UNIVERSITAT DE BARCELONA

FUNDAMENTAL PRINCIPLES OF DATA SCIENCE MASTER'S THESIS

---

# Motion Binary Latent Diffusion

---

*Author:*
Àlex PUJOL

*Supervisor:*
Dr. Sergio ESCALERA
Germán BARQUERO
Cristina PALMERO

*A thesis submitted in partial fulfillment of the requirements*
*for the degree of MSc in Fundamental Principles of Data Science*

*in the*

Facultat de Matemàtiques i Informàtica

January 17, 2024

UNIVERSITAT DE BARCELONA

# *Abstract*

Facultat de Matemàtiques i Informàtica

MSc

**Motion Binary Latent Diffusion**

by Àlex PUJOL

Human motion, a complex phenomenon studied for decades, has witnessed a significant surge in research with the advent of deep learning. Applications span from human-computer interaction to robotics, emphasizing the need for intelligent agents capable of natural interaction with humans. A crucial aspect in this development is Human Motion Generation from a text prompt, enabling users to synthesize motion sequences without technical expertise. While recent approaches leverage deep generative models like Diffusion Models and Vector Quantized Variational Autoencoders, challenges persist, such as resource overconsumption and the assumption that human motion intricacies can be represented by a limited set of codebook vectors. This thesis explores the foundations of diffusion models and variational autoencoders, examines the state-of-the-art in text-to-motion generation, and introduces a novel approach to motion representation and generation through binary latent spaces. The Motion Binary Variational Autoencoder is proposed, learning bidirectional mappings between motion sequences and binary tensors. Additionally, the Motion Binary Latent Diffusion model extends the concept of Bernoulli diffusion to the binary latent space of motion data. The thesis concludes with experimental evaluations of the proposed models, offering insights into their performance and suggestions for future research.

# *Acknowledgements*

I extend my sincere gratitude to my thesis advisor, Sergio Escalera, for their invaluable guidance and unwavering support throughout the research process. I also thank German Barquero and Cristina Palmero for their insightful comments, and fruitful advices. Appreciation goes to Leonardo, David, Paula and Patricia for enriching discussions and support, and to my friends and family for their continuous encouragement. This thesis reflects not only my effort but also the collaborative contributions of a supportive academic community, for which I am truly thankful.

# Contents

# 1 Introduction and Motivation

## 1.1 Human Motion Generation

As humans, our capacity for natural interaction with the surrounding world encompasses intricate, coordinated, and precise bodily movements, as well as communication through speech, gestures, and subtle body language. Despite its apparent simplicity, replicating such behavior in machines is a highly intricate challenge (Barquero et al., 2022), constituting an active and multidisciplinary research field spanning computer vision (Guo et al., 2022, Kim et al., 2022), computer graphics (Alexanderson et al., 2023, Ao, Zhang, and Liu, 2023), multimedia (Gao et al., 2022, Yoon et al., 2022), robotics (Nishimura, Nakamura, and Ishiguro, 2020, Gulletta, Erlhagen, and Bicho, 2020), and human-machine interaction (Kucherenko et al., 2019, Yin et al., 2022). The resolution of the task of human motion synthesis holds the potential to facilitate the creation of more natural, intuitive, and reliable movements in machines. This breakthrough could significantly enhance the interaction between humans and computers, with far-reaching implications across a diverse array of applications, particularly in the domains of robotics, virtual and augmented reality, animation, video games, autonomous driving, healthcare, and possibly introducing novel solutions (Zhu et al., 2023).

The task of human motion synthesis can be approached from various perspectives, tailored to the specific nature of the problem at hand. Motion forecasting, for example, involves utilizing past motion data to predict future steps in a movement (Barquero, Escalera, and Palmero, 2023, Lucas et al., 2022). Models that translate between text to motion, and vice versa, estimate the mappings between natural language and human movement. These models can synthesize motion based on a given text prompt (Chen et al., 2023a, Zhang et al., 2023b) or provide a textual description and "discuss" about a given movement (Jiang et al., 2023). Similarly, action to motion models label movement sequences with a corresponding action category, which is usually a one-hot vector representing a verb like "walking" or "running" (Lucas et al., 2022). Other approaches include utilizing music audio files to generate a dancing human figure (Qi et al., 2023b), or motion editing, which employs control-to-motion models to adjust movement based on a provided control signal, such as a pose or fixed joint (Zhang et al., 2021). Human Motion Generation (HMG) encapsulates seemingly disparate tasks, such as 3D-avatar reconstruction, aiming to rebuild a human mesh with or without texture from a given video of a person (Zhang et al., 2023c). Also, it includes motion retargeting, which transfers the motion of one person to another being (Tu et al., 2023), hand and facial motion synthesis (Kirschstein, Giebenhain, and Nießner, 2023), and behavior analysis (Palmero et al., 2022). Each of these facets contributes to the comprehensive exploration of human motion synthesis from diverse angles.

### 1.1.1 Motion Data and Motion Representation

An effective approach for representing motion data involves organizing a sequence of human body poses along a temporal dimension. However, poses can be represented using various methods. Two conventional approaches are through *keypoints* and *rotations*. Theoretically, both methods are considered equivalent, as it is feasible to transition between them and vice versa using forward and inverse kinematics. On one side, inverse kinematics

is a mathematical process which calculates the variable joint parameters needed to place the end of a kinematic chain, such as an animation character skeleton to a given position and orientation based relative to the start of the chain. On the other side, forward kinematics calculates the character skeleton based on given joint parameters.

A *keypoint*-based representation employs designated points corresponding to anatomical parts within a model, including joints, face, and other noteworthy locations. Consequently, a pose is characterized by an array of triads (or tuples) representing the 3D (or 2D) spatial coordinates for each keypoint. Motion data, in this context, constitutes a sequential arrangement of these poses over time. Despite this natural representation of human shapes demonstrates significant interpretability and can be directly derived from motion capture systems, they lack of applicability in some cases. Usually for animation or robotics, keypoint-based representation need to be transformed into rotations via the IK problem (Zhang, Black, and Tang, 2021).

Human pose can also be represented by joint angles, that is, the *rotation* of the body parts or segments, relative to their parents in a hierarchical structure. These rotations can be parameterized using a variety of formats, like axis angles and quaternions. Skinned Multi-Person Linear (SMPL) model is a widely used method to estimate human-shaped meshes from such rotations (Loper et al., 2015). Other models such as SMPL-X extend this to a more comprehensive model where body, face and hand are represented jointly (Pavlakos et al., 2019a). The SMPL and SMPL-x models are defined by a set of pose and shape parameters. In these models, each joint is characterized by the parameters of the relative 3D rotations relative to a standard skeletal kinematic tree.

Motion data collection adopts distinct approaches tailored to diverse scenarios. One prevalent method involves the use of markers, strategically placed on a subject and tracked within a controlled environment. Conversely, markerless techniques rely solely on synchronized cameras and computer vision algorithms. While offering versatility, these methods may sacrifice some precision compared to marker-based alternatives (Ye et al., 2022). Pseudo-labelling represents an approach suited for in-the-wild captures, utilizing human pose estimators algorithms to annotate keypoints or fitting body models based on available image evidence, albeit often resulting in less accurate motion representation (Pavlakos et al., 2019b, Cao et al., 2017). A contrasting methodology entails manual annotation, where a team of skilled artists meticulously crafts human motion using animation engines. While capable of producing high-quality movements, this method is resource-intensive, time-consuming, and lacks scalability.

## 1.2 Generative Models

Assume we want to generate new samples similar to a set $\mathcal{D} \subset \mathbb{R}^D$, of a particular data domain, where $D$ is the dimensionality of the specific data. For instance, the domain of RGB-colored images of size $w \times h$ has dimension $D = w \times h \times 3$. Then, for any element in this set, $\mathbf{x} \in \mathcal{D}$, a **generative model** addressing this problem would be designed to learn the underlyining data distribution $p_{\text{data}}(\mathbf{x})$ within the domain, and then be used to generate unseen samples.

Recent advancements in deep learning generative techniques, including Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), Diffusion Models (DMs) (Ho, Jain, and Abbeel, 2020, Song and Ermon, 2020, Karras et al., 2022), and attention-based models like Large Language Models (Vaswani et al., 2023, Guo et al., 2023), have provided versatile solutions for estimating the data distribution $p_{\text{data}}(\mathbf{x})$ from various perspectives. Over the past months, these models have undergone significant improvements, leading to a wide array of

applications (Yang et al., 2023). Notably, DMs have shown remarkable success in image generation (Podell et al., 2023) and across diverse modalities such as video (Yang, Srivastava, and Mandt, 2022), audio (Zhang et al., 2023a), and text (Han, Kumar, and Tsvetkov, 2023, Gong et al., 2023, Yuan et al., 2023). The emergence of Large Multimodal Models (Li, 2023) has further enhanced capabilities by leveraging diffusion models and attention mechanisms to generate comprehensive images based on a given text prompt and to "discuss" over them (Liu et al., 2023).

This recent evolution in generative models addresses the multimodality problem, involving the generation of samples from different data modalities (e.g., images, text, and audio) within a single model. Multimodal models typically aim to estimate partial conditioned distributions $q_{\text{data}}(\mathbf{x}|\mathbf{c})$, where $\mathbf{c}$ represents the conditioning data. When $\mathbf{x}$ and $\mathbf{c}$ pertain to different data modalities, the model deals with multimodality. This intriguing problem opens avenues for addressing more complex tasks, such as text-to-image (Liu et al., 2023, Zhang, Rao, and Agrawala, 2023), text-to-3D (Xu et al., 2023), or conditioned video generation, where the goal is to generate temporally coherent images, often accompanied by audio, in response to a given text prompt (Ho et al., 2022, Qi et al., 2023a).

The synthesis of human motion, which is the main focus of this thesis, is another such complex task. In particular, we explore the generation of human motion based on a given text prompt, see figure 1.1. Developing models for synthesizing motion from natural language empowers users without expertise in motion capture and animation, democratizing access to otherwise expensive technologies. However, conditioning the synthesis on text introduces additional challenges to the already complex field of HMG.

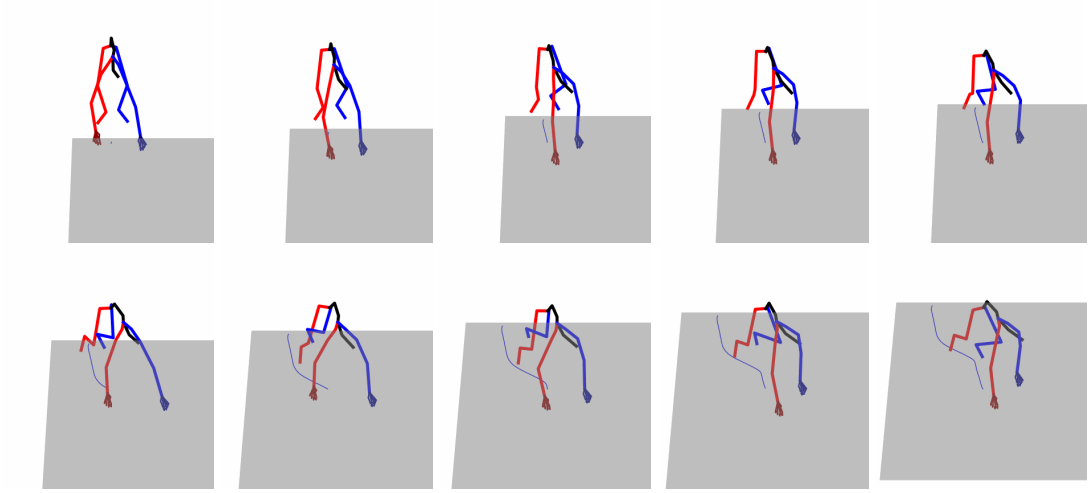

FIGURE 1.1: Extract of human motion with description "on hands and feet a person crawls four paces on an angle to the left, turns and crawls back, and then stands".

## 1.3 Challenges for Motion Generation

While one of the uses of HMG aims to be able to generate high-quality human motion synthetic data, it inherits the drawbacks of motion capture systems from the datasets. Hence, one of the major flaws is the lack of high-quality and diverse available datasets, since the predominant data collection techniques already mentioned are either expensive, not diverse or not accurate enough. A widely referenced dataset is AMASS, which is an amalgamation of different motion captured datasets, recorded using a variety of techniques, and standarised to a common motion representation based on SMPL (Mahmood et al., 2019). However, this

dataset is still limited in the number of subjects and the variety of motions with only 300 subjects and 40 hours of motion capture data. Other popular datasets are even much smaller. For instance, Human3.6M displays only the actions of 11 subjects (Ionescu et al., 2014). The lack of high-quality and diverse datasets is a major drawback for the development of HMG models, since it is difficult to train a model to generate a wide variety of motions without a large and diverse dataset.

Furthermore, synthesizing human motion from scratch extends beyond the mere application of deep generative models to human motion datasets. Human motion is highly non-linear and articulated, and subject to a wide variety of constraints, such as the laws of physics, the human anatomy, and purpose of the movement. Not only does the model need to generate temporally coherent human poses, it also needs to generate a smooth and plausible motion that is consistent with those constraints and the signal controls provided.

In the case of text-to-motion generation, the signal controls provided are the text prompts. They are usually short sentences describing the movement, such as "a person walking forward", and pose extra challenges. The model also needs to understand the semantics of the text prompt in order to generate the according motion. Moreover, labelling the data is problematical as the same movement can be described in several different ways, for instance, a motion showing a person "kicking something with the right leg" can be also described as "hitting something with the right foot". Another challenge is the lack of a clear and well-defined evaluation metric, which makes it difficult to compare the performance of different models, specially when trying to asses how well the model has understood the semantics of the text prompt. Additionally, a single sentence can be used to describe semantically different behaviors, that is "kicking something with the right leg" can refer to a soccer player kicking a ball, or a person kicking a door, and both motions are very different while sharing the same underlying pattern. Hence, a text conditioned HMG model should be able to deal with this many-to-many problem and generate a wide diversity of motions given a single text prompt, in order to be useful in the aforementioned applications.

In this thesis we explore a novel method to tackle the aforementioned semantical problem. We introduce a Binary Latent Diffusion Model (BLD) (Wang et al., 2023), that leverages the power of diffusion models and binary latent representations to generate high-quality samples from a compact yet expressive latent space. The thesis is divided in five chapters.

Within the first part we introduce the diffusion models and how they work, providing detailed and comprehensive explanations of the main concepts of Denoising Diffusion Probabilistic Models (DDPM) (Ho, Jain, and Abbeel, 2020), Latent Diffusion Models (LDM) (Rombach et al., 2021) and Vector Quantized Variational Autoencoders (VQ-VAE) (Oord, Vinyals, and Kavukcuoglu, 2018). We will also introduce the main concepts of the BLD model, which reinvents the LDM model to work with binary latents and is the main model used in this thesis. In the next chapter, we further discuss the topic of text-to-motion generation, the main challenges of this task, reviewing the state of the art and introducing the main contributions of this thesis. The following part, provides a detailed explanation of our proposed methodology and how the BLD model approaches the problem of motion generation, how it is implemented and a comparison with a classical Gaussian-based LDM model. The fourth chapter defines the datasets used for the experiments, the evaluation metrics, and details the expertimental setup. Then, it showcases the results obtained by the tested iterations of our proposed motion BLD model, and the binary autoencoders. Finally, within the concluding chapter, we summarize the main contributions of this thesis, and extract insights from the experimental outcomes. We also discuss the limitations of our proposed method and provide some ideas for future work.

# 2 Diffusion Models

## 2.1 Fundation of Diffusion Models

Diffusion models, a category of probabilistic generative models, were initially introduced by Sohl-Dickstein et al., 2015, and later gained prominence through the work of Ho, Jain, and Abbeel, 2020, demonstrating notable achievements in the domain of image generation. The foundational concept of diffusion models can be traced back to the principles of statistical physics and non-equilibrium thermodynamics (Jarzynski, 1997, Neal, 1998), where a Markov chain progressively transforms one probability distribution into another.

Since their introduction, academical and industrial communities have made a huge improvement in the field showcasing outstanding results in a diversity of applications. In this chapter we will focus on the foundations of such models, review in detail the diffusion algorithm, the latent diffusion models and different ways to represent data in a latent space, and finally introduce the Binary Latent Diffusion model, which is the algorithm of interest in this thesis.

### 2.1.1 Diffusion Process

The goal of the algorithm is to define a forward diffusion process that begins with a complex distribution $q_{\text{data}}(\mathbf{x})$ and ends with a simple and tractable distribution $\pi(\mathbf{y})$, which is usually a Gaussian or a Binomial. Then, a model $f_\theta(\mathbf{x})$ is trained to learn the reverse process bringing the simple distribution to an approximation of the complex one, where $\theta$ are the parameters to be tunned. To this end, the diffusion process, will be divided in two steps.

- **Forward Trajectory:** Consider the data domain $\mathcal{D}$ whose elements $\mathbf{x} \in \mathcal{D}$ can be any data type. Consider also the data distribution of the elements in this domain and denote it $q_{\text{data}}(\mathbf{x})$. From here we define a *forward trajectory* to be a Markov chain that perturbs $q_{\text{data}}(\mathbf{x})$ by repeatedly applying a Markov diffusion kernel $T_\pi(\mathbf{x}|\mathbf{x}';\beta)$, where $\pi(\mathbf{y})$ is a tractable distribution and $\beta$ is a diffusion step parameter. Hence, if we start with $q(\mathbf{x}^{(0)}) = q_{\text{data}}(\mathbf{x})$, the probability of the next step within the Markov chain will require the multiplication $q(\mathbf{x}^{(0,1)}) = q(\mathbf{x}^{(0)})T_\pi(\mathbf{x}^{(1)}|\mathbf{x}^{(0)};\beta)$. Then, the forward trajectory after $T$ steps of diffusion is given by

$$q(\mathbf{x}^{(0\ldots T)}) := q(\mathbf{x}^{(0)}) \prod_{t=1}^{T} q(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)})$$

  where $q(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}) = T_\pi(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)};\beta)$. Notice that this procedure allows us to transform any-to-any distribution, depending on the choice of the kernel $T$. However, since the aim of this procedure is to get to a tractable distribution $\pi(\mathbf{y})$, for the Gaussian we will choose the kernel to be

$$q(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}) = \mathcal{N}(\mathbf{x}^{(t)}; \mathbf{x}^{(t-1)}\sqrt{1-\beta_t}, \mathbf{I}\beta_t), \tag{2.1}$$

and for the binomial we will choose the kernel to be

$$q(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}) = \mathcal{B}(\mathbf{x}^{(t)}; \mathbf{x}^{(t-1)}(1-\beta_t) + 0.5\beta_t). \tag{2.2}$$

The choice of $\beta_t$'s is crucial for the success of the model. They can be learned by fixing $\beta_1$ or they can be defined to follow a schedule as a function of $t$. If the target function is tractable $q(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)})$ can be computed in a closed form given $\mathbf{x}^{(0)}$, for instance, in the Gaussian case we have

$$q(\mathbf{x}^{(t)}|\mathbf{x}^{(0)}) = \mathcal{N}(\mathbf{x}^{(t)}; \mathbf{x}^{(t-1)}\sqrt{\bar{\alpha}_t}, (1-\bar{\alpha}_t)\mathbf{I}) \tag{2.3}$$

with $\bar{\alpha}_t = \prod_{i=1}^{t}(1-\beta_i)$.

- **Reverse Trajectory:** Given the above procedure, if we were able to find a *reverse trajectory* $q(\mathbf{x}^{(T\ldots0)})$ we would be able to define a Markov chain that transforms $\pi(\mathbf{y})$ into the generative model we are looking for. However, computing $q(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)})$ is intractable, so we need to find an approximation. Therefore, we will train a model $p_\theta(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)})$ to learn the reverse process:

$$\begin{aligned} p(\mathbf{x}^{(T)}) &= \pi(\mathbf{x}^{(T)}) \\ p(\mathbf{x}^{(0\ldots T)}) &= p(\mathbf{x}^{(T)}) \prod_{t=1}^{T} q(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}) \\ &= p(\mathbf{x}^{(T)}) \prod_{t=1}^{T} p_\theta(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}). \end{aligned}$$

Given small enough diffusion steps, (Sohl-Dickstein et al., 2015) guarantees that if the forward diffusion kernel is a Gaussian or a Binomial then the reverse diffusion kernel will also be Gaussian or Binomial respectively. Therefore, our model will be trained to learn the mean and covariance matrix,

$$p_\theta(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}) = \mathcal{N}(\mathbf{x}^{(t-1)}; \mu_\theta(\mathbf{x}^{(t)}, t), \Sigma_\theta(\mathbf{x}^{(t)}, t)), \tag{2.4}$$

or the bit flip probability,

$$p_\theta(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}) = \mathcal{B}(\mathbf{x}^(t-1); \mathbf{b}_\theta(\mathbf{x}^{(t)}, t)).$$

At the end of the process, the generative model approximates the data distribution by

$$\begin{aligned} p_\theta(\mathbf{x}^{(0)}) &= \int p_\theta(\mathbf{x}^{(0\ldots T)}) d\mathbf{x}^{(1\ldots T)} = \\ &= \int q(\mathbf{x}^{(1\ldots T)}|\mathbf{x}^{(0)}) p(\mathbf{x}^{(T)}) \prod_{t=1}^{T} \frac{p(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)})}{q(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)})} d\mathbf{x}^{(1\ldots T)}. \end{aligned}$$

### 2.1.2 Training Objective

Observe that if we treat $\mathbf{x}^{(0)}$ as an observed variable, and $\mathbf{x}^{(1\ldots T)}$ as latent variables, then the diffusion process can be seen as a Variational Autoencoder (VAE) (Bank, Koenigstein, and Giryes, 2021). The forward and reverse trajectories would be equivalent to the encoder and decoder respectively, going from data to latent space and from latent back to the data.

Training is done by minimizing the negative log-likelihood of the model's data distribution,

$$L := -\mathbb{E}[\log p_\theta(\mathbf{x}^{(0)})] = -\int d\mathbf{x}^{(0)} q(\mathbf{x}^{(0)}) \log p_\theta(\mathbf{x}^{(0)}).$$

However, this is intractable, so the usual approach is to minimize instead the variational lower bound inspired from VAE's (Kingma and Welling, 2022, Sohl-Dickstein et al., 2015), which is given by

$$L \leq L_{\text{vlb}} := L_0 + \sum_{t=1}^{T-1} L_t + L_T, \tag{2.5}$$

where

$$\begin{aligned} L_0 &:= -\log p_\theta(\mathbf{x}^{(0)}|\mathbf{x}^{(1)}) \\ L_t &:= D_{\text{KL}}(q(\mathbf{x}^{(t)}|\mathbf{x}^{(t+1)}, {}^{(0)})||p_\theta(\mathbf{x}^{(t)}|\mathbf{x}^{(t+1)})) \\ L_T &:= D_{\text{KL}}(q(\mathbf{x}^{(T)}|\mathbf{x}^{(0)})||p(\mathbf{x}^{(T)})). \end{aligned}$$

Note that $q(\mathbf{x}^{(t)}|\mathbf{x}^{(t+1)}, {}^{(0)})$ can be computed using the Baye's rule, which implies that $L_t$'s can be computed in a closed form since we are comparing two Gaussians (or Binomials). Also, observe $L_T$ is constant, so we can ignore it during training. Thus, the task of estimating the data distribution has been reduced to finding the mean and covariance matrix of a sequence of Gaussians (or the bit flip probability of a sequence of Binomials) that minimize $L_{\text{vlb}}$.

### 2.1.3 Denoising Diffusion Probabilistic Models

Since the introduction of diffusion models, several improvements have been made to simplify the diffusion process and improve the performance of the model. In particular, it is worth mentioning the work of Ho, Jain, and Abbeel, 2020 who introduced the Denoising Diffusion Probabilistic Models (DDPM). They focused on working with Gaussian distributions and posed empirical evidence for reparameterizing the model and learning a much simpler version of the $L_{\text{vlb}}$. The idea is reducing the amount of learnable variables to be the noise between steps.

During the reverse process 2.4, the covariance matrix is predefined as a diagonal matrix $\sigma_\theta(\mathbf{x}^{(t)}, t) = \sigma_t^2 \mathbf{I}$. Recall that from the Baye's rule we can compute

$$q(\mathbf{x}^{(t)}|\mathbf{x}^{(t+1)}, \mathbf{x}^{(0)}) = \mathcal{N}(\mathbf{x}^{(t)}; \tilde{\mu}_\theta(\mathbf{x}^{(t)}, t), \tilde{\Sigma}_\theta(\mathbf{x}^{(t)}, t)).$$

Then, to represent the mean $\mu_\theta(\mathbf{x}^{(t)}, t)$ they reparameterize 2.3 as

$$\mathbf{x}^{(t)}(\mathbf{x}^{(0)}, \epsilon) = \sqrt{\bar{\alpha}_t}\mathbf{x}^{(0)} + \sqrt{1-\bar{\alpha}_t}\epsilon, \text{ for } \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}).$$

This leads to

$$\mu_\theta(\mathbf{x}^{(t)}, t) = \tilde{\mu}_t\left(\mathbf{x}^{(t)}, \frac{1}{\sqrt{\bar{\alpha}_t}}(\mathbf{x}^{(t)} - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(\mathbf{x}^{(t)}))\right) = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}^{(t)} - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}^{(t)})\right).$$

Therefore, it is enough to learn the denoising function $\epsilon_\theta$ between steps, with a simpler $L_{\text{vlb}}$ given by

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t, \mathbf{x}^{(0)}, \epsilon}\left[||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}^{(0)} + \sqrt{1-\bar{\alpha}_t}\epsilon, t)||_2^2\right]. \tag{2.6}$$

This new formulation of the training objective can be shown to be equivalent to the loss weighting used NCSN denoising score matching model (Song and Ermon, 2020), which is

a score-based generative modeling method that estimates the gradients of the data distributions at different levels of added noise. Therefore DDPMS can be seen as a connection between score-based and diffusion-based generative models. Their proposal also include fixing the $\beta_t$'s to a given schedule.

### 2.1.4 Conditioning and Guidance Techniques

The wide flexiblity of diffusion models allows them to be used in several different scenarios and adopt different properties. One of which is conditional generation. It is possible to condition the model to generate samples given a particular piece of information. For instance, text-to-image generation using diffusion models has been shown to be even more powerful than other methods such as GANs (Dhariwal and Nichol, 2021). There are different ways of approaching this can be summarized into three techniques:

- **Explicit Conditioning:** Conditinal sampling can be considered as training a model to learn the conditional distribution $p_\theta(\mathbf{x}^{(0)}|\mathbf{c})$. Here $\mathbf{c}$ is the conditioning information, which can be a text prompt, an image, or any other source of information. This approach is the most straightforward one, usually implemented by concatenating the conditioning information to the input of the model or by using an attention mechanism.

- **Classifier Guidance:** Another approach is to train a classifier to predict the conditioning information from the generated samples. Baye's rule allows us to use the gradient of the conditional distribution $p_\theta(\mathbf{c}|\mathbf{x}^{(t)})$, which comes from the classifier, to guide the model towards the desired output $p_\theta(\mathbf{x}^{(0)}|\mathbf{c})$.

- **Classifier-free Guidance:** By using the Baye's rule again we can get an implicit classifier by jointly training a conditional and an unconditional diffusion model (Nichol et al., 2022). In practice, both models are trained together by randomly sampling the condition of the diffusion model at a certain chance. Hence, we want to compute

  $$(1+\omega)\log p_\theta(\mathbf{x}_t|\mathbf{c}) - \omega \log p_\theta(\mathbf{x}_t).$$

  The difference to explicit conditioning is that this approach additionally accentuates the distance between the conditional and unconditional distributions, by means of the implicit classifier.

There are also other techniques that can be used to improve the performance of the diffusion models, some of which involve data augmentation techniques (Dhariwal and Nichol, 2021), or changing the schedule of the diffusion steps (Nichol and Dhariwal, 2021). However, when working with high resolution images, or large amounts of data, managing DDPMs can be computationally expensive. Therefore, in the next section we will briefly recall the definition of VAE, which will be used to reduce the dimensionality of the data in order to make the diffusion process more efficient. In particular we will focus this introduction on Vector Quantized Variational Autoencoders (VQ-VAE) (Oord, Vinyals, and Kavukcuoglu, 2018).

## 2.2 Latent Representations

Learning useful representations of the data without supervision is a key challenge in machine learning. The first models that provided a non-supervised solution for an efficent data representation are the so called Autoencoders (Bank, Koenigstein, and Giryes, 2021).

They consist of and Encoder $E_{\theta'}$, which is the first part of the model and maps the data $\mathbf{x}$ to a lower-dimensional representation $\mathbf{z}$. Then, a Decoder $D_{\theta''}$ reconstructs the original data sample from $\mathbf{z}$ to $\mathbf{x}$. The training objective of the model consists of the reconstruction loss which usually is done by means of the Mean Squared Error. That is, if $f_\theta = \{E_{\theta'}, D_{\theta''}\}$ is the autoencoder and $\hat{\mathbf{x}} = f_\theta(\mathbf{x}) = D_{\theta''}(E_{\theta'}(\mathbf{x}))$ is the reconstruction, then $L_{\text{rec}} = \text{MSE}(\mathbf{x}, \hat{\mathbf{x}})$.

Although they provide a feasible solution for dimensionality reduction and data compression, autoencoders suffer from a lack on diversity and provide a narrow latent space, due to their deterministic nature. They fail specially for generation related problems, where we not only want to represent an element of the data, but a distribution of its features in the latent space. Thus, the goal is to estimate a latent distribution were we can sample a latent element and then reconstruct a never-seen element.

Another approach to this issue comes from the hand of GANs (Goodfellow et al., 2014), where latent representations play a crucial role. GANs are a type of generative model that consists of two parts, a generator $G_\theta$ and a discriminator $D_\phi$. The former is trained to generate samples from a latent space $\mathbf{z}$, the latter learns to distinguish between real and fake samples.

Once trained, the latent space can be used to generate new elements by sampling from a latent distribution $p_{\mathbf{z}}$ and feeding the samples to the generator. GANs excel at generating high quality samples and are able to generate novel outputs. However, they are known to be unstable and hard to train, and they suffer from mode collapse, where the generator learns to produce limited outputs, failing to capture the diversity of the data distribution.

As an alternative to GANs, and extending the latent space of the Autoencoders to a stochastic one, we find the Variational Autoencoders (Kingma and Welling, 2019, Kingma and Welling, 2022). Recent advances in generative modelling rely on the use of a type of VAE to work on a better representation of the raw data (Rombach et al., 2021)). Fundamentaly, VAEs are also composed of two parts, an encoder and a decoder. The difference is that it learns stochastic mappings between the observed space $\mathbf{x} \in \mathcal{D} \subset \mathbb{R}^D$, whose empirical distribution is $q_{\text{data}}(\mathbf{x})$ is typically complicated, and a latent space $\mathbf{z} \in \mathbb{R}^d$, whose distribution can be relatively simple – and usually $d < D$. They are considered as a type of generative model whose intricate training objective can be reduced to the minimization of the Variational Lower Bound 2.5.

VAE models can be divided into two categories, regarding the type of the latent space. On the one hand, we have the *continuous representations*, that follow the above mentioned approach and the latent variables live in a continuous space. Sampling from a latent distribution that is not restricted by an adversarial, contrary to GANs, permits generating highly diverse and feasible samples. Despite their huge success, such models suffer from large variance, which makes them hard to train and unstable. On the other hand, we have the *discrete representations*, where an extra step is added to the encoder to quantize the latent space. Quantization is usually done by means of a technique called Vector Quantization (VQ), which does not suffer from large variance, while also avoids the "mode collapse" problem (Kingma and Welling, 2019). We will focus on the latter, since it is closely related to the topic of this thesis.

### 2.2.1 Vector Quantized Variational Autoencoders

Introduced by Oord, Vinyals, and Kavukcuoglu, 2018 VQ-VAEs are a type of VAE that uses VQ to discretize the latent space. The key idea is to define a set $\mathcal{V}_K$ of $K$ vectors of dimension $d$, $\mathcal{V}_K = \{e_i\}_{i \in [1,...,K]}$, namely a *codebook*. Then, for each row $x$ of $\mathbf{x}$, the output of the encoder $z_E(x)$ is projected to the nearest vector in the codebook, namely, $z_q(x)$, see Figure 2.1.

Two things can be observed from this definition. First, now each $\mathbf{z}$ can be represented by a sequence of integers $k_1, k_2, ..., k_n$, where $k_i \in \{1, ..., K\}$. Second, there is no real gradient to

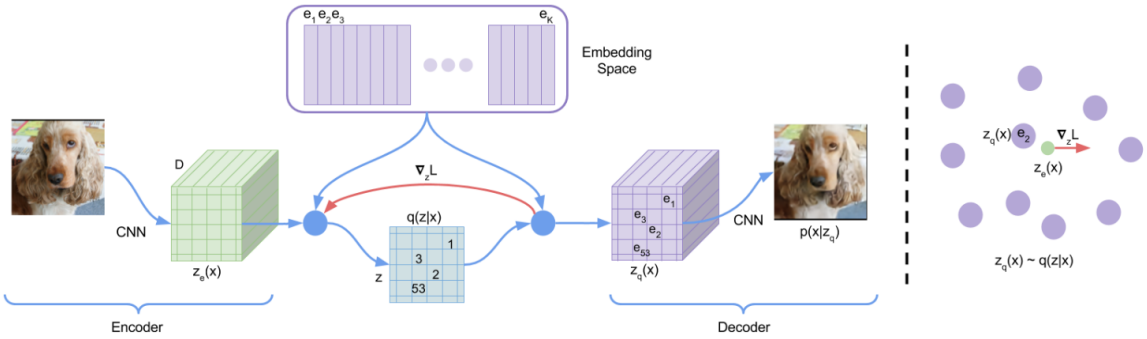


FIGURE 2.1: VQVAE architecture. A sample is encoded into a sequence of tokens, which are then decoded back to the original.

be computed, so the authors propose to use the Straight-Through Estimator (STE) technique, and skip the quantization by copying the gradients from the decoder input to the encoder output. With this, the quantization is only used during the forward pass of the optimization algorithm, and the gradients remain unaltered between the encoder and the decoder when backpropagating. This makes sense since the latents before and after quantization live in the same space, so the gradients contain useful information for the encoder to lower the reconstruction loss. The training objective adds a codebook loss and a commitment loss to the reconstruction loss. The former updates just the codebook vectors, and the latter updates just the encoder output. This is done to make sure that the encoder commits to a codebook and the encoder's output does not drift away from it. The total loss is

$$L = \log p(\mathbf{x}|\mathbf{z}_q(\mathbf{x})) + ||\text{sg}[\mathbf{z}_e(\mathbf{x})] + \mathbf{e}|| + \beta||\mathbf{z}_e(\mathbf{x}) + \text{sg}[\mathbf{e}]||, \tag{2.7}$$

where sg is the stop gradient operator, which is the identity during the forward pass and has partial derivatives of zero during the backward pass to avoid updating the gradient, and $\beta$ is a hyperparameter that controls the strength of the commitment loss.

This model has been shown to be very powerful in the field of image generation, for instance in Esser, Rombach, and Ommer, 2021, they use a VQVAE to encode images into sequences of integers (or *tokens*) and employ a transformer to generate high resolution images. However, the model is not limited to images, and it can be used to encode any type of data. In particular, in this thesis we will follow a similiar approach to encode motion into a sequence of binary tokens.

## 2.3 Latent Diffusion Models

The first work on latent diffusion models was introduced by Rombach et al., 2021. Similar to Ho, Jain, and Abbeel, 2020, they proposed to use a diffusion model to generate images, but running the diffusion process after a VAE, in the latent space, instead of the raw data space. This approach cuts down dramatically the training cost of high resolution generators and makes the inference speed faster. It is motivated by the fact that the semantic information of the data (an image in this case) remains after a compression of the image. There are two important things to notice from their proposed method.

First, the Autoencoder is specifically designed in such a way that the latent space has a perceptual correlation with the data space. This is done by choosing a well suited architecture for the encoder and the decoder which use convolutional layers to capture local correlations. It is also done by minimizing the perceptual loss (Zhang et al., 2018) between

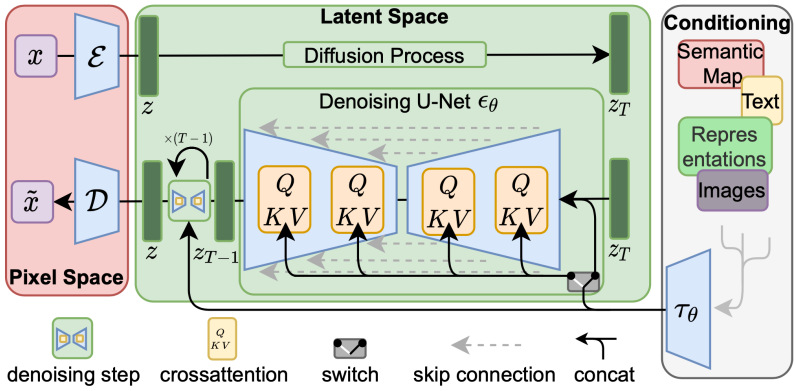


FIGURE 2.2: Architecture of the latent diffusion model. A sample is encoded to to latent space, and a U-Net is used to learn the reverse process of the diffusion model, given a condition. Then, the denoised sample is decoded.

the data and the reconstructed data, and adding an adversarial loss (Yu et al., 2022) that discriminates where a patch of the reconstructed image comes from the original image or from the generator. This makes the latent space to be a good representation of the data space, and it allows the diffusion model to generate high quality fine grained samples.

Second, by reducing to a perceptually simplified space, they exploit a property of the inductive bias in diffusion models which makes them particularly well suited for long-range signal generation. That is, the diffusion process deals with the compostition and semantic structure of the data, and leave the details to the decoder.

The training phase can be divided into two parts, where the VAE is trained to encode the data and then the diffusion model is trained over the latent space, or both models can be trained jointly. Also, any of the already discussed guiding techniques can be added to generate specific samples. In particular, they propose using a U-Net feeded with cross attention layers to embed the conditioning as a denoising function.

## 2.4 Binary Latent Diffusion Models

Since their introduction, latent diffusion models have been improved in several ways until achieving outstanding results in diverse fields and increasing the inference speed to the point of being able to generate high resolution images in almost real time (Sauer et al., 2023). However, although the leading research path has been on improving the Gaussian side of the diffusion model, there are other lines of research that explore the use of other distributions. This can be beneficial for several reasons. For instance, the nature of the data might be better represented by a different latent space, or the diffusion process might be more efficient in a different context. In particular, in this thesis we will focus on the recent work of Wang et al., 2023, where they propose to use a binary latent space instead of a continuous one and build a diffusion model on top of it, introducing the Binary Latent Diffusion Model (BLD).

The main idea of the paper is inspired by Esser, Rombach, and Ommer, 2021, where they propose to use a VQVAE to encode images into sequences of integer tokens, which can be seen as one-hot vectors. Then, a transformer is trained to predict the next element of the sequence to generate high resolution images. This approach is very powerful, but it suffers from a lack of expressivity in the latent space. On the other side, latent diffusion models do not face that problem but relay on a very intensive training process which is computationally expensive. The BLD model combines the best of both worlds by representing a data

point as a sequence of tokens, where each token is a binary vector instead of a one-hot or a continuous vector. This allows the model to provide compact yet expressive representations of the data. Also, they show that by means of a Bernoulli diffusion process it is more efficient to train the model, and it is possible to achieve similar results to the ones obtained by LDM, with 16 times less inference steps, without using any test-time acceleration techniques. This puts the BLD model in a promising position to be used in real applications and test it in different data modalities. Furthermore, using a compact yet expresive representation can be particularly useful when working with motion data, as we will see in the next chapter.

### 2.4.1 Binary Representations

Given a data point $\mathbf{x} \in \mathcal{D} \subset \mathbb{R}^D$, the goal is to learn a bidirectional mapping between itself and its binary representation, that is an autoencoder. Since they introduce their method with images, we will focus on that data modality, but the model can be extended to any other source of data, as we will see in the fore coming chapters. Therefore, we will consider $\mathbf{x} \in \mathbb{R}^{h \times w \times 3}$ to be an image of height $h$, width $w$ and three RGB channels.

The first step is to train an image encoder $E_\theta$, which uses 2D convolutional layers and a ResNet architecture together with a downsampling factor of $k$ to encode the image into a real-valued latent tensor $\mathbf{y} = E_\theta(\mathbf{x}) \in \mathbb{R}^{h/k \times w/k \times d}$, where $d$ is the latent dimension. Then, before quantization, a sigmoid $\sigma$ is applied to normalize the output of the encoder to the range $[0, 1]$.

The quantization into binary vectors can be done in two ways, deterministically or stochastically. In the former, the encoder output is thresholded

$$\mathbf{z} = (\sigma(E_\theta(\mathbf{x})) > 0.5).$$

In the latter, we sample from a Bernoulli distribution and get

$$\mathbf{z} \sim \text{Bernoulli}(\sigma(E_\theta(\mathbf{x}))).$$

Note that in both cases the method does not permit gradient propagation so the STE technique is used to copy the gradients from the decoder input to the encoder output,

$$\tilde{\mathbf{z}} = \text{STE}(\mathbf{z}) = \text{sg}[\mathbf{z}] + \mathbf{y} - \text{sg}[\mathbf{y}].$$

Finally, the decoder $D_{\theta'}$ is given $\tilde{\mathbf{z}}$, which is binary an identical to $\mathbf{z}$ and is trained to reconstruct the image from the binary representation, $\hat{\mathbf{x}} = D_{\theta'}(\tilde{\mathbf{z}})$.

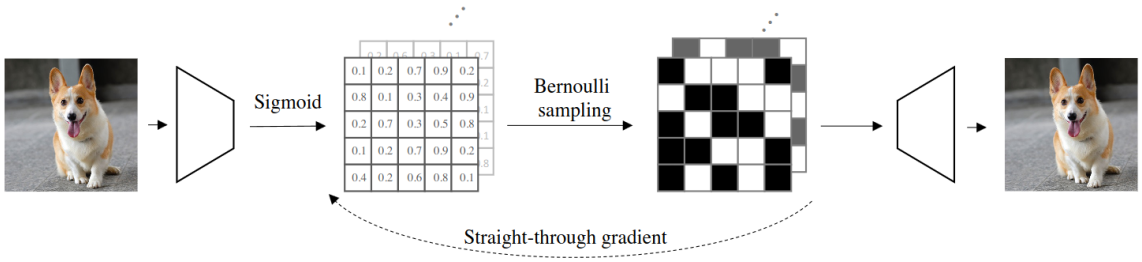


FIGURE 2.3: Binary VAE. The encoder maps the image to a latent space where Bernoulli vectors are sampled. Then the decoder reconstructs the image from the binary representation.

The training objective is different from the VAE model, since there is no need to estimate the posterior data distribution. Instead, the goal is to minimize the reconstruction

loss between the original image and the reconstructed image, which is given by a weighted combination of the Mean Squared Error, the Perception Loss and the Adversarial Loss, as in LDM,

$$L = \omega_{\text{MSE}}\text{MSE}(\hat{\mathbf{x}}, \mathbf{x}) + \omega_{\text{P}} L_{\text{P}}(\hat{\mathbf{x}}, \mathbf{x}) + \omega_{\text{A}} L_{\text{A}}(\hat{\mathbf{x}}, \mathbf{x}).$$

Once trained, on the one hand, the autoencoder is able to encode any image in a much expressive way than a one-hot vector, for instance, a 32-bit binary vector can represent $2^{32}$ different values, which is much more than the 32 values that a one-hot vector can represent. On the other hand, the image information is compacted over less bits compared to a real-valued, authors showed that an 8k-bit binary representation compares to a 131k-bit real-valued representation.

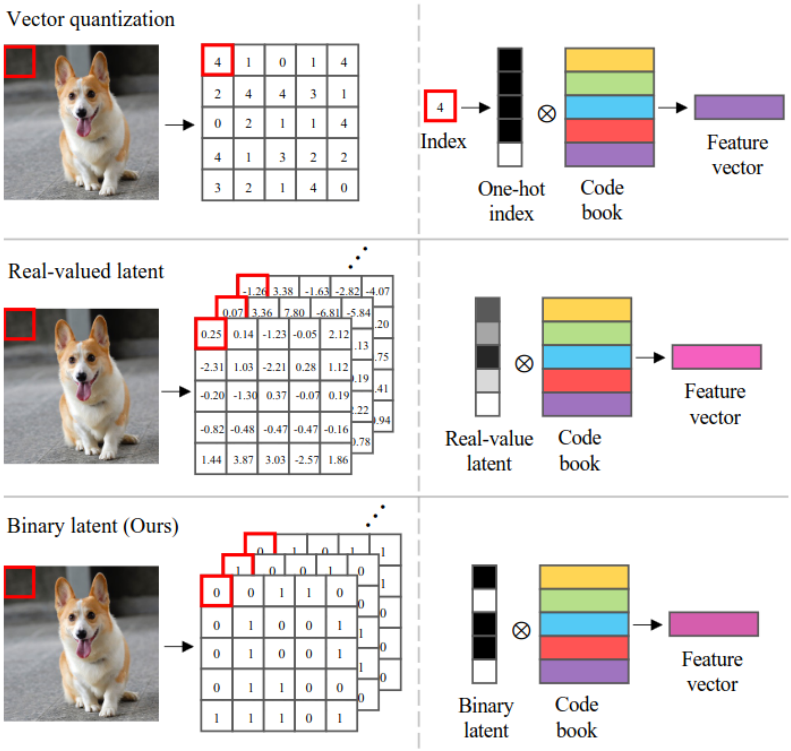


FIGURE 2.4: Heuristic comparison of model expressivity between the binary, continuous and quantized latent spaces.

### 2.4.2 Bernoulli Diffusion Process

Designing a diffusion process specifically tailored to model binomial like distributions is key for the efficiency of the model. As already discussed in the beginning of this chapter, a diffusion process is defined by a Markov chain that gradually transforms a distribution into another. This time, since we are working with binary tensors, the target simple distribution is a binomial, thus the diffusion kernel is similar to 2.2. The procedure is the same but taking into account that now the input of the diffusion model is the distribution of the latent codes $q(\mathbf{z}^{(0)}) = q_{\text{latent}}(\mathbf{z})$, hence the diffusion process becomes

$$q(\mathbf{z}^{(0\ldots T)}) := \prod_{t=1}^{T} q(\mathbf{z}^{(t)}|\mathbf{z}^{(t-1)}), \text{ where} \tag{2.8}$$

$$q(\mathbf{z}^{(t)}|\mathbf{z}^{(t-1)}) = \mathcal{B}(\mathbf{z}^{(t)}; \mathbf{z}^{(t-1)}(1-\beta_t) + 0.5\beta_t). \tag{2.9}$$

With a large enough $T$, the diffusion process will transform the complex distribution into $\mathcal{B}(\mathbf{z}^{(T)}; 0.5)$. Parting from a sample $\mathbf{z}^{(0)}$ and with 2.9, we can compute the posterior distribution of the latent codes in a closed form given an arbitrary step $t$,

$$\begin{cases} q(\mathbf{z}^{(t)}|\mathbf{z}^{(0)}, \mathbf{z}^{(T)}) = \mathcal{B}(\mathbf{z}^{(t)}; \bar{\alpha}_t \mathbf{z}^{(0)} + b_t), \text{ where} \\ \bar{\alpha}_t = \prod_{i=1}^{t}(1-\beta_i), \\ b_t = (1-\beta_t) b_{t-1} + 0.5\beta_t, \text{ and } b_1 = 0.5\beta_1. \end{cases} \tag{2.10}$$

The next step is to define the reverse diffusion process, which is done by training a model $f_\theta$ to estimate the kernel $p(\mathbf{z}^{(t-1)}|\mathbf{z}^{(t)})$, so we obtain

$$p_\theta(\mathbf{z}^{(t-1)}|\mathbf{z}^{(t)}) = \mathcal{B}(\mathbf{z}^{(t-1)}; f_\theta(\mathbf{z}^{(t)}, t)).$$

However, training $f_\theta$ to model the kernel can be challenging as it needs to accurately regress the sophisticated interpolations between $\mathbf{z}^{(0)}$ and $\mathbf{z}^{(T)}$. Therefore, the authors propose two different approaches, either to predict $\mathbf{z}^{(0)}$ directly at each step, or to predict the residual similar to the DDPM model.

- **Direct Prediction:** In this case, the model estimates directly the denoised sample at eacht step $t$,

  $$\hat{\mathbf{z}}^{(0)} = f_\theta(\mathbf{z}^{(t)}, t).$$

  In practice, $f_\theta$ is implemented as a transformer that takes as input the binary sequence $\mathbf{z}^{(t)}$ and predicts the logits of $\hat{\mathbf{z}^{(0)}}$ at each step. Then, during inference, we can recover the denoising kernel by the factorization property of a conditional Bernoulli distribution,

  $$\begin{aligned} p_\theta(\mathbf{z}^{(t-1)}|\mathbf{z}^{(t)}) =& q(\mathbf{z}^{(t-1)}|\mathbf{z}^{(t)}, \mathbf{z}^{(0)} = \mathbf{0}) p_\theta(\mathbf{z}^{(0)} = \mathbf{0}|\mathbf{z}^{(t)}) \\ &+ q(\mathbf{z}^{(t-1)}|\mathbf{z}^{(t)}, \mathbf{z}^{(0)} = \mathbf{1}) p_\theta(\mathbf{z}^{(0)} = \mathbf{1}|\mathbf{z}^{(t)}), \end{aligned}$$

  and the Baye's rule,

  $$q(\mathbf{z}^{(t-1)}|\mathbf{z}^{(t)}, \mathbf{z}^{(0)}) = \frac{q(\mathbf{z}^{(t)}|\mathbf{z}^{(t-1)}, \mathbf{z}^{(0)}) q(\mathbf{z}^{(t-1)}|\mathbf{z}^{(0)})}{q(\mathbf{z}^{(t)}|\mathbf{z}^{(0)})}.$$

  Finally, by following the schedule given by 2.10, we have

  $$p_\theta(\mathbf{z}^{(t-1)}|\mathbf{z}^{(t)}) = \mathcal{B}(\mathbf{z}^{(t-1)}|\frac{[(1-\beta_t)\mathbf{z}^{(t)} + 0.5\beta_t] \odot [\bar{\alpha} f_\theta(\mathbf{z}^{(t)}, t) + 0.5 b_t]}{\mathbf{Z}}), \tag{2.11}$$

  where

  $$\begin{aligned} \mathbf{Z} =& [(1-\beta_t)\mathbf{z}^{(t)} + 0.5\beta_t] \odot [\bar{\alpha} f_\theta(\mathbf{z}^{(t)}, t) + 0.5 b_t] \\ &+ [(1-\beta_t)(1-\mathbf{z}^{(t)}) + 0.5\beta_t] \odot [\bar{\alpha}(1 - f_\theta(\mathbf{z}^{(t)}, t)) + 0.5 b_t], \end{aligned}$$

  and $\odot$ is the element-wise product. The authors provide empirical evidence to show that this approach is more efficient than the naïve one since the target at each step is the same, $\mathbf{z}^{(0)}$, which is strictly binary.

- **Residual Prediction:** In this case, the model $f_\theta$ is trained to predict the residual between the input sample $\mathbf{z}^{(0)}$ and the noisy sample $\mathbf{z}^{(t)}$. Therefore, the target variable is $\mathbf{z}^{(0)} \oplus \mathbf{z}^{(t)}$, where $\oplus$ is the *XOR* operator. Then the original sample can be recovered

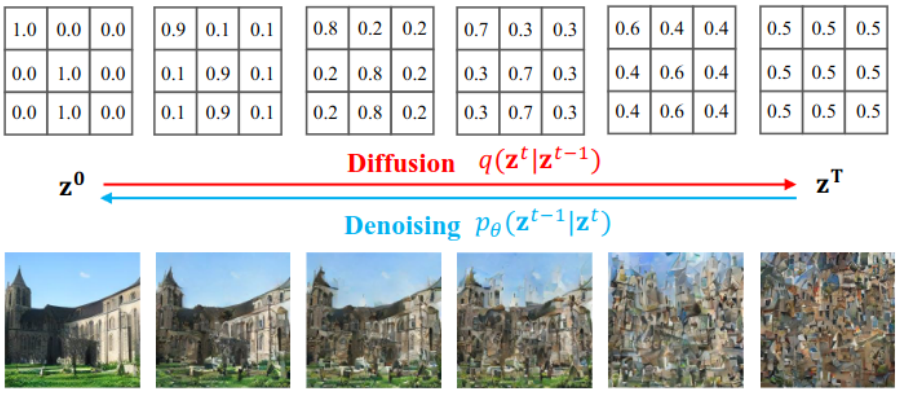


FIGURE 2.5: Bernoulli Diffusion Process.

by

$$\hat{\mathbf{z}}^{(0)} = (1 - \mathbf{z}^{(t)}) \odot f_\theta(\mathbf{z}^{(t)}, t) + \mathbf{z}^{(t)} \odot (1 - f_\theta(\mathbf{z}^{(t)}, t)),$$

and the same reasoning as in direct prediction can be applied to recover the denoising kernel for inference.

The training objective is simplified in a similar fashion to DDPM, by minimizing the Binary Cross Entropy (BCE) between the *target* (either $\mathbf{z}^{(0)}$ or $\mathbf{z}^{(0)} \oplus \mathbf{z}^{(t)}$), and the prediction. Through experimentation, they also propose to use a small $\lambda$-weighted regularization term, which consist on adding a penalty to the prediction of the model based on the $L_{vlb}$ 2.5. The total loss for the diffusion process is given by

$$L = \mathbb{E}_{t,\mathbf{z}^{(0)}} \text{BCE}(f_\theta(\mathbf{z}^{(t)}, t), target) + \lambda L_{vlb}. \tag{2.12}$$

The competitive results obtained after experimentation suggest that their approach to diffusion models is a promising one. In particular, one of the main advantages is that the model efficiently generates high quality samples with a small number of inference steps. Their study shows good results with as few as 8 denoising steps, which in turn DDPM would require 100 to perform comparatively well (Wang et al., 2023). Also, the expressive and compact binary representations seem to be a good choice to better represent other types of data besides images. Based on this we propose to use the BLD model to efficiently generate high quality samples of motion.

# 3 Text-to-Motion Models

As previously discussed in the introductory chapter, text-promoted human motion synthesis encounters several critical challenges. Firstly, the scarcity of large-scale motion datasets and the inherent complexity of non-linear and articulated human motion, pose common obstacles in the realm of Human Motion Generation. Secondly, the absence of a clear and well-defined mapping between text and motion necessitates models to learn intricate mappings capable of capturing the semantic nuances of text and generating diverse, plausible, and realistic sequences of movements. Thirdly, the absence of a clear and well-defined evaluation metric complicates the comparison of different models' performance. In this chapter, we delve into a review of the main contributions from state-of-the-art models, categorizing them into two primary paradigms: diffusion-based models and transformer-based models.

### State-of-the-art Review

Using a text description to create from scratch a sequence of human motion is somewhat related to the task of action-to-motion generation, which usually uses a one-hot vector to synthesize a movement. A text sentence, however, is a more complex representation of an action, and it is not clear how to map it to a vector useful for motion generation. Several natural language processing techniques can be used to extract a vector representation from a text sentence, such as word embeddings, sentence embeddings, or learned transformers. Besides, the first models that were proposed to the tackle text-to-motion (T2M) problem were based on the use of GANs and word embeddings (Sohl-Dickstein et al., 2015, Ahn et al., 2017). The standard has shifted towards the use of foundational models. For instance, CLIP (Radford et al., 2021), which is a pre-trained vision-language model that represents images and text in a common space, is used in most of the recent T2M models to align the text and motion representations. A relevant work that implemented CLIP successfully, as a guide for text-promting motion generation, is MotionCLIP (Tevet et al., 2022). Another example is the use of a pre-trained transformer-based model, such as GPT or BERT (Jiang et al., 2023, Zhang et al., 2023b), which can be fine-tuned to encode the text description into a sequence of tokens that can be used as input to a text conditioning layer.

The raise of diffusion-based models and VQ-VAEs has been a key factor in the development of the subsequent models, increasing the performance of the models and reaching higher levels of state-of-the-art. In this section we will review the main contributions of last years regarding both paradigms. In particular, we will focus on the models that apply diffusion in a continuous motion domain, and the models that use a VQ-VAE to encode the motion in a discrete fashion.

## 3.1 Diffusion Models for Text-to-Motion Generation

Inspired by the success of diffusion models in the field of image generation, efforts have been made to apply these models to HMG. Two of the worth mentioning attempts are Zhu et al., 2023 and Chen et al., 2023a, presenting Motion Diffusion Models (MDM) and Motion Latent Diffusion (MLD) respectively. Since one of the goals of this thesis is to explore the

use of diffusion models for T2M generation, we will delve in detail the main contributions of both models.

Consider $\mathcal{D}$ to be a dataset of pairs $(\mathbf{x}, \mathbf{c})$, where $\mathbf{x} \in \mathbb{R}^{DxL}$ is a sequence of $L$ frames of dimension $D$, according to the type of motion representation used, and $\mathbf{c}$ is a text prompt describing the movement in $\mathbf{x}$. The goal of diffusion models is to learn the conditional distribution $q(\mathbf{x}|\mathbf{c})$, by tunning the parameters $\theta \in \mathbb{R}^N$ of the model $f_\theta(\mathbf{x}, \mathbf{c})$, being $N$ the number of parameters.

- **MDM**'s (Zhu et al., 2023) authors propose using a diffusion model over the human motion data domain. They show remarkable results and provide empirical evidence of the capacity of diffusion models to solve the many-to-many problem intrinsic to HMG. Their model's backbone differs from the standard U-net architecture. Instead, a lightweighted transformer-based architecture is used, since it better fits the temporal and non-spatial nature of the data. The T2M is implemented by encoding the text description with CLIP and freezing its parameters, $\tau(\mathbf{c})$. The motion sequence $\mathbf{x}^{(0)}$ goes through a series of $T$ diffusion steps in a forward manner. Then the model instead of predicting the noise $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$, at each step, it reconstructs the original sequence $\hat{\mathbf{x}}^{(0)} = f_\theta(\mathbf{x}^{(t)}, \tau(\mathbf{c}), t)$. The simple training objective is modified to this end,

$$L_{\text{simple}} = \mathbb{E}_{\mathbf{x}^{(0)} \sim q(\mathbf{x}^{(0)}|\tau(\mathbf{c})), t \sim [1,T]} \left[ \|\mathbf{x}^{(0)} - f_\theta(\mathbf{x}^{(t)}, \tau(\mathbf{c}), t)\|_2^2 \right].$$

  The total loss is complemented with geometric losses common in the motion domain, that enforce physical properties and prevent artifacts. These regularization terms are the velocity loss, the foot contact loss, and the joint position loss (in case $f$ predicts angles instead). The model is ilustrated in figure 3.1.

  During training, each noised sequence $\mathbf{x}^{(t)}$ goes through a linear embedding layer and the CLIP encoding as well as the timestep is concatenated to the sequence. Then, everything is inputed to a positional encoding layer, and a transformer encoder. Finally, the first element of the resulting sequence is discarded, and the rest is the reconstruction $\hat{\mathbf{x}}^{(0)}$. Since the transformer uses a self-attention mechanism, the model is time-aware and able to capture the long-term dependencies of the motion sequence.

  At inference time, as the model predicts $\hat{\mathbf{x}}^{(0)}$ given $\mathbf{x}^{(t)}$, instead of the noise $\epsilon_t$, the prediction is noised back to $\mathbf{x}^{(t-1)}$. This procedure goes from $t = T$ until $t = 1$, and the final sequence is the generated denoised motion.

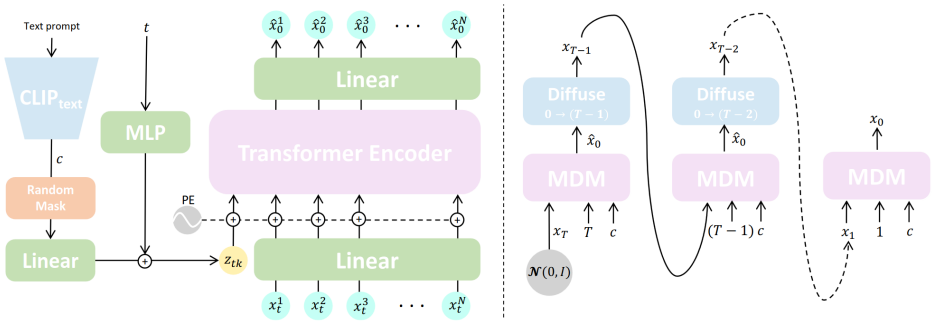


FIGURE 3.1: On the left, the training procedure. On the right the sampling of new motions.

Although MDM's approach is simple and effective, it has some limitations. Diffusion on raw motion sequences is inefficient and is resource demanding. Besides, the data coming from motion capture systems usually contain high-frequency noise, which

might interfere with the performance of the model. One effort to overcome these limitations is to use a VAE to encode the motion sequence into a low-dimensional latent space.

- **MLD** (Chen et al., 2023a) draws inspiration from the success of latent diffusion models. The authors propose a novel framework for T2M generation, where the motion is encoded into a latent space, previous to the diffusion process, then they sample from the latent space a sequence of latent vectors, and finally decode it to a sequence of motion frames.

  To this end, they build a transformer-based VAE. The encoder receives the motion sequence $\mathbf{x}$ and learnable distribution tokens as inputs, the encoded tokens are employed as Gaussian parametes $\mu_{\mathbf{x}}$ and $\sigma_{\mathbf{x}}$ to sample a latent vector $\mathbf{z} \in \mathbb{R}^{n \times d}$. Here $n$ is the number of layers and $d$ is the dimension of the latent space. Then, the decoder is trained to reconstruct the original sequence $\hat{\mathbf{x}}$, only by means of the Mean Squared Error (MSE) loss and the Kullback-Leibler (KL) divergence. To do so, a sequence of zero motion tokens $\mathbf{0}$, with the same length as $\mathbf{x}$ is used as input and the latent vector $\mathbf{z}$ is added via cross-attention. Thus, $\mathbf{z} \sim \mathcal{N}(\mu_{\mathbf{x}}, \sigma_{\mathbf{x}}) = \mathcal{N}(E_{\varphi'}(\mathbf{x}))$, $\hat{\mathbf{x}} = D_{\varphi''}(\mathbf{0}, \mathbf{z})$, and

  $$L_{\text{VAE}} = \text{MSE}(\mathbf{x}, \hat{\mathbf{x}})) + \omega_{\text{KL}} D_{\text{KL}}(\mathcal{N}(0,1) || \mathcal{N}(\mu_{\mathbf{x}}, \sigma_{\mathbf{x}})), \tag{3.1}$$

  where $\omega_{\text{KL}}$ is a hyperparameter that leverages the importance of the KL divergence. Their proposed VAE presents stronger motion reconstruction capabilities and richer diversity than previous similar models (Petrovich, Black, and Varol, 2022).

  The second step of the model is the diffusion process. Similar to MDM, they use a transformer-based model with long skip connections to better fit the sequential data. They follow the same diffusion process as in DDPM (2.1.3), where the noise is added to the latent vector $\mathbf{z}^{(0)}$ and the model predicts the random noise $\epsilon \sim \mathcal{N}(0,1)$ at each step. The conditioning of the text prompt is done in the same way as in MDM, by encoding the text description with CLIP and freezing its parameters. However, they also experiment with other learnable transformer-based text embeddings, $\tau_\phi$. As they follow the DDPM diffusion process, their simple training objective is the same as in 2.6.

  At inference time, random noise is sampled from $\mathcal{N}(0,1)$ and a text prompt $\mathbf{c}$ is given to the model. Both, $\mathbf{c}$ and timestep $t$ are embedded and concadenated to the latent vector $\mathbf{z}^{(t)}$, to predict $\epsilon_t = f_\theta(\mathbf{z}^{(t)}, \tau_\phi(\mathbf{c}), t)$. Then, the latent vector is denoised to $\mathbf{z}^{(t-1)}$ and the process is repeated until $t = 1$. Finally, the latent vector $\mathbf{z}^{(0)}$ is used as memory in the decoder's cross-attention, together with the zero motion tokens $\mathbf{0}$, to generate a new motion sequence. Figure 3.2 ilustrates the MLD model architecture and training procedure. Experiments show that the model's ability to generate diverse and realistic motions is highly dependent on the quality of the latent space. Empirical evidence shows that a smaller latent space leads to better results, which lead to choose a latent dimension of size $n = 1$ and $d = 256$. In this manner, the entire motion is encoded into a single latent vector.

Both approches, MDM and MLD, implement classifier-free guidance 2.1.4, to align the text and motion representations. During training, the conditioning dropped out, $\mathbf{c} = \emptyset$ with 10% of probability, forcing the model to learn unconditional motion generation. Durin sampling, the model receives a text prompt and generates a motion with a linear combination of the unconditional and conditional predictions.

Within the two approaches, MLD present better results than MDM. However, both models demonstrate that diffusion-based techniques are able to properly harness the low-frequency

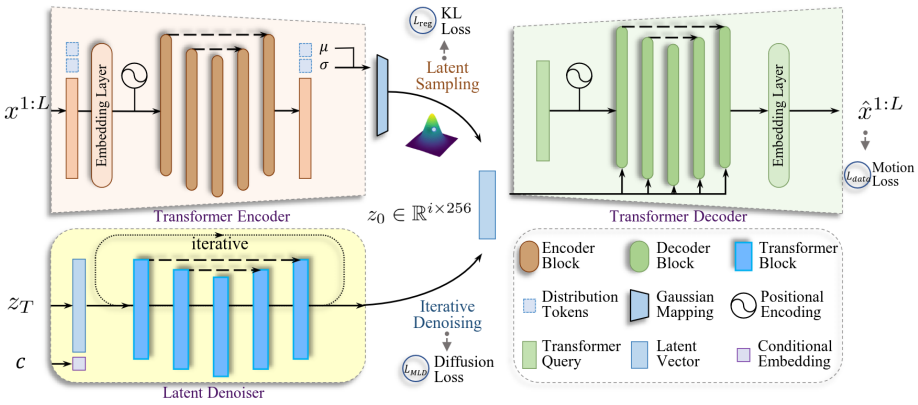


FIGURE 3.2: MLD model architecture. On the left, the training procedure, the model receives the noised sequence $\mathbf{x}^{(t)}$ and the text description $\mathbf{c}$, then it predicts the reconstruction $\hat{\mathbf{x}}^{(0)}$. On the right, the sampling of new motions, given $T$ and random noise.

signals of human motion and generate realistic and diverse motions. Despite of this, there are still some limitations that need to be addressed. First, the diffusion process is still slow and resource demanding. Second, the models are not able to generate long sequences of motion. The forth coming models tackle the problem from a different perspective, aiming to avoid those limitations.

## 3.2 Discrete representations for Motion Generation

Auto regressive models have been the natural candidates to handle sequence to sequence problems, and T2M can be seen as such. By factorizing distributions over the time dimension, predictions can be conditioned on past sequences of arbitrary length. However, the main drawback of these models is that they are costly and inefficient to train. Additionally, when applied to motion data, the models lead to unstable training and unrealistic prediction. Furthermore, during inference, the models accumulate errors over time, which leads to drifts and artifacts (Fragkiadaki et al., 2015, Martinez, Black, and Romero, 2017). Inspired by the success of quantization techniques in the field of image generation, several works have proposed the use of discrete representations for motion generation: PoseGPT (Lucas et al., 2022), T2M-GPT (Zhang et al., 2023b), and MotionGPT (Jiang et al., 2023). Using a VQ-VAE to encode the motion allows to train a model in a lower-dimensional discrete space. Thus, an auto regressive learns to generate the motion by predicting a discrete sequence of tokens, rather than regressing the motion directly. Heuristically, quantizing motion data into discrete tokens can be seen as a way to reduce a complex and high-dimensional problem into a sequence of "words", which can be handled by a language model, i.e. a transformer. We will review in detail these models, since they are closely related to the work of this thesis.

As before, if $\mathcal{D}$ is a dataset of pairs $(\mathbf{x}, \mathbf{c})$, where $\mathbf{x} \in \mathbb{R}^{DxL}$ is a sequence of $L$ frames of dimension $D$, and $\mathbf{c}$ is a text prompt describing the movement in $\mathbf{x}$, the goal of these models is to estimate the conditional distribution $q(\mathbf{x}|\mathbf{c})$, by tunning the parameters $\theta \in \mathbb{R}^N$ of the model $f_\theta(\mathbf{x}, \mathbf{c})$.

- **Pose-GPT** (Lucas et al., 2022) is the first model that uses a VQ-VAE to encode the motion into a discrete latent space, and a transformer-based model to generate the motion. Similar to 2.2.1, the motion $\mathbf{x}$ of length $L$ is first encoded into a sequence of latent vectors $\mathbf{z} = E_{\varphi'}(\mathbf{x})$ of length $L' \leq L$. In order to maintain the temporal

structure of the movement, the encoder consists of a transformer encoder with causal self attention.

Once encoded, the vectors are projected into a codebook of size $K$. The decoder is also a transformer with causal self-attention, which receives the latent vectors $\mathbf{z}_q$ and outputs a sequence of motion $\hat{\mathbf{x}}$. Recall that, since quantization is not differentiable, the gradients are replaced by the straight-through estimator and they use the standard training objective for the VQ-VAE (2.7).

The next step consists in training the transformer-based auto-regressive model $f_\theta(\mathbf{x}, \mathbf{c})$ After quantization, the latent vectors $\mathbf{z}_q$ are represented by a sequence of integers $\mathbf{k}$ of length $L'$, corresponding to the indicies of the codebook vectors. Then, given the first $j$ tokens $\mathbf{k}_{1:j}$, the transformer is trained to maximize the log-likelihood of the next token $\mathbf{k}_{j+1}$. Contrary to diffusion models, human motion is generated sequentially by sampling from the model, and feeding the generated token back to the input sequence. The latent vectors can be recovered by projecting to the codebook vectors $\mathbf{e}_i$. It is important to notice that expetimentally they tested the model on action-to-motion, hence, $\mathbf{c}$ is a one-hot vector that indicates the action to be performed. However, as we will see in the next studies, this can be easily extended.

- **T2M-GPT** (Zhang et al., 2023b) also proposes a VQ-VAE to encode the motion into a discrete latent space, and a transformer-based model to learn the latent motion sequence. However, they use a slightly different approach.

  First, the training objective of the VQ-VAE is enhenced with a reconstruction loss.

  Second, the quantization strategy is different. Instead of a naïve quantization that suffers from codebook collapse (Razavi, Oord, and Vinyals, 2019), they implement a *soft quantization* strategy, and *code reset*, to improve the performance and avoid the issue. The major change relies in the architecture of the encoder and decoder, where the transformer is replaced by a series of one-dimensional convolutional layers, residual blocks, and ReLu, similar to Esser, Rombach, and Ommer, 2021. Downsampling and upsampling is performed by strided 1D-convolutions along the time dimension, to harness the local temporal structure of the motion data into the latent variables. The outcome of the encoder has length $L' = L/2^l$ where $l$ is the number of downsampling layers. The decoder uses nearest inter for upsampling.

  Predicting the next token is done similarly to the previous model. However, they enable the use of a text prompt $\mathbf{c}$, by embedding it via a frozen CLIP model. When sampling, the model starts from the embedded text token and generates the motion sequentially.

  The method achieved state-of-the-art results, proving to be a competitive alternative to the diffusion-based models. However it has a main drawback. Authors observed a slight jitter on the legs and hands movement, they claim that this is due to the VQ-VAE architecture, and proposed that with a better design could it be solved. Furthermore, common to all the models seen until now, the models might miss some details of the motion when given a long description. This is due to the problems inherited from the text embedding layer, that is not able to capture the entire semantics of a long text and encode it into a fixed length vector. The following approach aims to solve this issue by tackling the task from a different perspective, that is understanding motion data as a proper language.

- **Motion-GPT** (Jiang et al., 2023), similar to T2M-GPT, uses a VQ-VAE to encode the motion into a discrete latent space, and a transformer-based model to generate the motion. However the procedure is different. They note that human motion exhibits a

semantic coupling similar to natural language, which is interpreted as body language. Upon this observation, they propose to treat motion as a foreign language. In this manner, the motion and language data are integrated together into the same vocabulary. Their proposed framework uses exactly the same VQ-VAE architecture as the previous model to create a "motion vocabulary". Thus, a sequence of motion $\mathbf{x}$ is encoded into a sequence of integers $\mathbf{k}$, which are treated as words. Then, a model is trained to jointly learn descriptions and movement combining the data into a single vocabulary $V = \{V_t, V_m\}$. Following this reasoning, the training strategy is similar to other text-to-text models, where the model predicts the next token from $V$ given the previous ones.

Despite the promising results obtained by MotionGPT, one limitation of the model is that they assume that motion can be represented as a sequence of words. While this assumption seems plausible, in practice the codebook is very limited to $K$ vectors, which might not be enough to represent the entire motion vocabulary. Based on this, we propose using a Binary Latent Diffusion model, with a quantization strategy that permits encoding motion token in a wider vocabulary. Recall that in the previous chapter 2.4, we saw that the integers of the codebook can be seen as one-hot vectors of size $K$, allowing only $K$ different tokens. However, a binary latent vector of the same size can represent $2^K$ different tokens, a much higher degree of information.

These methods claim that motion data is highly redundant, specially when recorded at high FPS. This redundancy can be exploited by a VAE and further compressed into a discrete latent space with a VQ-VAE. We propose a similar approach, but using a compact yet expressive latent space, given by a binary VAE 2.3. In other to achieve T2M, we propose adapting the Bernoulli diffusion process to the binarized motion data, exploiting the diffusion model's generation capabilities.

# 4 Methodology: Motion Binary Latent Diffusion

Chapter 3 highlighted that current state-of-the-art models in HMG predominantly employ either Diffusion Models or VQ-VAEs with a transformer architecture. The former models showcase the effectiveness of a diffusion process in destroying motion data through the addition of noise, followed by successful reconstruction. Chen et al., 2023b outperforms Zhu et al., 2023 in both performance and cost-efficiency. In alignment with these findings, we propose incorporating a VAE as a precursor to the diffusion model. Despite the remarkable outcomes achieved by these models, diffusion models for motion data generation have been surpassed by models that use a discrete representation for motion data.

Radically different to motion diffusion models, Jiang et al., 2023, Zhang et al., 2023b and Lucas et al., 2022, introduced the use a VQ-VAE. Paraphrasing the authors of MotionGPT, the intrinsic semantics within a human movement have a correspondance with natural language, which we call body language. In this sense, the three models are deeply related. Adding a vector quantization to the VAE, which uses a finite codebook to encode motion data into tokens, translates sequences of poses into "words". A tokenized movement is nothing else but a sentence written in an imaginary language, in which one can either look for the likemost probable subsequent word based in a text conditioning, or merge text and motion and learn both languages at the same time. These models effectively generate plausible and diverse sequences of movement, outperforming the existent diffusion models. Despite of that, such models are very sensitive to a proper training of the VQ-VAE as Zhang et al., 2023b point out. We observe that although there exists a semantic coupling between human motion and natural language, body language can be more subtle, intricate and diverse than mare "words". Therefore, we require a quantization method that allows for a wider expressivity. Note that an action can be performed in several ways and each pose may require the semantic meaning of many words to be properly represented. For instance, running can be fast or slow, can have an intention and be a happy run or an agressive one, can be in a specific place, or interacting with another entity in a scene. Hence, the latent poses of the movement may not be well represented in a single word, they may require from the vocabulary of the codebook the "words" with semantic meaning, "run", "slow", "happy", "mountain", besides other words describing physical posture, behaviour or purpose of the motion. Furthermore, at every step, the meaning of the pose can change, at least the physical posture itself, leading to a different representation. With this in mind, after quantization we do not want a one-hot representation, which would be picking one word of the dictionary, but a binary vector representation, that is vector of zeros and ones indicating which are the useful words within the codebook that properly represents the encoded motion. The codebook size ot the above methods is $K = 1024$, thus there are only 1024 possible motion tokens or latent pose representations. However, as discussed in 2.4.1, a codebook of size $K = 32$ allow for $2^{32}$ different token representations. We claim that by means of binary quantization, human motion can be encoded in a much expressive manner leading to a richer generative model able to synthesize high-quality motion data. To this end, inspired by the huge success of diffusion models as generative priors in other data domains, we propose adapting the Binary Latent

Diffusion model. In this section we will discuss different ways of motion binarization, discuss the adaptation of the BLD and detail the implementation of the Motion Binary Latent Diffusion model.

## 4.1 Motion Binary Variational Autoencoder

Let the data domain $\mathcal{D}$ be a human motion dataset, $\mathbf{x}_{1:L} \in \mathcal{D}$ a motion sequence of length $L$, and $\mathbf{c}_\mathbf{x}$ a text prompt describing the movement. Each frame of the movement consists of a human pose represented by a vector of rotations of dimension $J$, thus $\mathbf{x}_i \in \mathbb{R}^J$, for $i = 1, ..., L$. Our goal is to define a Motion Binary VAE (MBVAE) similar to 2.3, that encodes the motion sequence into a binary latent and reconstructs the original motion from it. Within this section we provide a discussion on how to define the MBVAE and test different versions inspired from the VQ-VAEs of the models from chapter 3. We will experiment with three different types of latent space. First, we encode the motion frame-wise, that is, each frame is encoded into a binary vector. Second, chunks of frames are encoded into a single binary vector, reducing the length of the latent motion. Third, the whole sequence is encoded into a single binary vector. We will refer to these three types of latent space as *frame-wise*, *some-frames* and *full-sequence* respectively.

### 4.1.1 Frame-wise binary quantization

Directly applying the binary VAE from 2.3 to the motion sequence requires to encode each frame into a binary vector. To this purpose, the convolutional layers of the original model are replaced by 1-dimensional convolutional layers. Thus, if $\mathbf{x}_i \in \mathbb{R}^J$ is the $i$-th frame of the sequence, the encoder extracts the local relation between the rotations of the pose and encodes it into a binary vector of reduced dimension. To be more precise, a linear embedding layer is applied to the input pose, reducing its dimension to $J' < J$. Then, a 1D convolutional layer with kernel size 3 and stride 1 is applied to the embedded pose. We use $c'$ of such filters, thus the output after the convolutional step is a tensor of dimension $c' \times J'$. Then a block of 2 residual layers, an attention layer and a final convolutional layer with kernel size 3 and stride 2 are applied to the output tensor. The stride of the last CNN reduces the dimension of the latent joints by a factor of 2, performing as a downsampling layer. Repeating this process $l$ times leads to a latent representation of dimension $c \times J'/2^l$. Finally, another convolution with $c$ filters kernel size 3 and stride 1 outputs a real-valued tensor of dimension $c \times J'/2^l$. The latent space is then obtained by applying a sigmoid function to the output tensor, and then a binary quantization. The decoder is the inverse of the encoder, with nearest neighbor interpolation layers as upsamplers.

We propose two different ways of quantizing the motion latent space.

- **Binary:** As a first approach, we apply the binary quantization straightforwardly to the latent space. This is, sampling from a Bernoulli distribution with probability of success equal to the latent value, as in 2.4.1. Therefore, for any $\mathbf{x}_{1:L}$, and given the encoder $E_{\theta'}$, for $i = 1, ..., L$, the latent representation of the $i$-th frame is $\mathbf{z}_i = E_{\theta'}(\mathbf{x}_i)$, and the binary latent representation is $\mathbf{b}_i \sim \text{Bernoulli}(\mathbf{z}_i)$. The latent motion sequence is then $\mathbf{b}_{1:L} = (\mathbf{b}_1, ..., \mathbf{b}_L)$. The decoder $D_{\theta''}$ reconstructs the original motion from $\mathbf{b}_{1:L}$.

- **BinaryVQ:** The second approach is to apply the binary quantization prior to the VQ-VAE. That is, we define a VQ-VAE as in 2.1, with codebook of size $K$, and then apply the binary quantization to the latent space. Thus, for any $\mathbf{x}_{1:L}$, given the encoder $E_{\theta'}$, and the codebook $C_K$, for $i = 1, ..., L$, the latent representation of the $i$-th frame is

$\mathbf{z}_i = E_{\theta'}(\mathbf{x}_i)$, and the binary latent representation is $\mathbf{b}_i = \text{bin}(\mathbf{z}_i)$. The binary quantization function $\text{bin}(\cdot)$ is composed by a projection layer that maps $\mathbf{z}_i \in \mathbb{R}^{c \times J'/2^l}$ to $\mathbf{b}_i \in \{0,1\}^{K \times J'/2^l}$, and a Bernoulli sampling layer. The resulting binary tensor $\mathbf{b}_i$ is then passed to the vector quantization layer, where the binary tensor is multiplied by the codebook $C_K$ to obtain the quantized tensor $\mathbf{q}_i$. Heuristically, in this approach, each column of the tensor $\mathbf{b}_i$ is a binary vector of length $K$, with ones in the positions of the vectors of the codebook that best represent the latent joint of the pose. The decoder $D_{\theta''}$ reconstructs the original motion from $\mathbf{q}_{1:L}$. Observe that both approaches are equivalent, since the decoder $D_{\theta''}$ is the same in both cases, and in the first method it could learn an implicit codebook inside its convolutional and residual layers. Figure 4.1 shows a comparison between both of them.

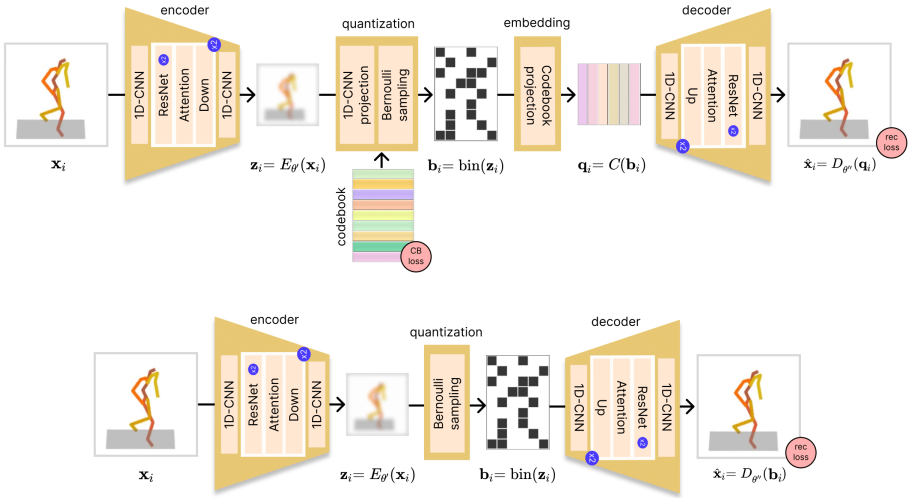


FIGURE 4.1: Top: MBVAE with BinaryVQ. Bottom: MBVAE with Binary. The latter is equivalent to the former, since the decoder could learn an implicit codebook inside its convolutional and residual layers. As input, a pose in your favourite motion representation format. As output, its reconstruction.

### 4.1.2 Some-frames binary quantization

Although the frame-wise representation successfully encodes the motion sequence into a robust binary latent space, it has a drawback. Poses in movement data are highly correlated and redundant, specially when recorded at high frame rates. Therefore, encoding each frame into a binary vector can be inefficient. To overcome this problem, in spirit of the VQ-VAE from Zhang et al., 2023b and Jiang et al., 2023, we propose applying the 1D convolutional layers along the temporal dimension, and then binarize the latent space. Therefore, model is the same as the frame-wise binary quantization, but $\mathbf{x}_{1:L}$ is processed sequentially, that is, the kernel of the 1D CNNs moves along the $L$-dimensional temporal sequence, combining the frames the kernel size permits. With the same notation as before, and the new direction of the convolutional layers, the encoder extracts the local temporal relation between the rotations of the pose and outputs $\mathbf{z} \in \mathbb{R}^{c' \times L/2^l}$. Here, $c'$ can be seen as the latent joints and $L/2^l$ as the number of latent frames. Binary quantization and decoding also remain the same, only considering the new dimensions of the latent space.

When applying the binary quantization, each of the binary columns of length $K$ of $\mathbf{b} = \text{bin}(\mathbf{z})$ represents a latent frame. Therefore, the ones in the binary vector indicate which are

the elements of the codebook that best represent the entire latent frame. Observe that this representation is more compact than the frame-wise one, since it encodes the whole latent frame into a single binary vector 4.2.

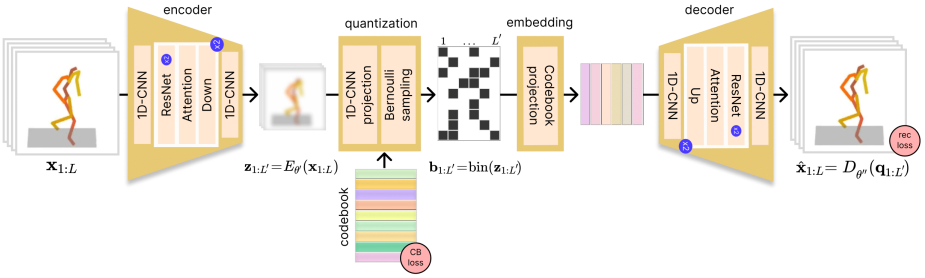


FIGURE 4.2: MBVAE with BinaryVQ and some-frames binary quantization. As input, a sequence of the entire motion.

### 4.1.3 Full-sequence binary quantization

The semantic meaning of a movement is not only encoded in the poses, but also in the temporal relation between them. In this sense, the frame-wise and some-frames binary quantization methods may not be able to capture the whole significance of the motion. Therefore, we propose using the full-sequence binary quantization, that is, as in 3.2, the encoder processes the whole sequence at once, and then binarizes the latent space. Then a transformer decoder manages the latent representation as cross-attention context, and reconstructs the original motion given a sequence of zero-motion tokens $\mathbf{0}$, as in MLD. Thus, $\mathbf{x}_{1:L}$ is processed all at once by the transformer encoder $E_{\theta'}$, and outputs the mean $\mu_{\mathbf{x}}$ and variance $\sigma_{\mathbf{x}}$ of a Gaussian distribution $\mathcal{N}(\mu_{\mathbf{x}}, \sigma_{\mathbf{x}}) = \mathcal{N}(E_{\theta'}(\mathbf{x}))$. The binary quantization is done after the sampling step, $\mathbf{z} \sim \mathcal{N}(\mu_{\mathbf{x}}, \sigma_{\mathbf{x}})$, thus $\mathbf{b} = \text{bin}(\mathbf{z})$. The decoder $D_{\theta''}$ reconstructs the original motion from $\mathbf{b}$ and $\mathbf{0}$, leading to a sequence of poses $\hat{\mathbf{x}}_{1:L} = D_{\theta''}(\mathbf{b}, \mathbf{0})$. From the binary vector quantized perspective this can be seen as choosing the best words from the codebook that better represent the entire sequence, see Figure 4.3.

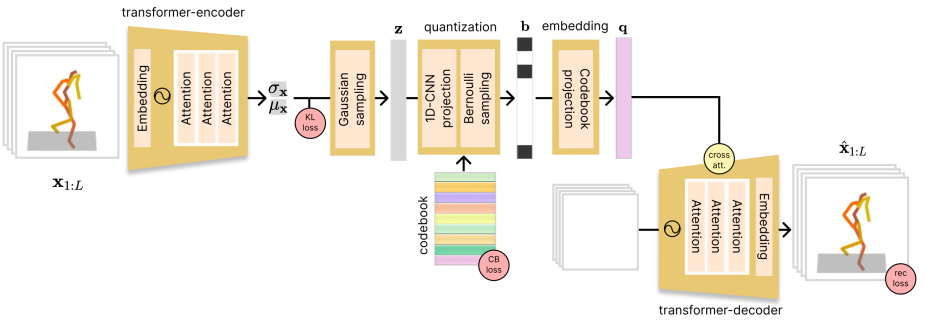


FIGURE 4.3: MBVAE with BinaryVQ and full-sequence binary quantization. Input is also a sequence of the entire motion.

### 4.1.4 Training objective

As a simple training objective for the three methods, we propose to minimize a reconstruction loss between $\mathbf{x} = \mathbf{x}_{1:L}$ and $\hat{\mathbf{x}} = \hat{\mathbf{x}}_{1:L} = D_{\theta''}(\mathbf{q}_{1:L})$, that is, a weighted sum of the mean

squared error and the smooth $L_1$ loss. Observe that for the sake of simplicity, we do not consider velocities and other features of the pose, but only the rotations,

$$L_{\text{rec}}(\mathbf{x}, \hat{\mathbf{x}}) = \omega_{\text{MSE}}\text{MSE}(\mathbf{x}, \hat{\mathbf{x}}) + \omega_1 L_1(\mathbf{x}, \hat{\mathbf{x}}),$$

where $\omega_{\text{MSE}}$ and $\omega_1$ are hyperparameters that weight the contribution of each loss. In case of the full-sequence binary quantization, a further regularization term is added to the training objective. Similar to the MLD variational autoencoder 3.1, the KL divergence between the prior distribution and the posterior distribution of the latent space is also considered, leading to the following training objective,

$$L(\mathbf{x}, \hat{\mathbf{x}}) = L_{\text{rec}}(\mathbf{x}, \hat{\mathbf{x}}) + \omega_{\text{KL}} D_{\text{KL}}(\mathcal{N}(0,1) || \mathcal{N}(E_{\theta'}(\mathbf{x}))).$$

Besides the reconstruction loss and the KL regularization, whenever we use a codebook, we add a commitment loss to the training objective, as in 2.7. Additionally, recall that the binary quantization is non-differentiable, thus we use the straight-through estimator to back-propagate the gradients.

## 4.2 Motion Binary Latent Diffusion Model

With a learned binary latent space, we propose adapting the Binary Latent Diffusion model to motion data, introducing the Motion Binary Latent Diffusion (MBLD) model. The main difference with the original model relies within the architecture of the denoising function. In the original model, they use a transformer decoder with 2-dimensional attention layers. However, in our case, the latent space is a binary vector, hence we propose using a similar denoising function as in 3.1. That is, a transformer decoder with 1-dimensional self-attention layers along the temporal axis that we denote by $f_\theta$. The reason why we use the same architecture as in the MDM model is because it already has been proven to work well for motion data in the raw motion domain, and we expect it to work well in the binary latent domain, since the binary latent representation $\mathbf{b}_{1:L'}$ can be seen as a motion sequence of length $L'$. Notice that $\mathbf{b}_{1:L'}$ has latent frames and latent joints or rotations, in concordance with the input from the original model of the MDM.

We explore the performance of the model with the frame-wise binary quantization. In this case, the sequence $\mathbf{x}_{1:L}$ with textual description $\mathbf{c}$ is encoded to $\mathbf{b}_{1:L}$, and the Bernoulli Diffusion Process 2.4.2 is applied. The denoising function $f_\theta$ is trained to predict the flip probability of each bit at each step of the diffusion process, given the text contidion. Recall that the procedure begins with $\mathbf{b}^{(0)}_{1:L}$, where the superindex indicates the step of the diffusion Markov chain. Then, at each step $t$, $f_\theta$ estimates the binary tensor $\mathbf{b}^{(0)}_{1:L} \oplus \mathbf{b}^{(t)}_{1:L}$, where $\oplus$ is the element-wise XOR operation, or the original sample $\mathbf{b}^{(0)}_{1:L}$, as in BLD. When training, a random step $t$ is sampled from a uniform distribution $t \sim \mathcal{U}(0, T)$, where $T$ is the number of diffusion steps. Adding noise to the binary latent space is done by the schedule 2.10 defined in chapter 2, with a linear distribution of the $\beta_t$. Additionally, the text $\mathbf{c}$ is embedded into a vector of dimension $d$, projected to fit the dimension of the latent space by means of a linear layer, and concatenated to the binary tensor. Classifier-free guidance is used to condition the diffusion process with the text prompt 2.1.4, that is, at inference time, the denoising step is computed by the following equation,

$$(1 + \omega) f_\theta(\mathbf{x}_t, \mathbf{c}, t) - \omega f_\theta(\mathbf{x}_t, \varnothing, t).$$

Therefore, $f_\theta$ must learn both, the conditional and unconditional denoising steps. Figure 4.4 ilustrates the diffusion process and sampling procedure.

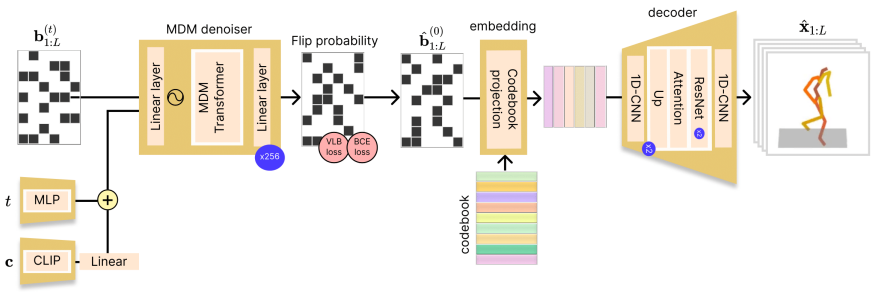


FIGURE 4.4: MBLD with frame-wise binary quantization. When training $\mathbf{b}^{(0)}_{1:L}$ is a sequence of binarized poses, encoded from the target motion. When sampling, input is $\mathbf{b}^{(0)}_{1:L}$, a random binary tensor, and the models denoise it in $T = 256$ steps.

### 4.2.1 Training objective

The training objective minimizes a weighted combination of the Binary Cross Entropy between the predicted and the true binary tensor from the XOR operation, and the KL divergence between the prior and the posterior distribution of the latent space, as in 2.12, leading to the following training objective,

$$L = \mathbb{E}_{t,\mathbf{z}^{(0)}} \mathrm{BCE}\Big(f_\theta\big(\mathbf{b}^{(t)}_{1:L}, \mathbf{c}, t\big), \mathbf{b}^{(0)}_{1:L} \oplus \mathbf{b}^{(t)}_{1:L}\Big) + \lambda L_{vlb}, \tag{4.1}$$

where $L_{vlb}$ is the variational lower bound, i.e. the KL divergence 2.5, and $\lambda$ is a hyperparameter that weights the contribution of each loss. During training, we reconstruct the original binary latent at each step with

$$\hat{\mathbf{b}}^{(0)}_{1:L} = (1 - \mathbf{b}^{(t)}_{1:L}) \odot f_\theta(\mathbf{b}^{(t)}_{1:L}, t) + \mathbf{b}^{(t)}_{1:L} \odot (1 - f_\theta(\mathbf{b}^{(t)}_{1:L}, t)),$$

and keep track of an accuracy metric, which measures the percentage of bits that are correctly predicted by the denoising function.

### 4.2.2 Sampling

In order to generate new motion sequences from $\mathbf{c}$, we start from a random binary tensor $\mathbf{b}^{(t)}_{1:L}$ sampled from a Bernoulli distribution with probability of success 0.5. Then, we run the denoising process. At each step $t$ we estimate $f_\theta\big(\mathbf{b}^{(t)}_{1:L}, \mathbf{c}, t\big)$ and $f_\theta\big(\mathbf{b}^{(t)}_{1:L}, \mathbf{0}, t\big)$, to compute the classifier-free guidance. Then, from 2.11 we can compute the transition kernel $p_\theta(\mathbf{b}^{(t-1)}_{1:L} | \mathbf{b}^{(t)}_{1:L})$ and sample the next binary tensor $\mathbf{b}^{(t-1)}_{1:L}$ from it. Repeating this process $T = 256$ times, we obtain a sequence of binary tensors $\mathbf{b}^{(0)}_{1:L}$. Finally, we reconstruct the original motion sequence by means of the decoder $D_{\theta''}$, that is, $\hat{\mathbf{x}}_{1:L} = D_{\theta''}(\mathbf{b}^{(0)}_{1:L})$.

# 5 Experiments

The experimental results are presented in the following sections. The first part of the chapter entails a discussion of the implementation details, evaluation metrics, and datasets. Subsequently, we delve into the results obtained by the MBVAE, offering a detailed analysis of its performance across various experiments, emphasizing metrics such as reconstruction loss. The last section focuses on the outcomes derived from the MBLD model, exploring the influence of the diffusion process on the binary latent space and assessing its denoising capabilities.

## 5.1 Experimental Setup

### 5.1.1 Text-to-Motion Datasets

Common human motion datasets that include text descriptions are scarce. The most relevants are include KIT (Plappert, Mandery, and Asfour, 2016) and the recently released HumanML3D (Guo et al., 2022). Since the latter is bigger and common to all state-of-the-art models, we have decided to use HumanML3D. It is a large-scale dataset that contains 14,616 motion sequences, along with 44,970 sentences describing the motion. The motion data comes from HumanAct12 and AMASS (Mahmood et al., 2019), which are collections of several smaller motion captured datasets. Authors of HumanML3D have standarized the motion sequences to 20FPS and to a default human skeletal template. Motion sequences longer than 200 frames have been randomly cropped to fit this restriction. As a result each motion clip is of minimum and maximum length of 40 and 200 frames, respectively. Each pose $\mathbf{x}_i$ of the motion $\mathbf{x}_{1:L}$, is defined as a 263-dimensional vector, which includes positions, rotations and orientations of 22 joints, and a global root position.

### 5.1.2 Evaluation Metrics

Although assessing the quality of synthesized human movement is not a trivial task, several metrics have been proposed to measure the performance of the models from different perspectives. They can be summarized in three categories fidelity, diversity and condition consistency (Zhu et al., 2023, Chen et al., 2023b). Following the autors of MLD and T2M-GPT, we will focus on the following metrics; MSE, FID, DIV, MM, R-Precision and MMD, each of them measuring a different aspect of the model's performance.

**Fidelity** metrics aim to evaluate the overall quality of the generated motion.

1. **Mean Squared Error (MSE)**: measures the average of the squared differences between the prediction and the real motion. Relying solely on comparison to the ground truth joints and rotations is not enough to assess the quality of the generated motion, since countless of alternative sequences could be equally valid, but not similar to the ground truth.

2. **Fréchet Inception Distance (FID)**: estimates the distance between the distribution of a feature space of the generated motion and the ground truth. The metric leverages well-designed motion feature extractors, and uses the Fréchet distance between two

multivariate Gaussians. To this end, given a generative model $f_\theta$ and real data $\mathcal{D}$, one can synthesize an artificial dataset $\mathcal{D}'$, to then fit $\mathcal{N}(\mu, \sigma)$ and $\mathcal{N}(\mu', \sigma')$, respectively. Then, the FID is computed as follows,

$$\text{FID} = d_F(\mathcal{N}(\mu, \sigma), \mathcal{N}(\mu', \sigma'))^2 = \|\mu - \mu'\|_2^2 + \text{tr}\big(\sigma + \sigma' - 2(\sigma\sigma')^{\frac{1}{2}}\big).$$

**Diversity** metrics aim to measure the model's ability to generate different motions.

1. **Diversity (DIV)**: measures the global diversity of the model by randomly splitting the generated dataset into two sets of the same size $S$. Then

$$\text{DIV} = \frac{1}{S}\sum_{i=1}^{S} \|\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_j\|_2^2,$$

   where $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{x}}_j$ are two random samples from the two sets, respectively. In our experiments we set $S = 300$.

2. **MultiModality (MM)**: measures the diversity but conditioning to a set $\mathcal{C}$ of size $C$, of different text prompts. To this end, consider the generated motion sequences that satisfy a condition $c \in \mathcal{C}$, i.e. $\mathcal{D}'_{\mathbf{c}} := \{\hat{\mathbf{x}} \in \mathcal{D}' \mid \mathbf{c} \text{ is a description of } \hat{\mathbf{x}}\}$. Then, split $\mathcal{D}'_{\mathbf{c}}$ into two sets of the same size $S'$, and compute the diversity as before,

$$\text{MM} = \frac{1}{C \cdot S'}\sum_{c \in \mathcal{C}}\sum_{i=1}^{S'} \|\hat{\mathbf{x}}_{\mathbf{c},i} - \hat{\mathbf{x}}_{\mathbf{c},j}\|_2^2,$$

   where $\hat{\mathbf{x}}_{\mathbf{c},i}$ and $\hat{\mathbf{x}}_{\mathbf{c},j}$ are two random samples from the splits of $\mathcal{D}'_{\mathbf{c}}$, respectively.

**Condition Consistency** metrics aim to measure the accuracy of the model to generate motions that satisfy a given condition.

1. **R-Precision**: measures the accuracy of the model to generate motions that satisfy a given condition. It ranks the Euclidean distances among the features of the generated motion and the features of the ground truth, and averages the accuracy of the top-$k$ results.

### 5.1.3 Implementation details

Throughout the training process, the models leverage the HumanML3D dataset, and akin to other methodologies, we employ CLIP ViT-B/16 as a text encoding prior. Such CLIP version, encodes text into a vector of dimension $d = 512$. This standardized approach ensures consistency and facilitates seamless integration with existing frameworks and datasets, contributing to the reproducibility and transparency of the research outcomes. The models are trained with the Adam optimizer and a learning rate of $10^{-4}$. The hyperparameters of the training objective are set to $\omega_{\text{MSE}} = 1$, $\omega_1 = 1$, $\omega_{\text{KL}} = 0.1$. All versions run for 400 over the training set of 1528 motion samples from HumanML3D, which are no longer than $L = 196$, to avoid selecting randomly cropped clips, and no shorter than 150. We remain with 1528 different motion samples, with 4 descriptions each. Sequences are padded with zeros until reaching the maximum length. The linear projection fits the $J = 263$ joints and rotations of the motion poses into a $J' = 256$-dimensional vector. The codebook size is set to $K = 32$ for the binary vector quantized models. For frame-wise models, the downsampling factor is set to $l = 4$ and the inner channels of the encoder are $c' = 16$, which leads to a latent space of size $16 \times 16$. The some-frames models have downsampling factor $l = 2$ and a latens dimension $16 \times L'$, where $L' = 196/2^2 = 49$. This is different in the case of the full-sequence

models, where $K = 256$ and the latent space for the entire motion is reduced to size 256, as MLD authors suggest (Chen et al., 2023b). The guidance weight for the Classifier-free Guidance is set to $\omega = 0.5$. Finally, the steps for the diffusion process and the sampling process are set to $T = 256$, as in the BLD (Wang et al., 2023).

All the implemented models are developed using PyTorch and trained efficiently on a single NVIDIA GeForce RTX 3090 GPU. The complete source code is accessible on GitHub at motion-binary-latent-diffusion.

## 5.2 Comparisons on MBVAE

Experiments were undertaken to evaluate the performance of the MBVAE across various models. The comparison involved assessing the capabilities of straightforward binary quantization, binary vector quantization, and models without quantization. The goal of this comparison was to determine the most effective approach for representing motion in a binary latent space. Noticeably, the latent space dimensions for frame-wise are the largest, at $16 \times 16 \times 196$ for the complete motion, while the some-frames models have a latent space of size $16 \times 49$, and the full-sequence models have a latent space of size 256. Therefore, the frame-wise models have the most information about the motion at pose level. Consider also that the binary latent space of the vector quantized models is a bit larger, since it is prior to the codebook projection, thus, the binary latent spaces are of size $16 \times 32 \times 196$, $32 \times 49$ and 256, respectively. The latent space of no quantized models is the default 32-bit float, therefore we achieve up to a factor of 32 higher compresion rate. The table 5.1 summarizes the memory footprint of a single motion sequence for each model embedded into the latent space.

| Model | Binary | Binary-VQ | No-Quant |
|---|---|---|---|
| Frame-wise | 50,176bits $\simeq$ 6.3kB | 100,352bits $\simeq$ 12.6kB | 1,605,632bits $\simeq$ 200.7kB |
| Some-frames | 784bits $\simeq$ 0.1kB | 1,568bits $\simeq$ 0.2kB | 25,088bits $\simeq$ 3.1kB |
| Full-sequence | 256bits $\simeq$ 0.03kB | 256bits $\simeq$ 0.03kB | 8,192bits $\simeq$ 1.02kB |

TABLE 5.1: Memory footprint of the different models for a latent representation of a motion sequence of 196 frames and 263 joints and rotations per pose.

The plots 5.1 indicate that frame-wise models exhibit superior fitting to the reconstruction loss, primarily owing to their richer information content about motion at the pose level. Closely following are the some-frame models, which achieve a lower reconstruction loss than the full-sequence models. In terms of MSE and L1 losses, the Binary and Binary-VQ models perform similarly, while the no-quantized models achieve a lower loss. This difference is accentuated in the total loss, in the case of the VQ models, due to the codebook loss 5.2. However, we can observe that if the binary space is large enough, i.e two times bigger in our case, the frame-wise Binary can overcome the No-Quant some-frames model. This is due to the fact that the binary latent space is able to capture the motion information. Therefore, with this simple training objective, the binary latent space is able to achive comparatively similar results to the No-Quant models, despite the need of a similar memory footprint in the latent space. This is a remarkable result, since it indicates that the binary latent space is able to represent motion information, but it needs further regularization to reduce the memory footprint effectively.

Regarding the codebook loss 5.2, we can observe that all models with Biinary-VQ achieved a similar codebook even with different latent spaces and codebook sizes, 32 (frame-wise and some-frames) and 256 (full-sequence). The codebook loss ensures the binary latent space is as sparse as possible, choosing the minimum number of codewords to represent a binary

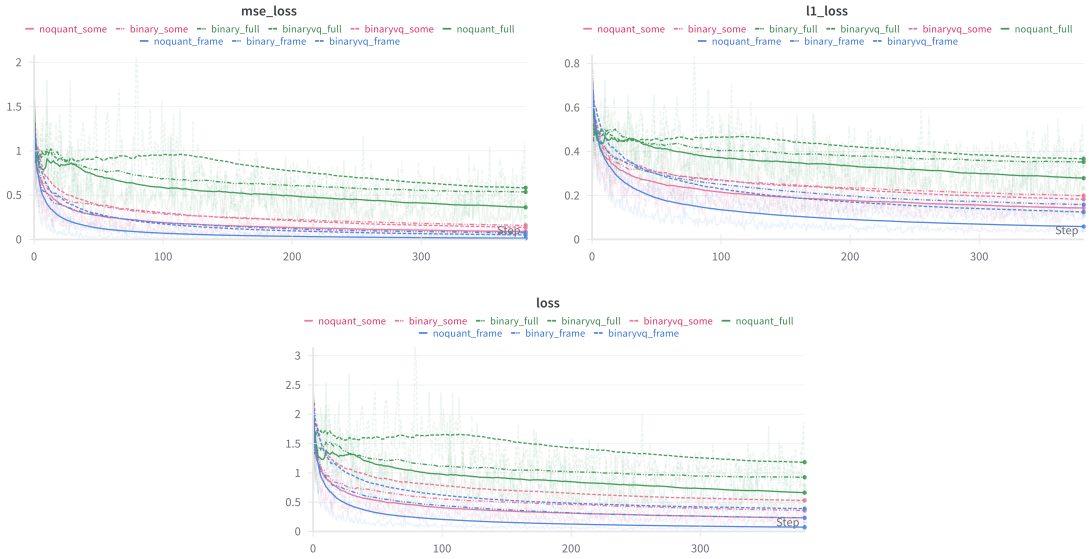


FIGURE 5.1: Comparison of the training losses between **frame-wise**, **some-frames** and **full-sequence**.

vector. Therefore, it will never drop to zero, unless the model collapses. Thus, the fact that all models achieve similar codebook losses indicates that the proportion of codewords used to represent the binary latent space is comparable.

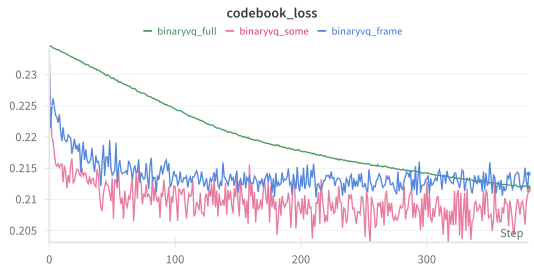


FIGURE 5.2: Codebook loss of Binary-VQ versions of the models at training stage.

Full-sequence models face a trade-off between reconstruction loss and KL divergence, which hampers the training process. Interestingly, even when removing KL regularization, full-sequence models fail to achieve a lower reconstruction loss compared to frame-wise and some-frames models 5.3. This discrepancy arises from their reduced ability to capture pose information, resulting in a latent space that inadequately represents motion.

## 5.3 Exploration on MBLD

Since we observe that the binary latent space is able to represent motion information, we proceed to test the performance of the MBLD through a two-pronged approach.

**Experiment 1: Frame-wise diffusion.** Initially, a diffusion model is trained on the latent space of the frame-wise MBVAE, utilizing a codebook of size $K = 32$ for the BinaryVQ versions. Notably, we observed a pronounced sensitivity to hyperparameter tuning, particularly concerning the balancing weights of the losses. Remarkably, the model exhibits greater stability when estimating the original poses $\mathbf{b}_{1:L}^{(0)}$ instead of the flip probability—contrary to

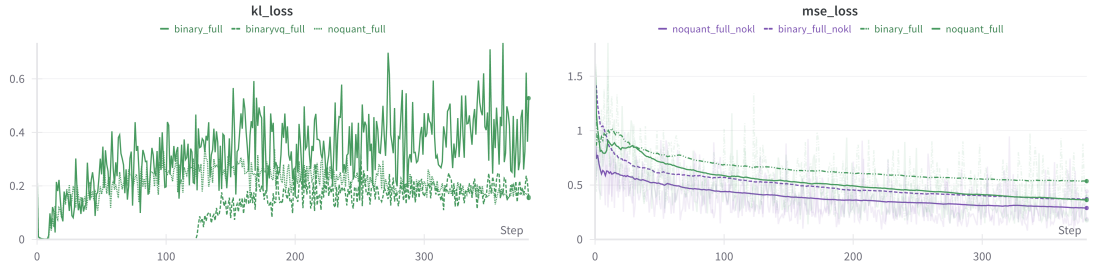


FIGURE 5.3: Left: KL regularization of full-sequence models. Right: Comparison between models with and without KL regularization.

the suggestion of the BLD authors for image generation (Wang et al., 2023), see figure 5.4. This is not surprising, since it is equivalent to predicting the original motion $\mathbf{x}_{1:L}^{(0)}$, which is the aim of the diffusion process in other relevant works, such as MDM (Zhu et al., 2023) or BeLfusion (Barquero, Escalera, and Palmero, 2023). As we can see in figure 5.4, the model struggles to minimize the BCE loss when predicting the flip probability. We can also observe that the VLB regularization term is not able to minimize and bounces back an forth during the training process, depending on the level of noise of the sampled noise $\mathbf{b}_{1:L}^{(t)}$ of the Bernoulli diffusion process. Furthermore, if we take a closer look at the accuracy of the model, we can see that the overall trend is to increase, although also done in a very unstable manner. These results indicate that the model is able to recover the motion for the very first steps of the diffusion process, but encounters challenges when denoising the motion for the inherently noisy steps of the diffusion process.

Despite struggling to achieve complete denoising and resulting in a somewhat shaky movement at inference time, the model successfully recovers the pose of the motion. This implies that pose information encoded in the binary latent space can be effectively reconstructed given a text prompt through a conditioned Bernoulli diffusion process, as depicted in Figure 5.5.

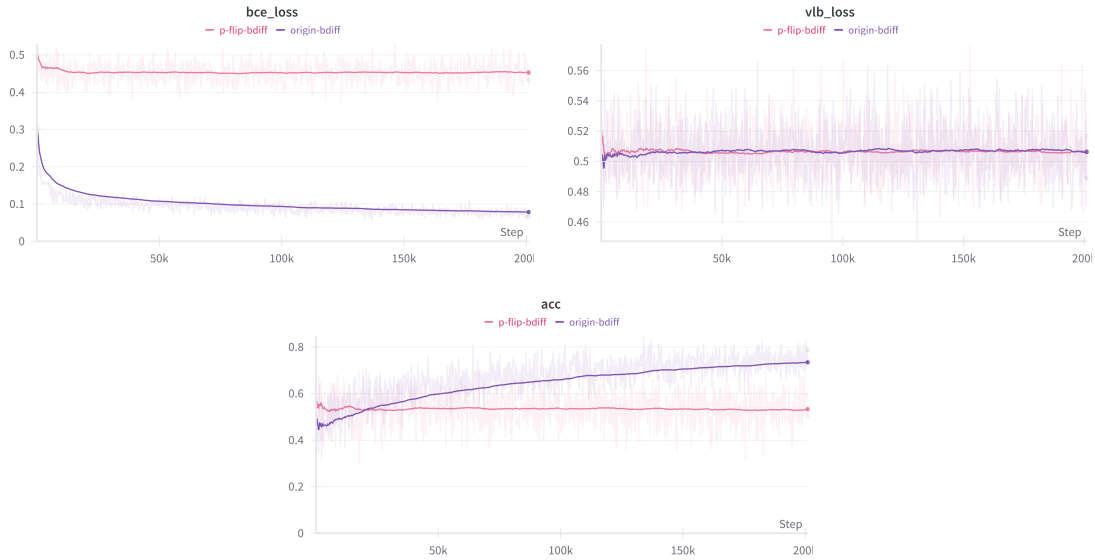


FIGURE 5.4: Comparison between models with different prediction target: **flip probability** and **original pose**.

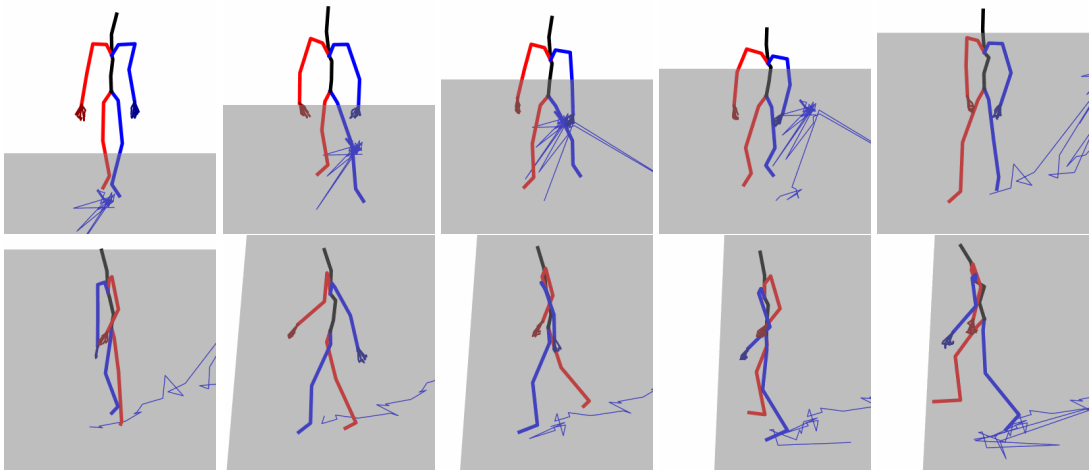

FIGURE 5.5: Sampled motion: "A person walks forward and slightly to the left".

**Experiment 2: Full-sequence diffusion.** The initial experiment led us to the realization that the binary latent space derived from the frame-wise MBVAE lacks sufficient regularization for a denoising model to effectively approximate the reverse process of a Bernoulli diffusion. Consequently, in a subsequent iteration, a diffusion model was trained on the latent space of the full-sequence MBVAE. Despite yielding suboptimal results when compared with the other autoencoders, this model is subject to additional regularization through the inclusion of the KL loss, inherited from MLD (4.1.3).

In this second run, both the diffusion model and the MBLD model undergo training on identical datasets, employing identical hyperparameters and completing the same number of epochs (2000). Notably, both latent spaces maintain a consistent size, constituting a 256-dimensional vector —one continuous (8,192bits) and the other binary (256bits). Evaluation of the joint-level reconstruction loss, as depicted in Figure 5.6, reveals that the MLD model achieves a lower reconstruction loss compared to the Motion Binary Latent Diffusion (MBLD) model. It is noteworthy that the MBLD model exhibits signs of overfitting, as evidenced by the increasing reconstruction loss on the validation set. This indicates that a 256-bit binary latent space rapidly harnesses the information of the training set, and begins to memorize the training data without generalizing to the validation set. This may indicate that the training set is not sufficiently large to achieve a more robust binary latent space.

For a comprehensive comparison of metrics, refer to Table 5.2. Firstly, it is important to note that the MLD model does not achive the good results reported by the authors (Chen et al., 2023b). In order to fit the computational demands, we have reduced the number of layers of the diffusion model, used a smaller version on CLIP, used a smaller subset of HumanML3D, and reduced the number of epochs. However, we are interested in comparing the results of the MLD and MBLD models, which are trained on the same conditions. The FID metric indicates what we have already seen in the plots; the MBLD, despite being able to recover the pose of the motion during training, does not generalize well to the test set increasing higher the FID. The other metrics, MM and DIV, indicate that both models achieve a similar degree of diversity, which is remarable since the MBLD latent is 32 times smaller. This suggests that the binary latent space is able to capture the motion information into a compact yet expressive representation. Furthermore, regarding the top-$k$ accuracies of R-precision, both models behave similarly in terms of conditioning to a text prompt. The reason why there is a large gap between in FID, but not in the other metrics, is due to the fact that the FID, as well as the MSE and L1 losses, are fidelity metrics. Therefore, it is not surprising that if MSE and L1 overfit, the FID will also overfit. However, the other metrics

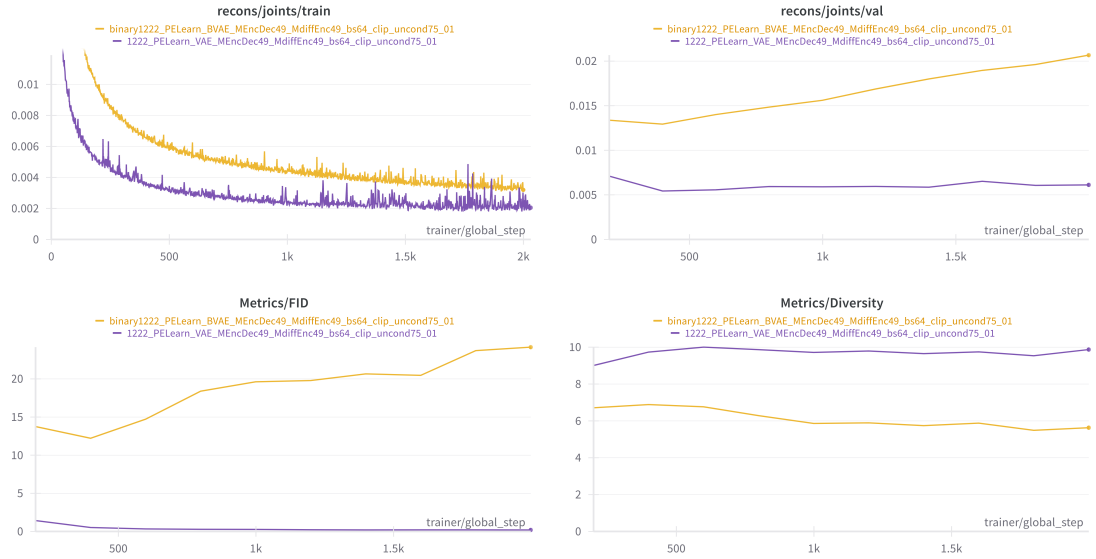


FIGURE 5.6: Reconstruction loss at joint level obtained by the MLD and MBLD models. The MLD model is able to achieve a lower reconstruction loss than the MBLD model.

measure different aspects of the motion, and remain similar. In light of these results, we can conclude that the binary latent space is able to harness the motion information, rapidly overfitting the training set, but not generalizing well to the test set. Hence, the binary latent space may still require further regularization and a larger training set to achieve a more robust representation of motion.

| **Model** | **FID** | **MM** | **DIV** | **Top-1** | **Top-2** | **Top-3** |
|---|---|---|---|---|---|---|
| **GT** | - | - | 9.5 | 0.514 | 0.705 | 0.797 |
| **MLD** | 17.504 | 4.744 | 5.284 | 0.03 | 0.062 | 0.093 |
| **MBLD** | 31.49 | 4.041 | 4.292 | 0.034 | 0.067 | 0.1 |

TABLE 5.2: Metrics obtained by the MLD and MBLD models on the test set.

# 6 Conclusion

In this thesis we have explored the feasibility of representing motion through binary latent spaces. To this end, we first reviewed the theoretical foundations of the Diffusion Models and Variational Autoencoders, gaining insight into the inner working of these models. We then discussed the Binary Latent Diffusion model, which introduces a Bernoulli diffusion process to the binary latent space of a VAE.

Then, we curatioulsy examined the state-of-the-art in text-to-motion generation. We observed that diffusion models have been successfully applied to the task of motion generation, but were surpassed by the more recent Vector Quantized-based models. Such models, however, assume that motion can be represented as a small set of codebook vectors. We claimed that this assumption is not always valid, and that binary latent spaces could be a more suitable representation for motion.

We then proposed the Motion Binary Variational Autoencoder, a VAE that learns a bidirectional mapping between motion sequences and binary latent spaces. We also introduced the Motion Binary Latent Diffusion model, which extends the BLD model to the binary latent space of the MBVAE.

Finally, we conducted experiments to assess the performance of the proposed models. We observed that various iterations of the MBVAE successfully acquire a binary latent space that accurately captures motion dynamics. However, upon introducing noise into the binary latent space, it becomes evident that the MBLD model, while capable of pose recovery with meticulously fine-tuned hyperparameters, struggles with effective motion denoising. The observed erratic movement stems from the diffusion model's limitation in generalizing motion, underscoring the need for additional regularization.

As a prospective course of action, we recommend incorporating an adversarial loss into the latent space, aligning with the approach suggested by the authors of the LDM (Rombach et al., 2021). To further enhance regularization, one could consider computing the reconstruction loss of velocities and joint positions following forward kinematics. Additionally, we propose leveraging the recently released MotionX dataset, renowned for its extensive collection of motion sequences and detailed descriptions. We believe that greater improvements can be achieved by training on this larger dataset, specially after realizing that the model easily overfits the training data. Despite the models being trained on a relatively small dataset and for a limited number of epochs, the outcomes obtained exhibit promise. The results affirm that binary latent spaces, derived through a MBVAE, can effectively represent motion while concurrently posing the alternative of minimizing the memory footprint of the models. Furthermore, the MBLD model's proficiency in pose recovery implies that the binary latent space could be effectively managed by a Bernoulli diffusion model if subjected to further regularization.

# Bibliography


Ahn, Hyemin et al. (2017). *Text2Action: Generative Adversarial Synthesis from Language to Action*. arXiv: 1710.05298 [cs.LG].

Alexanderson, Simon et al. (2023). "Listen, Denoise, Action! Audio-Driven Motion Synthesis with Diffusion Models". In: *ACM Transactions on Graphics* 42.4, 1–20. ISSN: 1557-7368. DOI: 10.1145/3592458. URL: http://dx.doi.org/10.1145/3592458.

Ao, Tenglong, Zeyi Zhang, and Libin Liu (2023). *GestureDiffuCLIP: Gesture Diffusion Model with CLIP Latents*. arXiv: 2303.14613 [cs.CV].

Bank, Dor, Noam Koenigstein, and Raja Giryes (2021). *Autoencoders*. arXiv: 2003.05991 [cs.LG].

Barquero, German, Sergio Escalera, and Cristina Palmero (2023). "BeLFusion: Latent Diffusion for Behavior-Driven Human Motion Prediction". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2317–2327.

Barquero, German et al. (2022). *Didn't see that coming: a survey on non-verbal social human behavior forecasting*. arXiv: 2203.02480 [cs.CV].

Cao, Zhe et al. (2017). *Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields*. arXiv: 1611.08050 [cs.CV].

Chen, Xin et al. (2023a). *Executing your Commands via Motion Diffusion in Latent Space*. arXiv: 2212.04048 [cs.CV].

— (2023b). *Executing your Commands via Motion Diffusion in Latent Space*. arXiv: 2212.04048 [cs.CV].

Dhariwal, Prafulla and Alex Nichol (2021). *Diffusion Models Beat GANs on Image Synthesis*. arXiv: 2105.05233 [cs.LG].

Esser, Patrick, Robin Rombach, and Björn Ommer (2021). *Taming Transformers for High-Resolution Image Synthesis*. arXiv: 2012.09841 [cs.CV].

Fragkiadaki, Katerina et al. (2015). *Recurrent Network Models for Human Dynamics*. arXiv: 1508.00271 [cs.CV].

Gao, Jibin et al. (2022). "PC-Dance: Posture-Controllable Music-Driven Dance Synthesis". In: *Proceedings of the 30th ACM International Conference on Multimedia*. MM '22. Association for Computing Machinery, 1261–1269. ISBN: 9781450392037. DOI: 10.1145/3503161.3548099. URL: https://doi.org/10.1145/3503161.3548099.

Gong, Shansan et al. (2023). *DiffuSeq: Sequence to Sequence Text Generation with Diffusion Models*. arXiv: 2210.08933 [cs.CL].

Goodfellow, Ian J. et al. (2014). *Generative Adversarial Networks*. arXiv: 1406.2661 [stat.ML].

Gulletta, Gianpaolo, Wolfram Erlhagen, and Estela Bicho (2020). "Human-Like Arm Motion Generation: A Review". In: *Robotics* 9.4. ISSN: 2218-6581. DOI: 10.3390/robotics9040102. URL: https://www.mdpi.com/2218-6581/9/4/102.

Guo, Chuan et al. (2022). "Generating Diverse and Natural 3D Human Motions From Text". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5152–5161.

Guo, Zishan et al. (2023). *Evaluating Large Language Models: A Comprehensive Survey*. arXiv: 2310.19736 [cs.CL].

Han, Xiaochuang, Sachin Kumar, and Yulia Tsvetkov (2023). *SSD-LM: Semi-autoregressive Simplex-based Diffusion Language Model for Text Generation and Modular Control*. arXiv: 2210.17432 [cs.CL].

Ho, Jonathan, Ajay Jain, and Pieter Abbeel (2020). *Denoising Diffusion Probabilistic Models*. arXiv: 2006.11239 [cs.LG].

Ho, Jonathan et al. (2022). *Imagen Video: High Definition Video Generation with Diffusion Models*. arXiv: 2210.02303 [cs.CV].

Ionescu, Catalin et al. (2014). "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7, pp. 1325–1339.

Jarzynski, C. (Nov. 1997). "Equilibrium free-energy differences from nonequilibrium measurements: A master-equation approach". In: *Physical Review E* 56.5, 5018–5035. ISSN: 1095-3787. DOI: 10.1103/physreve.56.5018. URL: http://dx.doi.org/10.1103/PhysRevE.56.5018.

Jiang, Biao et al. (2023). *MotionGPT: Human Motion as a Foreign Language*. arXiv: 2306.14795 [cs.CV].

Karras, Tero et al. (2022). *Elucidating the Design Space of Diffusion-Based Generative Models*. arXiv: 2206.00364 [cs.CV].

Kim, Jinwoo et al. (2022). "A Brand New Dance Partner: Music-Conditioned Pluralistic Dancing Controlled by Multiple Dance Genres". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3480–3490. DOI: 10.1109/CVPR52688.2022.00348.

Kingma, Diederik P. and Max Welling (2019). "An Introduction to Variational Autoencoders". In: *Foundations and Trends® in Machine Learning* 12.4, 307–392. ISSN: 1935-8245. DOI: 10.1561/2200000056. URL: http://dx.doi.org/10.1561/2200000056.

Kingma, Diederik P and Max Welling (2022). *Auto-Encoding Variational Bayes*. arXiv: 1312.6114 [stat.ML].

Kirschstein, Tobias, Simon Giebenhain, and Matthias Nießner (2023). *DiffusionAvatars: Deferred Diffusion for High-fidelity 3D Head Avatars*. arXiv: 2311.18635 [cs.CV].

Kucherenko, Taras et al. (July 2019). "Analyzing Input and Output Representations for Speech-Driven Gesture Generation". In: *Proceedings of the 19th ACM International Conference on Intelligent Virtual Agents*. IVA '19. ACM. DOI: 10.1145/3308532.3329472. URL: http://dx.doi.org/10.1145/3308532.3329472.

Li, Chunyuan (2023). *Large Multimodal Models: Notes on CVPR 2023 Tutorial*. arXiv: 2306.14895 [cs.CV].

Liu, Haotian et al. (2023). *Visual Instruction Tuning*. arXiv: 2304.08485 [cs.CV].

Loper, Matthew et al. (2015). "SMPL: A Skinned Multi-Person Linear Model". In: *ACM Trans. Graph.* 34.6. ISSN: 0730-0301. DOI: 10.1145/2816795.2818013. URL: https://doi.org/10.1145/2816795.2818013.

Lucas, Thomas et al. (2022). "PoseGPT: Quantization-based 3D Human Motion Generation and Forecasting". In: *European Conference on Computer Vision (ECCV)*.

Mahmood, Naureen et al. (Oct. 2019). "AMASS: Archive of Motion Capture as Surface Shapes". In: *International Conference on Computer Vision*, pp. 5442–5451.

Martinez, Julieta, Michael J. Black, and Javier Romero (2017). *On human motion prediction using recurrent neural networks*. arXiv: 1705.02445 [cs.CV].

Neal, Radford M. (1998). *Annealed Importance Sampling*. arXiv: physics/9803008 [physics.comp-ph].

Nichol, Alex and Prafulla Dhariwal (2021). *Improved Denoising Diffusion Probabilistic Models*. arXiv: 2102.09672 [cs.LG].

Nichol, Alex et al. (2022). *GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models*. arXiv: 2112.10741 [cs.CV].

Nishimura, Yusuke, Yutaka Nakamura, and Hiroshi Ishiguro (2020). "Long-Term Motion Generation for Interactive Humanoid Robots Using GAN with Convolutional Network". In: *Companion of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*. HRI '20. Cambridge, United Kingdom: Association for Computing Machinery, 375–377. ISBN: 9781450370578. DOI: 10.1145/3371382.3378386. URL: https://doi.org/10.1145/3371382.3378386.

Oord, Aaron van den, Oriol Vinyals, and Koray Kavukcuoglu (2018). *Neural Discrete Representation Learning*. arXiv: 1711.00937 [cs.LG].

Palmero, Cristina et al. (2022). "ChaLearn LAP Challenges on Self-Reported Personality Recognition and Non-Verbal Behavior Forecasting During Social Dyadic Interactions: Dataset, Design, and Results". In: *Understanding Social Behavior in Dyadic and Small Group Interactions*. Ed. by Cristina Palmero et al. Vol. 173. Proceedings of Machine Learning Research. PMLR, pp. 4–52. URL: https://proceedings.mlr.press/v173/palmero22b.html.

Pavlakos, Georgios et al. (2019a). "Expressive Body Capture: 3D Hands, Face, and Body from a Single Image". In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 10975–10985.

— (2019b). *Expressive Body Capture: 3D Hands, Face, and Body from a Single Image*. arXiv: 1904.05866 [cs.CV].

Petrovich, Mathis, Michael J. Black, and Gül Varol (2022). *TEMOS: Generating diverse human motions from textual descriptions*. arXiv: 2204.14109 [cs.CV].

Plappert, Matthias, Christian Mandery, and Tamim Asfour (Dec. 2016). "The KIT Motion-Language Dataset". In: *Big Data* 4.4, 236–252. ISSN: 2167-647X. DOI: 10.1089/big.2016.0028. URL: http://dx.doi.org/10.1089/big.2016.0028.

Podell, Dustin et al. (2023). *SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis*. arXiv: 2307.01952 [cs.CV].

Qi, Chenyang et al. (2023a). *FateZero: Fusing Attentions for Zero-shot Text-based Video Editing*. arXiv: 2303.09535 [cs.CV].

Qi, Qiaosong et al. (Oct. 2023b). "DiffDance: Cascaded Human Motion Diffusion Model for Dance Generation". In: *Proceedings of the 31st ACM International Conference on Multimedia*. MM '23. ACM. DOI: 10.1145/3581783.3612307. URL: http://dx.doi.org/10.1145/3581783.3612307.

Radford, Alec et al. (2021). *Learning Transferable Visual Models From Natural Language Supervision*. arXiv: 2103.00020 [cs.CV].

Razavi, Ali, Aaron van den Oord, and Oriol Vinyals (2019). *Generating Diverse High-Fidelity Images with VQ-VAE-2*. arXiv: 1906.00446 [cs.LG].

Rombach, Robin et al. (2021). *High-Resolution Image Synthesis with Latent Diffusion Models*. arXiv: 2112.10752 [cs.CV].

Sauer, Axel et al. (2023). *Adversarial Diffusion Distillation*. arXiv: 2311.17042 [cs.CV].

Sohl-Dickstein, Jascha et al. (2015). *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*. arXiv: 1503.03585 [cs.LG].

Song, Yang and Stefano Ermon (2020). *Generative Modeling by Estimating Gradients of the Data Distribution*. arXiv: 1907.05600 [cs.LG].

Tevet, Guy et al. (2022). *MotionCLIP: Exposing Human Motion Generation to CLIP Space*. arXiv: 2203.08063 [cs.CV].

Tu, Shuyuan et al. (2023). *MotionEditor: Editing Video Motion via Content-Aware Diffusion*. arXiv: 2311.18830 [cs.CV].

Vaswani, Ashish et al. (2023). *Attention Is All You Need*. arXiv: 1706.03762 [cs.CL].

Wang, Ze et al. (2023). *Binary Latent Diffusion*. arXiv: 2304.04820 [cs.CV].

Xu, Jiale et al. (2023). *Dream3D: Zero-Shot Text-to-3D Synthesis Using 3D Shape Prior and Text-to-Image Diffusion Models*. arXiv: 2212.14704 [cs.CV].

Yang, Ling et al. (2023). *Diffusion Models: A Comprehensive Survey of Methods and Applications*. arXiv: 2209.00796 [cs.LG].

Yang, Ruihan, Prakhar Srivastava, and Stephan Mandt (2022). *Diffusion Probabilistic Modeling for Video Generation*. arXiv: 2203.09481 [cs.CV].

Ye, Hang et al. (2022). *Faster VoxelPose: Real-time 3D Human Pose Estimation by Orthographic Projection*. arXiv: 2207.10955 [cs.CV].

Yin, Tairan et al. (2022). "The One-Man-Crowd: Single User Generation of Crowd Motions Using Virtual Reality". In: *IEEE Transactions on Visualization and Computer Graphics* 28.5, pp. 2245–2255. DOI: 10.1109/TVCG.2022.3150507.

Yoon, Youngwoo et al. (Nov. 2022). "The GENEA Challenge 2022: A large evaluation of data-driven co-speech gesture generation". In: *Proceedings of the 2022 International Conference on Multimodal Interaction*. ICMI '22. ACM. DOI: 10.1145/3536221.3558058. URL: http://dx.doi.org/10.1145/3536221.3558058.

Yu, Jiahui et al. (2022). *Vector-quantized Image Modeling with Improved VQGAN*. arXiv: 2110.04627 [cs.CV].

Yuan, Hongyi et al. (2023). *SeqDiffuSeq: Text Diffusion with Encoder-Decoder Transformers*. arXiv: 2212.10325 [cs.CL].

Zhang, Chenshuang et al. (2023a). *A Survey on Audio Diffusion Models: Text To Speech Synthesis and Enhancement in Generative AI*. arXiv: 2303.13336 [cs.SD].

Zhang, Jianrong et al. (2023b). *T2M-GPT: Generating Human Motion from Textual Descriptions with Discrete Representations*. arXiv: 2301.06052 [cs.CV].

Zhang, Jinsong et al. (2021). *PISE: Person Image Synthesis and Editing with Decoupled GAN*. arXiv: 2103.04023 [cs.CV].

Zhang, Lvmin, Anyi Rao, and Maneesh Agrawala (2023). *Adding Conditional Control to Text-to-Image Diffusion Models*. arXiv: 2302.05543 [cs.CV].

Zhang, Richard et al. (2018). *The Unreasonable Effectiveness of Deep Features as a Perceptual Metric*. arXiv: 1801.03924 [cs.CV].

Zhang, Yan, Michael J. Black, and Siyu Tang (2021). *We are More than Our Joints: Predicting how 3D Bodies Move*. arXiv: 2012.00619 [cs.CV].

Zhang, Zechuan et al. (2023c). *Global-correlated 3D-decoupling Transformer for Clothed Avatar Reconstruction*. arXiv: 2309.13524 [cs.CV].

Zhu, Wentao et al. (2023). *Human Motion Generation: A Survey*. arXiv: 2307.10894 [cs.CV].