprint arXiv:1909.11480*, 2019.

I. Shenbin, A. Alekseev, E. Tutubalina, V. Malykh, and S. I. Nikolenko. RecVAE: A new Variational Autoencoder for Top-N Recommendations with Implicit Feedback. In *Proceedings of the 13th international conference on web search and data mining*, pages 528–536, 2020.

K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556*, 2014.

S. Sinha and A. B. Dieng. Consistency Regularization for Variational auto-Encoders. *Advances in Neural Information Processing Systems*, 34:12943–12954, 2021.

M. Śmieja, Ł. Struski, J. Tabor, B. Zieliński, and P. Spurek. Processing of Missing Data by Neural Networks. *Advances in neural information processing systems*, 31, 2018.

P. Smolensky. Information Processing in Dynamical Systems: Foundations of Harmony Theory. 1986.

C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. Ladder Variational Autoencoders. *Advances in neural information processing systems*, 29:3738–3746, 2016.

Y. Song and S. Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

Y. Song and S. Ermon. Improved Techniques for Training Score-Based Generative Models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

Y. Song and D. P. Kingma. How to Train Your Energy-Based Models. *arXiv preprint arXiv:2101.03288*, 2021.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-Based Generative Modeling through Stochastic Differential Equations. *arXiv preprint arXiv:2011.13456*, 2020.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*, 2021.

D. J. Stekhoven and P. Bühlmann. MissForest—Non-Parametric Missing Value Imputation for Mixed-Type Data. *Bioinformatics*, 28(1):112–118, 2012.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to Sequence Learning with Neural Networks. *Advances in neural information processing systems*, 27, 2014.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

B. Tang and D. S. Matteson. Probabilistic Transformer for Time Series Analysis. *Advances in Neural Information Processing Systems*, 34:23592–23608, 2021a.

B. Tang and D. S. Matteson. Probabilistic Transformer For Time Series Analysis. In *Advances in Neural Information Processing Systems*, volume 34, 2021b.

D. Tang, D. Liang, T. Jebara, and N. Ruozzi. Correlated Variational Auto-Encoders. In *International Conference on Machine Learning*, pages 6135–6144. PMLR, 2019.

M. Thahir, T. Sharma, and M. K. Ganapathiraju. An efficient heuristic method for active feature acquisition and its application to protein-protein interaction prediction. In *BMC proceedings*, volume 6, pages 1–9. BioMed Central, 2012.

A. Thin, N. Kotelevskii, A. Doucet, A. Durmus, E. Moulines, and M. Panov. Monte Carlo Variational Auto-Encoders. In *International Conference on Machine Learning*, pages 10247–10257. PMLR, 2021.

M. E. Tipping and C. M. Bishop. Probabilistic Principal Component Analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

J. Tomczak. Hierarchical VAEs, 2021. URL https://jmtomczak.github.io/blog/9/9_hierarchical_lvm_p1.html. Accessed: January 30, 2023.

J. Tomczak and M. Welling. VAE with a VampPrior. In *International Conference on Artificial Intelligence and Statistics*, pages 1214–1223. PMLR, 2018.

J. M. Tomczak. Learning Informative Features from Restricted Boltzmann Machines. *Neural Processing Letters*, 44:735–750, 2016.

J. M. Tomczak. *Deep Generative Modeling*. Springer, 2022.

J. M. Tomczak and M. Welling. Improving Variational Auto-Encoders Using Householder Flow. *arXiv preprint arXiv:1611.09630*, 2016.

V. Tresp, S. Ahmad, and R. Neuneier. Training Neural Networks with Deficient Data. *Advances in neural information processing systems*, 6, 1993.

B. Tzen and M. Raginsky. Neural Stochastic Differential Equations: Deep Latent Gaussian Models in the Diffusion Limit. *arXiv preprint arXiv:1905.09883*, 2019.

A. Vahdat and J. Kautz. NVAE: A Deep Hierarchical Variational Autoencoder. *arXiv preprint arXiv:2007.03898*, 2020.

I. Valera and Z. Ghahramani. Automatic Discovery of the Statistical Types of Variables in a Dataset. In *International Conference on Machine Learning*, pages 3521–3529. PMLR, 2017.

S. Van Buuren. *Flexible Imputation of Missing Data*. CRC press, 2018.

A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional Image Generation with PixelCNN Decoders. *Advances in neural information processing systems*, 29, 2016.

A. Van Den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel Recurrent Neural Networks. In *International conference on machine learning*, pages 1747–1756. PMLR, 2016.

A. Van Den Oord, O. Vinyals, et al. Neural Discrete Representation Learning. *Advances in neural information processing systems*, 30, 2017.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention Is All You Need. *Advances in neural information processing systems*, 30, 2017.

P. Vincent. A Connection between Score Matching and Denoising Autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

M. J. Vowels, N. C. Camgoz, and R. Bowden. NestedVAE: Isolating Common Factors via Weak Supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9202–9212, 2020.

M. J. Vowels, N. C. Camgoz, and R. Bowden. VDSM: Unsupervised Video Disentanglement with State-Space Modeling and Deep Mixtures of Experts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8176–8186, 2021.

A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang. Phoneme recognition using time-delay neural networks. *IEEE transactions on acoustics, speech, and signal processing*, 37 (3):328–339, 1989.

Y. Wang and J. P. Cunningham. Posterior Collapse and Latent Variable Non-identifiability. In *Third Symposium on Advances in Approximate Bayesian Inference*, 2020.

P. J. Werbos. Generalization of Backpropagation with Application to a Recurrent Gas Market Model. *Neural networks*, 1(4):339–356, 1988.

R. J. Williams and D. Zipser. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural computation*, 1(2):270–280, 1989.

C. Wolf, M. Karl, and P. van der Smagt. Variational Inference with Hamiltonian Monte Carlo. *arXiv preprint arXiv:1609.08203*, 2016.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Z. Xiao, Q. Yan, and Y. Amit. Likelihood Regret: An Out-Of-Distribution Detection Score for Variational Auto-Encoder. *Advances in neural information processing systems*, 33:20685–20696, 2020.

S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated Residual Transformations for Deep Neural Networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

K. Xu, A. Srivastava, and C. Sutton. Variational Russian Roulette for Deep Bayesian Nonparametrics. In *International Conference on Machine Learning*, pages 6963–6972, 2019.

W. Yan, Y. Zhang, P. Abbeel, and A. Srinivas. VideoGPT: Video Generation using VQ-VAE and Transformers. *arXiv preprint arXiv:2104.10157*, 2021.

H. H. Yang and S.-i. Amari. Adaptive Online Learning Algorithms for Blind Separation: Maximum Entropy and Minimum Mutual Information. *Neural computation*, 9(7):1457–1482, 1997.

R. Yang, D. Wang, Z. Wang, T. Chen, J. Jiang, and G. Xia. Deep Music Analogy via Latent Representation Disentanglement. *arXiv preprint arXiv:1906.03626*, 2019.

Y. Yang, Z. Xue, and A. Whinston. Self-Enhancing Multi-filter Sequence-to-Sequence Model. *Procedia Computer Science*, 215:537–545, 2022.

X. Zeng, A. Vahdat, F. Williams, Z. Gojcic, O. Litany, S. Fidler, and K. Kreis. LION: Latent Point Diffusion Models for 3D Shape Generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

C. Zhang, J. Bütepage, H. Kjellström, and S. Mandt. Advances in Variational Inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

B. Zhao, H. Lu, S. Chen, J. Liu, and D. Wu. Convolutional neural networks for time series classification. *Journal of Systems Engineering and Electronics*, 28(1):162–169, 2017.