Institut für Maschinelle Sprachverarbeitung
Universität Stuttgart
Pfaffenwaldring 5B
D-70569 Stuttgart

Master Thesis

KGGLDM: Knowledge Graph Guided Diffusion Models for Advanced Learning

Akshat Gupta

Studiengang: M.Sc. Computational Linguistics

Prüfer*innen: Prof. Dr. Stefen Staab
Prof. Dr. Thang Vu
Betreuer: Mr. Mojtaba Nayyeri

Beginn der Arbeit: 19.01.2024
Ende der Arbeit: 25.07.2024

Statement of Authorship

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst habe und dabei keine andere als die angegebene Literatur verwendet habe. Alle Zitate und sinngemäßen Entlehnungen sind als solche unter genauer Angabe der Quelle gekennzeichnet. Die eingereichte Arbeit ist weder vollständig noch in wesentlichen Teilen Gegenstand eines anderen Prüfungsverfahrens gewesen. Sie ist weder vollständig noch in Teilen bereits veröffentlicht. Die beigefügte elektronische Version stimmt mit dem Druckexemplar überein.

[1] (Akshat Gupta)

[1] Non-binding translation for convenience: This thesis is the result of my own independent work, and any material from the work of others which is used either verbatim or indirectly in the text is credited to the author including details about the exact source in the text. This work has not been part of any other previous examination, neither completely nor in parts. It has neither completely nor partially been published before. The submitted electronic version is identical to this print version.

Abstract

Diffusion Models (DMs) have achieved state-of-the-art performance in generative tasks across various domains. However, existing DMs lack precise control and struggle with conditional generation and task-specific adaptation. This thesis proposes a novel approach to address these limitations by incorporating Knowledge Graphs (KGs) guidance into Latent Diffusion Model (LDM). KGs provide a rich structured representation of conceptual and relational knowledge within domains. By leveraging Knowledge Graph Embedding (KGE) during the generative process, the proposed model can grasp context, relationships, and attributes encoded in the vector space. This allows for enhanced semantic control, ensuring the generated output exhibits high quality while adhering to the specified conceptual constraints from the KGs. The integration of KGs with DMs represents an unexplored direction with the potential to significantly improve the precision, controllability, and domain specificity of generative models.

This thesis explores the mathematical foundations and empirical evaluations of this innovative knowledge-guided diffusion modeling framework, demonstrating its effectiveness in enhancing the quality, diversity, and semantic consistency of generated images, particularly in capturing complex attribute combinations and relationships. Our proposed approach, Knowledge Graph Guided Latent Diffusion Model (KGGLDM) achieves superior performance with the lowest Fréchet Inception Distance (FID) score of **16.5** and high attribute consistency (**55.1**% accuracy across **40** features, **99.9**% for prominent single attributes). Human evaluations confirms these results, with KGGLDM receiving top scores for visual quality (**4.5=5**) and semantic consistency (**4.4=5**).

Acknowledgements

First and foremost, I am deeply thankful to my supervisor Mr. Mojtaba Nayyeri for his invaluable guidance, support, and encouragement throughout this research journey. His expertise and insights have been instrumental in shaping the direction of this work. I am also grateful to the members of my thesis committee for their valuable feedback and suggestions.

I would like to extend my appreciation to my colleagues and friends for their encouragement and assistance. Their support has been a constant source of motivation during challenging times. I am also indebted to the staff and faculty members of the department for their assistance and resources.

Furthermore, I would like to thank my family for their unwavering support and understanding throughout this endeavor. Their love, encouragement, and belief in my abilities have been a driving force behind my academic pursuits.

This work was performed on the HoreKa supercomputer funded by the Ministry of Science, Research and the Arts Baden-Württemberg and by the Federal Ministry of Education and Research, Germany.

Lastly, I am grateful to all the researchers whose work has inspired and influenced this thesis. Their contributions to the field have provided a solid foundation for my research.

Inhaltsverzeichnis

1 Introduction 13

2 Background 17

2.1 Knowledge Graphs 17

2.2 Knowledge Graph Embeddings 18

2.3 Di usion Models 19

2.4 Knowledge Guided Generative Models 21

2.5 Motivation 21

2.6 Research Objectives 22

3 Literature Review 25

3.1 Ontology Creation 25

3.2 Knowledge Graph Representation 27

3.3 Knowledge Graph Embeddings 29

3.4 Generative Modeling 30

3.4.1 Autoencoder 31

3.4.2 Variational Autoencoder 32

3.4.3 Vector Quantized VAEs 33

3.4.4 Generative Adversarial Network 35

3.4.5 Limitations of Autoencoders, Variational Autoencoders, and Generative Adversatial Networks in Fine-Grained Control over Generation 36

3.5 Di usion Models 38

3.5.1 History of Di usion Models 38

3.5.2 Di usion Model Architecture 40
3.6 Knowledge-Guided Generative Models 43
3.7 LLaVA Model 43
3.7.1 Architecture and Training 44
3.7.2 Capabilities and Applications 45
3.7.3 Relevance to Knowledge-Guided Generative Models 45

4 Methodology 47
4.1 Ontology Creation 47
4.1.1 Concept Identi cation 47
4.1.2 Attribute Assignment 47
4.1.3 Relationship Establishment 48
4.1.4 Ontology Formalization 48
4.1.5 Ontology Validation 49
4.2 Knowledge Graph Construction 50
4.2.1 Entity Mapping 50
4.2.2 Attribute Representation 50
4.2.3 Relationship Representation 51
4.2.4 Triple Generation 51
4.3 Knowledge Graph Embedding Creation 51
4.3.1 Data Preparation 52
4.3.2 PyKEEN Con guration 52
4.3.3 Model Training 52
4.3.4 Embedding Evaluation 52
4.3.5 Embedding Extraction 54

4.3.6 Encoding Multiple Features Embeddings 55
4.4 Knowledge Graph Embedding Integration into Di usion Model 56
4.4.1 Embedding Conditioning 57
4.4.2 Training and Inference 57

5 Experiments 59
5.1 Dataset 59
5.2 Experimental Setup 60
5.2.1 Model Con gurations 60
5.2.2 Training Settings 63
5.3 Evaluation Strategy 65
5.4 Baselines 67
5.5 Hardware and Software Speci cations 68

6 Results 69
6.1 Knowledge Graph Statistics 69
6.2 Comparison of Knowledge Graph Embedding Methods 69
6.3 Latent Space Analysis of Knowledge Graph Embeddings 71
6.3.1 Autoencoder for Dimensionality Reduction 71
6.4 UMAP Visualization of CelebA Dataset Features 72
6.5 Quantitative Results 76
6.6 Human Evaluation Score 77
6.7 Comparison of Knowledge Graph Guided Di usion Model and Stable Di usion 77
6.7.1 Attribute Accuracy across features 77
6.7.2 Detailed Attribute Analysis 78

7 Discussion 83
7.1 Interpretation of Results 83
7.1.1 Quantitative Performance 83
7.1.2 Qualitative Assessment 83
7.1.3 Attribute-Speci c Performance 84
7.2 Comparison with Existing Literature 84
7.3 Implications and Potential Applications 85

8 Conclusion 87
8.1 Summary of Contributions 87
8.2 Implications and Future Directions 88
8.3 Future Research Directions 88

Abbildungsverzeichnis

1 Overview of knowledge graph guided di usion model 14

2 Sample output of KGGLDM 16

3 Example of an ontology structure showing classes, hierarchies, and relationships with Knowledge Graph 19

4 Example of an ontology structure showing classes, hierarchies, and relationships with Knowledge Graph 20

5 Overview of di usion process (35) 20

6 Knowledge Graph Representation (1) 27

7 Visualization of the Di usion Process: Forward Noise Addition and Reverse Denoising 38

8 Cosine Schedule for Noise Scales 41

9 Training (left) and Sampling (right) Algorithms for Di usion Models (35) 42

10 Classi er guided di usion model 44

11 Visual representation of the facial ontology 49

12 Unet Architecture 61

13 KGGLDM architecture 63

14 Uniform Manifold Approximation and Projection (UMAP) visualizations for gender attribute 73

15 UMAP visualizations for age attribute 73

16 UMAP visualizations for hair color 74

17 UMAP visualizations for smiling 75

18 UMAP visualizations for eye glasses 75

19 Comparison for Female with Black Hair and Old, left is Stable Di usion (SD) and on right is KGGLDM 79

20 Comparison for Female Old and Bald, left is SD and on right is KGGLDM 79

21 Comparison for Male and Old, left is SD and on right is KGGLDM . 80

22 Comparison for Male and Young, left is SD and on right is KGGLDM 81

23 Comparison for Male Old with Blonde Hair, left is SD and on right is KGGLDM 81

Tabellenverzeichnis

1 Sample feature data for CelebA dataset with 5 features. 61

2 Statistics of the Knowledge Graph 69

3 Performance comparison of knowledge graph embedding methods . . 70

4 Quantitative Evaluation Metrics for Baseline Models 76

5 Quantitative Evaluation Metrics for Our KGGLDM Approach 76

6 Human Evaluation Scores 77

7 Attribute Accuracy Comparison 78

List of Abbreviations

AI Arti cial Intelligence

AE Autoencoder

CelebA CelebFaces Attributes

ComplEx Complex Embeddings

ConvE Convolutional Embeddings

ConvKB Convolutional Knowledge Base Embeddings

CPU Central Processing Unit

DDPM Denoising Di usion Probabilistic Model

DM Di usion Model

DNN Deep Neural Network

ELBO Evidence Lower Bound

FID Fréchet Inception Distance

GAN Generative Adversarial Network

GCN Graph Convolutional Network

GPU Graphics Processing Unit

IS Inception Score

KG Knowledge Graph

KGE Knowledge Graph Embedding

KGGLDM Knowledge Graph Guided Latent Di usion Model

LDM Latent Di usion Model

LLM Large Language Model

LLaVA Large Language and Vision Assistant

ML Machine Learning

MRR Mean Reciprocal Rank

MSE Mean Squared Error

ODP Ontology Design Patterns

OWL Web Ontology Language

RDF Resource Description Framework

RESCAL Relational Scoring for Collaborative Filtering

RGB Red, Green, Blue

SPARQL SPARQL Protocol and RDF Query Language

SD Stable Di usion

SSIM Structural Similarity Index Measure

TransD Translation on Dynamic Mapping

TransE Translating Embeddings

TransH Translating on Hyperplanes

TransR Translation on a Relation-speci c Space

UMAP Uniform Manifold Approximation and Projection

VAE Variational Autoencoder

VQ-VAE Vector Quantized Variational Autoencoder

WGAN Wasserstein Generative Adversarial Network

1 Introduction

DMs (35) have emerged as a promising class of deep generative models (76) with state-of-the-art sample quality and high delity across image, audio, and video domains. The di usion process involves adding noise to the data and then recovering the original data by conditioning them on noisy data. While simple di usion models (35) can produce high-quality outputs, they often su er from limitations such as challenges in conditional image generation (5), controllability issues (97), lack of speci city, and di culty in task-speci c adaptation (23) (35).

To address these limitations, guided di usion models (21) have been proposed. Guidance enables the model to generate images with conditional control over attributes such as gender, hair color, smile presence, and various facial features. However, existing approaches to guidance in di usion models remain limited in terms of their precision and semantic manipulation capabilities, hindering their potential for generating highly controllable and contextually accurate outputs.

KGs (41) have excelled in systematically encoding conceptual and complex relational knowledge within domains, thereby serving as rich semantic representations (66). By structuring facts and semantics from data as interconnected entities and relations, knowledge graphs provide an intuitive framework suitable for enhancing machine-learning pipelines. The primary goal is to discover and represent knowledge in a structured format, thereby facilitating utility across various applications (41).

This thesis explores a novel approach by bridging the gap of di usion modeling and knowledge graphs, unveiling a potentially groundbreaking direction that serves as the central theme of this work. We propose incorporating knowledge graph guidance into LDM (73) models to augment precise control over sample generation using domain conceptual knowledge. The main research questions that this thesis aims to address are:

1. How can a KGs be e ectively constructed and encoded in vector space to guide the di usion model?

2. How can KGs e ectively guide LDM to improve control over generation?

Abbildung 1: Overview of knowledge graph guided di usion model

3. Does knowledge graph-enhanced guidance better capture contextual delity compared to existing guidance approaches?

4. How can Large Language and Vision Assistant (LLaVA) models be leveraged to evaluate the attribute accuracy of images generated by di usion models?

To tackle these research questions, we propose a method that utilizes KGE to guide the data generation process in LDM. During the generation process, the model leverages the guidance from KGE to grasp the context, relationships, and attributes of the entities encoded in the vector space. This approach ensures that the generated output is not only of high quality but also exhibits semantic richness, contextual accuracy, and relevance.

Figure 1 illustrates the overall architecture of our proposed approach, KGGLDM. The KGs block, represented on the left, encodes structured information about facial attributes and their relationships. This information is transformed into embeddings, compressed into latent space using autoencoders and then used to condition the LDM during the image generation process. The di usion model, depicted on the upper right block, progressively re nes noise into a high-quality image, guided at each step by the KGE.

The main contributions of this thesis are fourfold. First, we present a novel framework incorporating KGs guidance into LDM, enabling precise control over sample generation using domain conceptual knowledge. Second, we employ Autoencoder (AE)

for embedding vector generation. Third, to demonstrate the e ectiveness of our approach, we utilize the CelebFaces Attributes (CelebA) dataset, a large-scale face attributes dataset with more than 200,000 celebrity images, each with 40 attribute annotations. This dataset is particularly suitable for our research as it provides a rich set of structured attributes that can be naturally represented in a KGs. The diversity of facial features and attributes in CelebA allows us to showcase the power of KGs guidance in controlling various aspects of image generation, from hair color and style to facial expressions and accessories. Fourth, we introduce a novel evaluation methodology using LLaVA models to assess the attribute accuracy of images generated by di usion models, providing a more comprehensive evaluation of the generated outputs.

Figure 2 presents a sample grid of images generated by our KGGLDM. These results visually demonstrate the model's capability to generate diverse facial images while adhering to speci ed attributes, showcasing the e ectiveness of our knowledge-guided approach.

Abbildung 2: Sample output of KGGLDM

2 Background

KGs and DMs represent two signi cant advancements in Arti cial Intelligence (AI), each addressing distinct aspects of knowledge representation and generation. Knowledge graphs, emerging from the eld of knowledge representation, provide a structured framework for encoding entities and their relationships in a graph-based format (37). These graphs capture semantic information and domain-speci c knowledge, enabling complex reasoning and inference tasks. Recent years have seen a surge in the application of KGs across various domains, from natural language processing to recommendation systems (41).

Concurrently, DMs have revolutionized the landscape of generative modeling. Introduced by (79) and further developed by (35), DMs learn to reverse a gradual noising process, generating high quality samples by iteratively denoising random noise. These models have demonstrated remarkable performance in image synthesis, audio generation, and other generation tasks, often surpassing previous state-of-the-art methods such as Generative Adversarial Network (GAN) and Variational Autoencoder (VAE) (21). The success of DMs lies in their stable training dynamics and the ability to capture complex data distributions.

However, while KGs excel at representing structured information and DMs at generating diverse content, the integration of these two paradigms remains largely unexplored. This research aims to bridge this gap, using the semantic richness of KGs to enhance DMss controllability and contextual awareness.

2.1 Knowledge Graphs

KGss (19) are gaining attention because of their semantic representation, which enables a nuanced understanding of entity relationships (58). A KGs is made up of entities and relationships. Entities, which constitute the nodes of these graphs, encompass a broad spectrum of objects and concepts ranging from tangible entities such as individuals and organizations to more abstract notions. The relationships,

depicted as edges, establish the interconnections between these entities and structure the graph into a meaningful representation of real-world interactions.

Figure 3 illustrates the structure of a KGs, showcasing both the ontology-view and instance-view layers. The ontology layer (top) de nes conceptual classes and their relationships, while the instance layer (bottom) represents speci c entities and their connections. This hierarchical organization allows for e cient representation of complex, real-world information systems, demonstrating how abstract concepts relate to concrete instances.

The essence of knowledge graphs is their ability to incorporate semantic depth within these relationships, surpassing binary linkages to represent contextually rich connections. Such semantic relationships (30; 49; 67), characterized by labels like „has\, „looks like\ or „owns\, enrich the graph with a level of complexity that mirrors the intricacies of real-world data interrelations. KGs integrate heterogeneous data sources into a uni ed and comprehensive schema that is essential to obtain insightful analyses.

2.2 Knowledge Graph Embeddings

KGE (32; 81; 96) transforms the discrete entities and relation names of a KGs into continuous vector spaces with the aim of preserving the semantics and structural information. As illustrated in Figure 4, this process maps the input KGs to a low-dimensional vector space representation, where entities and relations are represented as points in this space. In a typical KGs, $KG = (E; R; T)$, where $E; R$ are the sets of entities and relation names, and T is the set of relational triples $(e_i; r_j; e_k)$, the goal is to learn an embedding function f that maps entities and relation names to vectors $(e_i; r_j; e_k)$, so that the relationships are encoded in how the vectors interact mathematically, e.g., $e_i + r_j \quad e_k$.

Di erent techniques such as Translating Embeddings (TransE), neural models (92), and tensor factorization (44) de ne the embedding function and scoring criteria di erently. The right side of Figure 4 demonstrates how these embeddings result in a structured arrangement of entities in the vector space, preserving their relationships

Abbildung 3: Example of an ontology structure showing classes, hierarchies, and relationships with Knowledge Graph

from the original KGs. Such embeddings enable KGs to be leveraged for Machine Learning (ML) applications such as link prediction, recommendation systems, and claim veracity assessment in a computationally e cient manner.

2.3 Di usion Models

The Denoising Di usion Probabilistic Model (DDPM) (35; 79) has emerged as an advancement in generative modeling, particularly for its e cacy in high-quality image synthesis. This model is based on the gradual addition of noise to the data and then learning to reverse this di usion process to generate new samples from scratch, as illustrated in Figure 5.

The forward di usion process incrementally adds Gaussian noise to the data following a Markov chain, represented by the arrows moving from x_T to x_0 in the gure. In the reverse process, a Deep Neural Network (DNN) is trained to predict the noise that was added at each step conditioned on current noisy data, gradually

Abbildung 4: Example of an ontology structure showing classes, hierarchies, and relationships with Knowledge Graph

Abbildung 5: Overview of di usion process (35)

denoising and reconstructing the original data distribution. This is depicted by the red dashed arrow in the image, where $q(x_t|x_{t-1})$ is unknown and estimated using a variational lower bound.

DDPM's ability to produce diverse, high- delity samples without common issues like mode collapse makes it an appealing generative model. However, the stochastic nature of the process poses challenges in controlling and conditioning the generated outputs precisely. The image showcases this process, starting from a noisy input x_T and progressively re ning it to generate a clear face image x_0.

2.4 Knowledge Guided Generative Models

Knowledge-guided generative models represent an emerging paradigm that integrates structured domain knowledge, typically encoded in KGss, into the generative process of deep learning models. Using the rich semantic relationships and conceptual information captured by KGss, these models aim to improve the coherence, contextual relevance and controllability of the generated output. The incorporation of KGss has been explored in various generative frameworks, such as GANs, VAEs, and language models. In the context of GANs, KGss have been employed to guide the image generation process towards speci c semantic concepts and relationships, enabling the creation of visually coherent and semantically meaningful images. Similarly, knowledge-guided language models have demonstrated improved factual accuracy and contextual consistency in the generated text by aligning the generation process with the domain knowledge encoded in the graphs. However, the integration of KGss with more advanced generative models, such as DMss, remains a relatively underexplored area.

2.5 Motivation

Di usion models have shown impressive capabilities in generating high-quality synthetic data across various domains. However, they often lack ne-grained control and struggle to incorporate semantic knowledge e ectively. This thesis proposes KGGLDM (Knowledge Graph Guided Latent Di usion Model), a novel approach that integrates KGs embeddings into the latent space of di usion models. The key points are:

1. Current di usion models lack ne-grained control and semantic knowledge incorporation.
2. KGGLDM aims to bridge the gap between di usion models and structured knowledge representation.

3. It leverages KGs to guide the generation process towards more coherent, interpretable, and controllable outputs.

4. The approach enables speci cation of complex attributes and relationships in generated data.

5. Potential applications include data augmentation, virtual avatar creation, and criminal investigation.

6. The research contributes to advancing knowledge-guided generation in generative models.

This approach has the potential to signi cantly advance the eld of generative models by incorporating structured domain knowledge, leading to more semantically grounded and controllable outputs.

2.6 Research Objectives

This research endeavors to investigate and develop innovative methodologies for the integration of knowledge graphs into di usion models, thereby facilitating knowledge-guided data generation. The primary objectives of this study are as follows:

1. To explore and develop methods for representing and encoding KGs in a format conducive to integration with LDMs.

    - Investigate diverse KGs embedding techniques, including but not limited to TransE, DistMult, and Complex Embeddings (ComplEx), for the transformation of entities and relationships into continuous vector representations.
    - Evaluate the qualitative aspects and expressiveness of KGs embeddings in capturing semantic relationships and hierarchical data structures.

2. To develop novel architectures and techniques for conditioning the LDM's generation process on KGs embeddings, while maintaining the model's capacity to generate high- delity and diverse samples.
    - Design and implement the KGGLDM by integrating KGs embeddings into the latent space of the di usion model.
    - Ensure that the integration of KGs embeddings does not compromise the quality and diversity of the generated samples.
3. To propose evaluation metrics and strategies for assessing the quality, coherence, and KGs adherence of the generated data.
    - Develop quantitative metrics to measure the visual quality, diversity, and realism of the generated samples, such as Inception Score (IS) or FID.
    - Introduce novel evaluation metrics to assess the coherence and consistency of the generated data with respect to the KGs, capturing adherence to speci ed attributes and relationships.
    - Conduct qualitative evaluations, including human perceptual studies and expert assessments, to gather subjective feedback on the quality and coherence of the generated samples.
4. To demonstrate the e cacy of KGGLDM through comprehensive experiments and evaluations on diverse datasets and generation tasks.
    - Conduct experiments on the CelebA dataset to generate facial images with controlled attributes and relationships based on the KGs.
    - Evaluate the performance of KGGLDM in comparison to baseline models, including class-guided di usion models and unconditional di usion models.
    - Analyze the impact of various KGs embedding techniques and conditioning mechanisms on generation quality and KGs adherence.

The integration of KGs with di usion models has the potential to signi cantly impact the eld of generative modeling. This approach may lead to more controllable and interpretable generative models, enabling applications in diverse domains such as personalized content creation, data augmentation for machine learning, and advanced image editing tools. Furthermore, it could pave the way for generative models that can leverage domain-speci c knowledge in elds such as healthcare, robotics, and scienti c discovery, leading to more accurate and context-aware data generation.

This research aims to advance knowledge-guided generation, paving the way for semantically grounded and controllable generative models. The remainder of this thesis is structured as follows:

- Chapter 2 provides a background, introduction to KGs, and di usion models.
- Chapter 3 provides a comprehensive literature review covering KGs, di usion models, and related work in knowledge-guided generation.
- Chapter 4 details our methodology, including ontology creation, KGs construction, and the integration of KGs embeddings into di usion models.
- Chapter 5 describes our experimental setup using the CelebA dataset.
- Chapter 6 presents and analyzes our results, while Chapter 7 discusses the implications of our ndings.
- Chapter 8 concludes the thesis and outlines directions for future research.

3 Literature Review

In this literature review, we explore the key research areas that lay the foundation for Knowledge Graph Guided Latent Di usion Models (KGGLDM). We begin by knowledge graphs, discussing ontology creation, KGs representation, and the application of KGs embeddings in various domains. Next, we provide an overview of generative modeling techniques, including AE, VAE, vector-quantized VAEs, and GAN. We highlight the strengths and limitations of these approaches in the context of generating high-quality and diverse data. Subsequently, we focus on di usion models, tracing their historical development, architectural design, training strategies, and applications. Finally, we investigate the intersection of KGs and generative models, examining conditional generative models, knowledge-aware language models, and knowledge-guided image generation. By synthesizing the relevant literature across these domains, we aim to establish a comprehensive understanding of the current state-of-the-art and identify the gaps that motivate the development of KGGLDM.

3.1 Ontology Creation

Ontologies play a fundamental role in the creation and organization of knowledge graphs, providing a formal speci cation of the concepts, relationships, and constraints within a domain. An ontology serves as a shared vocabulary and a conceptual framework that enables consistent interpretation, reasoning, and integration of knowledge across di erent systems and applications (29).

The process of ontology creation involves several key steps, including domain analysis, conceptualization, formalization, and implementation. Domain analysis involves understanding the scope and requirements of the domain, identifying the relevant concepts, and gathering domain knowledge from experts and existing sources. Conceptualization focuses on structuring the domain knowledge into a coherent model, de ning classes, properties, and relationships. Formalization involves expressing the conceptual model using a formal ontology language, such as Web Ontology Language (OWL). OWL (55) provides a set of constructs for de ning classes, properties,

and axioms. Finally, implementation involves encoding the ontology using a suitable representation format and integrating it into the knowledge graph.

Ontology development methodologies have been proposed to guide the process of creating high-quality ontologies. One widely adopted methodology is Ontology Development 101(63), which provides a step-by-step guide for building ontologies, including de ning classes, arranging them in a taxonomic hierarchy, de ning properties and relationships, and specifying constraints (see Figure 3). This structured approach helps ensure consistency and completeness in ontology design.

The choice of ontology language is crucial in ontology creation, as it determines the expressiveness and reasoning capabilities of the resulting ontology. OWL has emerged as the de facto standard for representing ontologies on the Web, o ering three sublanguages with increasing levels of expressiveness: OWL Lite, OWL DL, and OWL Full (55). Each sublanguage provides a di erent trade-o between expressive power and computational complexity, allowing ontology designers to choose the appropriate level based on their requirements. Ontology reuse and integration are important considerations in ontology creation, as they promote interoperability and reduce the e ort required to build new ontologies from scratch. Ontology repositories, such as BioPortal (91) and Ontology Design Patterns (Ontology Design Patterns (ODP)) (24), provide access to a wide range of existing ontologies that can be reused or extended for speci c domains. Ontology alignment and mapping techniques (65) enable the integration of multiple ontologies, allowing for the creation of more comprehensive and interconnected knowledge graphs.

In the context of Knowledge Graph Guided Latent Di usion Models (KGGLDM), ontologies serve as a structured and semantically rich foundation for representing domain knowledge. By leveraging ontologies, KGGLDM can incorporate domain-speci c concepts, relationships, and constraints into the generative process, enabling the production of coherent and meaningful data. The choice of ontology and its integration with the knowledge graph have a signi cant impact on the quality and interpretability of the generated data, making ontology creation a critical aspect of knowledge-guided generative models.

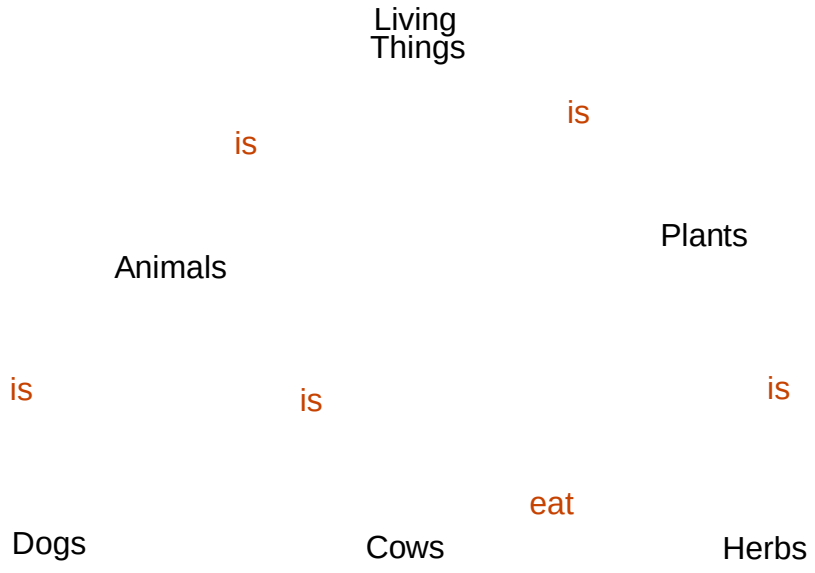

Abbildung 6: Knowledge Graph Representation (1)

3.2 Knowledge Graph Representation

KGs representation is a fundamental aspect of working with KGs, as it determines how the information encoded in the graph is structured and stored for e cient processing and reasoning (41). The choice of representation directly impacts the ease of querying, manipulating, and integrating KGs with other systems (25). Figure 6 illustrates a typical KGs structure, showcasing entities as nodes and relationships as edges.

One of the most common representations for KGs is the Resource Description Framework (Resource Description Framework (RDF)) (45). RDF is a standard model for representing information on the web, using a triple-based structure consisting of a subject, predicate, and object. Each triple represents a fact or relationship between entities, where the subject and object are nodes in the graph, and the predicate is the labelled edge connecting them. RDF provides a exible and extensible framework for modelling complex domain knowledge and supports reasoning and

inference through the use of ontologies and rule-based systems (13).

Another popular representation for KGs is the property graph model (72). In a property graph, entities are represented as nodes, and relationships between entities are represented as edges. Both nodes and edges can have associated properties or attributes, providing additional contextual information. Property graphs o er a more intuitive and expressive representation compared to RDF, as they allow for the direct attachment of metadata to graph elements. This representation is widely used in graph databases and has gained traction in various applications, such as social network analysis and recommendation systems (6).

The choice between RDF and property graph representations often depends on the speci c requirements of the application and the ecosystem in which the KGs operates. RDF has the advantage of being a well-established standard with a rich set of tools and libraries for querying (e.g., SPARQL Protocol and RDF Query Language (SPARQL)) and reasoning. On the other hand, property graphs provide a more natural and e cient representation for certain use cases and have a growing ecosystem of graph databases and processing frameworks (31).

The choice of KGs representation and the use of ontologies have signi cant implications for the scalability, interoperability, and usability of knowledge-based systems. Researchers have explored various techniques for e cient storage, querying, and reasoning over large-scale KGs (2). These include graph partitioning, indexing, and compression techniques, as well as distributed and parallel processing approaches (39).

In the context of Knowledge Graph Guided Latent Di usion Models (KGGLDM), the representation of the KGs plays a vital role in how the structured knowledge is integrated into the generative process. The choice of representation impacts the ease of extracting relevant information, aligning entities and relationships with the latent space of the di usion model, and incorporating domain-speci c constraints and reasoning capabilities. By carefully considering the representation and leveraging the power of ontologies, KGGLDM can e ectively harness the rich semantic information present in KGs to guide the generation of coherent and meaningful

data.

Formally, a KGs can be de ned as a directed labeled graph $G = (E; L; R)$, where E is the set of entities, L is the set of relation types or predicates, and $R \subseteq E \times L \times E$ is the set of relationships or triples. Each triple $(s; r; o)$ represents a relationship r between a subject entity s and an object entity o.

3.3 Knowledge Graph Embeddings

KGE have emerged as a powerful technique for representing the entities and relationships in a knowledge graph within a continuous vector space (89). The objective of KGE is to learn low-dimensional vector representations that capture the semantic and structural information of the graph, enabling e cient storage, computation, and reasoning (15).

Various embedding techniques have been proposed in the literature, each with its unique approach to modeling the interactions between entities and relations. One of the earliest and most well-known methods is TransE (11), which interprets relationships as translations in the embedding space. In TransE, the embedding of a tail entity is obtained by adding the relation embedding to the head entity embedding. This simple yet e ective approach has inspired numerous extensions and improvements, such as Translating on Hyperplanes (TransH), Translation on a Relation-speci c Space (TransR) (47), and Translation on Dynamic Mapping (TransD) (40), which introduce additional complexity to capture more intricate relational patterns.

Another family of knowledge graph embedding techniques is based on semantic matching, where the plausibility of a triple is determined by the similarity between the embeddings of the head entity, relation, and tail entity. Methods like Relational Scoring for Collaborative Filtering (RESCAL) (62), DistMult (94), and ComplEx (82) uses tensor factorization or multiplicative interactions to compute the score of a triple. These approaches have shown e ectiveness in capturing symmetric and anti-symmetric relations and have been extended to incorporate additional information, such as entity types and relation paths.

More recently, neural network-based approaches have gained prominence in KGE. Techniques like Convolutional Embeddings (ConvE) and Convolutional Knowledge Base Embeddings (ConvKB) (59) utilize convolutional neural networks to learn expressive embeddings by extracting local and global patterns from the graph structure. Graph Convolutional Network (GCN)s (77) have also been applied to knowledge graph embedding, leveraging the graph's connectivity information to enrich the entity representations. The choice of KGE technique depends on various factors, such as the characteristics of the knowledge graph, the desired properties of the embeddings, and the downstream application requirements. Comprehensive surveys and empirical studies (75; 89) have been conducted to compare and evaluate di erent embedding methods across various benchmarks and tasks.

KGE have found extensive applications in tasks such as link prediction, entity alignment, and recommender systems. By providing a compact and informative representation of the KGs, embeddings enable e cient computation and integration with ML models, opening up new possibilities for knowledge-informed AI systems.

In the context of KGGLDM, KGE play a crucial role in incorporating structured knowledge into the generative process. By leveraging the semantic and relational information captured by the embeddings, KGGLDM aims to enhance the coherence, interpretability, and controllability of the generated data, pushing the boundaries of knowledge-guided generative models.

3.4 Generative Modeling

Generative modeling is a rapidly evolving eld in ML that focuses on building models capable of generating new data samples that resemble the training data distribution. These models have gained signi cant attention due to their ability to capture complex data distributions and generate realistic examples in various domains, such as images, text, and audio. Generative models can be broadly categorized into explicit density models, such as PixelRNN (64) and PixelCNN (84), which explicitly de ne the probability distribution over the data, and implicit density models, such as Generative Adversarial Networks (GANs) (27) and Variational Autoencoders (VAEs)

(43), which learn to generate samples without explicitly de ning the probability distribution. These models have been successfully applied in a wide range of tasks, including image synthesis (14), text generation (68), and data augmentation (7). Despite their success, generative models still face challenges such as mode collapse, training instability, and di culty in evaluating the quality of generated samples. Ongoing research focuses on addressing these challenges and further improving the performance and applicability of generative models.

3.4.1 Autoencoder

An AE is a type of neural network that learns to compress and reconstruct input data, typically for unsupervised representation learning or dimensionality reduction. The basic architecture of an AE consists of an encoder function $f(x)$ that maps the input data x to a lower-dimensional latent representation z, and a decoder function $g(z)$ that reconstructs the input data from the latent representation (26).

Mathematically, given an input $x \in R^d$, the encoder function $f : R^d \to R^p$ maps the input to a latent representation $z \in R^p$, where $p < d$. The encoder function can be parameterized by a neural network with weights, such that $z = f(x)$. The decoder function $g : R^p \to R^d$ maps the latent representation back to the original input space, producing a reconstructed input $\hat{x} = g(z)$, where represents the weights of the decoder network.

The AE is trained to minimize the reconstruction loss between the input x and the reconstructed input $\hat{x}$. A common choice for the reconstruction loss is the Mean Squared Error (MSE):

$$L(x; \hat{x}) = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{x}_i)^2 \tag{1}$$

where n is the number of input samples. The optimization objective is to nd the encoder and decoder parameters and that minimize the reconstruction loss:

$$; = \arg\min_{;} L(x; g(f(x))) \tag{2}$$

During training, the AE learns to compress the input data into a lower-dimensional latent space while preserving the essential information needed for reconstruction. The latent representation learned by the AE can capture meaningful features and structures in the data, making it useful for various downstream tasks such as clustering (93), anomaly detection (98), and data generation (54).

Variants of AE have been proposed to improve their performance and capabilities. Denoising AE (87) are trained to reconstruct clean input data from corrupted versions, enhancing their robustness and generalization. VAEs (43) introduce a probabilistic framework by modeling the latent space as a distribution, enabling the generation of new samples from the learned distribution. Adversarial AE (54) combine the reconstruction objective with an adversarial loss to encourage the latent space to follow a desired prior distribution.

3.4.2 Variational Autoencoder

VAEs (43; 71) are a class of generative models that extend the traditional AE framework by introducing a probabilistic interpretation of the latent space. VAEs aim to learn a generative model of the data distribution by modelling the joint distribution of the input data and the latent variables. In a VAE, the encoder network parameterizes an approximate posterior distribution $q_{\phi}(z|x)$ over the latent variables z given the input data x, while the decoder network parameterizes the likelihood distribution $p_{\theta}(x|z)$ of the input data given the latent variables. The objective of a VAE is to maximize the Evidence Lower Bound (ELBO) (10), which is a lower bound on the log-likelihood of the data:

$$\log p(x) \quad L(\theta; \phi; x) = E_{q_{\phi}(z|x)}[\log p_{\theta}(x|z)] \quad D_{KL}(q_{\phi}(z|x) || p(z)) \tag{3}$$

where D_{KL} denotes the Kullback-Leibler divergence between the approximate posterior $q_{\phi}(z|x)$ and the prior distribution $p(z)$ over the latent variables. The prior distribution is typically chosen to be a simple distribution, such as a standard Gaussian $N(0; I)$.

The ELBO consists of two terms: the reconstruction term $\mathbb{E}_{q(z|x)}[\log p(x|z)]$, which encourages the decoder to reconstruct the input data accurately, and the regularization term $D_{KL}(q(z|x)||p(z))$, which encourages the approximate posterior to be close to the prior distribution. To make the optimization tractable, the approximate posterior is often parameterized as a multivariate Gaussian distribution with a diagonal covariance matrix:

$$q(z|x) = \mathcal{N}(z; \mu(x); \sigma^2(x)I) \tag{4}$$

where $\mu(x)$ and $\sigma(x)$ are the mean and standard deviation vectors predicted by the encoder network. During training, the VAE is optimized using stochastic gradient descent by sampling from the approximate posterior using the reparameterization trick (43). The reparameterization trick enables backpropagation through the sampling process by expressing the latent variable z as a deterministic function of the mean and standard deviation vectors and a random noise vector $\mathcal{N}(0; I)$:

$$z = \mu(x) + \sigma(x) \odot \epsilon \tag{5}$$

where $\odot$ denotes element-wise multiplication. VAEs have been successfully applied to various tasks, including image generation (28), text generation (12), and unsupervised representation learning (34). They provide a principled way to learn a generative model of the data distribution while also learning a structured latent space that captures meaningful factors of variation in the data.

3.4.3 Vector Quantized VAEs

Vector Quantized Variational Autoencoder (VQ-VAE)s (70; 85) are a class of generative models that combine the principles of VAEs with vector quantization. VQ-VAEs aim to learn a discrete latent representation of the input data by quantizing the continuous latent space learned by a VAE. In a VQ-VAE, the encoder network maps

the input data x to a continuous latent representation $z_e(x)$. The continuous latent representation is then quantized using a discrete codebook $\mathcal{C} = \{e_1, \ldots, e_K\}$, where each $e_i \in \mathbb{R}^D$ is a codebook vector and K is the size of the codebook. The quantization is performed by finding the nearest codebook vector to the continuous latent representation:

$$z_q(x) = e_k, \quad \text{where} \quad k = \arg\min_i \|z_e(x) - e_i\|_2 \tag{6}$$

The decoder network then reconstructs the input data from the quantized latent representation $z_q(x)$. The training of a VQ-VAE involves three loss terms: the reconstruction loss, the codebook loss, and the commitment loss. The reconstruction loss ensures that the decoder can accurately reconstruct the input data from the quantized latent representation:

$$L_{recon} = \|x - \hat{x}\|_2^2 \tag{7}$$

where $\hat{x}$ is the reconstructed input data. The codebook loss encourages the codebook vectors to be close to the continuous latent representations:

$$L_{codebook} = \|sg[z_e(x)] - e_k\|_2^2 \tag{8}$$

where $sg[]$ denotes the stop-gradient operation, which treats its argument as a constant during backpropagation. The commitment loss encourages the continuous latent representations to be close to the codebook vectors:

$$L_{commit} = \|z_e(x) - sg[e_k]\|_2^2 \tag{9}$$

The overall training objective of a VQ-VAE is a weighted sum of these three loss terms:

$$L_{VQ\text{-}VAE} = L_{recon} + L_{codebook} + L_{commit} \tag{10}$$

where and are hyperparameters that control the balance between the di erent loss terms. VQ-VAEs have been shown to learn discrete and interpretable latent representations, which can be useful for various downstream tasks such as image generation (70), speech synthesis, and unsupervised learning (16). The discrete nature of the latent space in VQ-VAEs allows for more e cient and controllable generation compared to continuous latent spaces in traditional VAEs.

3.4.4 Generative Adversarial Network

Generative Adversarial Networks (GANs) (27) are a class of generative models that learn to generate realistic samples by framing the learning process as a two-player adversarial game. GANs consist of two neural networks: a generator network G and a discriminator network D. The generator aims to generate realistic samples that fool the discriminator, while the discriminator tries to distinguish between real samples from the training data and fake samples generated by the generator. The generator network G takes a random noise vector z drawn from a prior distribution $p_z(z)$ and maps it to a generated sample $G(z)$. The discriminator network D takes an input sample x (either real or generated) and outputs a scalar probability $D(x)$ indicating the likelihood of the sample being real. The training of a GAN involves optimizing the following minimax objective:

$$\min_G \max_D V(D;G) = E_{x \sim p_{data}(x)}[\log D(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{11}$$

The discriminator D is trained to maximize the log-likelihood of assigning the correct label to both real samples from the training data and fake samples generated by the generator. The generator G is trained to minimize the log-likelihood of the discriminator assigning the correct label to the generated samples, which is equivalent to maximizing $\log(1 - D(G(z)))$. The training process alternates between updating the discriminator and the generator. The discriminator is updated by ascending its stochastic gradient:

$$\nabla_d \frac{1}{m} \sum_{i=1}^{m} [\log D(x^{(i)}) + \log(1 - D(G(z^{(i)})))] \tag{12}$$

where $_d$ are the parameters of the discriminator, m is the batch size, $x^{(i)}$ are real samples from the training data, and $z^{(i)}$ are noise vectors sampled from the prior distribution. The generator is updated by descending its stochastic gradient:

$$r_{\ g} \frac{1}{m} \sum_{i=1}^{m} \log(1 \quad D(G(z^{(i)})) \tag{13}$$

where $_g$ are the parameters of the generator. GANs have achieved remarkable success in generating high-quality samples across various domains, including images (42), music, and text. However, training GANs can be challenging due to issues such as mode collapse, where the generator learns to generate a limited variety of samples, and instability, where the generator and discriminator fail to converge to a stable equilibrium. Various techniques have been proposed to address these challenges, such as the Wasserstein GAN Wasserstein Generative Adversarial Network (WGAN) (8), which uses the Wasserstein distance as a more stable objective function, and the Spectral Normalization GAN (57), which enforces Lipschitz continuity on the discriminator to improve training stability.

3.4.5 Limitations of Autoencoders, Variational Autoencoders, and Generative Adversatial Networks in Fine-Grained Control over Generation

While AEs, VAEs, and GANs have shown remarkable success in generating realistic and diverse samples, they often lack ne-grained control over the generation process. This limitation hinders their applicability in scenarios where precise control over speci c attributes or characteristics of the generated samples is required.

AEs learn to compress and reconstruct input data, capturing the most salient features in the latent space. However, the latent space learned by AEs is often entangled and lacks interpretability (9). It is challenging to disentangle speci c factors of variation in the latent space, making it di cult to control individual attributes of the generated samples independently.

VAEs address some of the limitations of AEs by introducing a probabilistic framework and enforcing a structured latent space. VAEs aim to learn a disentangled

representation where each dimension of the latent space corresponds to a meaningful factor of variation (34). However, achieving perfect disentanglement is challenging, and the level of control over individual attributes is often limited. The generated samples from VAEs tend to be blurry and lack ne details, as the reconstruction objective and the regularization term in the VAE loss function prioritize overall similarity rather than precise attribute control.

GANs have shown impressive results in generating high-quality samples, particularly in the image domain. However, GANs also su er from limited control over speci c attributes of the generated samples. The latent space learned by GANs is often unstructured and lacks explicit interpretability (50). Modifying speci c attributes of the generated samples requires discovering and manipulating relevant directions in the latent space, which can be challenging and requires post-hoc analysis techniques (78).

E orts have been made to improve the controllability of generative models. InfoGAN (18) extends the GAN framework by introducing a mutual information objective to learn disentangled representations. However, the level of control achieved by InfoGAN is still limited, and it requires careful design of the latent code and the mutual information estimator. Other approaches, such as Conditional VAEs (80) and Conditional GANs (56), incorporate additional conditioning information to guide the generation process. While these approaches provide some level of control, they often rely on explicitly labelled data and may struggle with ne-grained attribute manipulation.

The lack of ne-grained control over the generation process in AEs, VAEs, and GANs can be attributed to several factors, including the entangled nature of the latent space, the di culty in achieving perfect disentanglement, and the trade-o between reconstruction quality and attribute control. Overcoming these limitations and enabling precise control over speci c attributes while maintaining the generation quality remains an open challenge in the eld of generative models (38).

Abbildung 7: Visualization of the Di usion Process: Forward Noise Addition and Reverse Denoising

3.5 Di usion Models

DMss have emerged as a powerful class of generative models, capable of producing high-quality samples in various domains, including images, audio, and video. These models are inspired by non-equilibrium thermodynamics and learn to reverse a di usion process, which gradually adds noise to the data until it becomes indistinguishable from a standard normal distribution (35; 79). As illustrated in Figure 7, the forward di usion process progressively adds noise to an image, while the generative reverse process learns to denoise the corrupted data step by step. Through this process, DMss can generate realistic samples that capture the underlying data distribution. Recent advancements, such as improved architectures, training techniques, and sampling strategies, have further enhanced the performance and e ciency of DMss, making them a promising approach for generative modeling (21; 61).

3.5.1 History of Di usion Models

The development of DMss can be traced back to the study of non-equilibrium thermodynamics and the di usion process. In the early 2000s, researchers explored the idea of using di usion processes for data generation and modeling. One of the earliest works in this direction was the study of di usion maps by Coifman and Lafon (20), which introduced a framework for dimensionality reduction and data representation using di usion processes.

In 2015, Sohl-Dickstein et al. (79) proposed a deep unsupervised learning ap-

proach using non-equilibrium thermodynamics, laying the foundation for modern DMss. They introduced a generative model that learned to reverse a di usion process, starting from a noise distribution and gradually denoising the data to generate samples. This work highlighted the potential of di usion-based models for capturing complex data distributions.

The denoising di usion probabilistic model (DDPM) (35), introduced by Ho et al. in 2020, marked a signi cant milestone in the development of DMss. DDPM utilized a variational inference framework and a noise scheduling scheme to improve the training stability and sample quality of DMss. The authors demonstrated impressive results on image generation tasks, showcasing the ability of DMss to generate high-quality samples.

Since the introduction of DDPM, various advancements and extensions have been proposed to further improve the performance and capabilities of DMss. Nichol and Dhariwal (61) introduced improvements to the architecture and training process of DMss, resulting in enhanced sample quality and e ciency. Dhariwal and Nichol (21) demonstrated that DMss can outperform generative adversarial networks (GANs) on image synthesis tasks, establishing DMss as a competitive alternative to GANs.

Researchers have also explored the application of DMss to various domains beyond image generation. For example, Chen et al. (17) proposed WaveGrad, a di usion-based model for audio generation, showcasing the ability of DMss to generate high- delity audio samples. Luo and Hu (53) introduced Di Wave, another di usion model for audio synthesis, further demonstrating the versatility of DMss.

The success of DMss has led to their adoption in a wide range of tasks, including image super-resolution (46), image inpainting (52), and text-to-image synthesis (69). These applications showcase the potential of DMss to generate high-quality samples and solve complex generative tasks.

Recent research has focused on improving the e ciency and scalability of DMss. E orts have been made to reduce the computational cost and memory requirements of DMss, enabling their application to larger-scale datasets and higher-resolution data. Techniques such as progressive growing (36) and multiscale architectures (83)

have been proposed to address these challenges.

The history of DMss highlights the rapid progress and growing interest in this eld. From the early exploration of di usion processes for data representation to the current state-of-the-art models, DMss have demonstrated their potential as a powerful generative modelling framework. As research continues to advance, it is expected that DMss will play an increasingly important role in various generative tasks and applications.

3.5.2 Di usion Model Architecture

The architecture of di usion models typically consists of two main components: the forward di usion process and the reverse di usion process.

Forward Di usion Process : The forward di usion process is a Markov chain that gradually adds noise to the input data over a xed number of timesteps, as illustrated in Figure 7. Let x_0 denote the original input data, and x_t denote the noisy data at timestep t. The forward di usion process is de ned by a sequence of noise scales $_1; \ _2; : : : ; \ _T$, where T is the total number of timesteps.

At each timestep t, the noisy data x_t is obtained by adding Gaussian noise to the previous timestep's data $x_{t \ 1}$ according to the following equation:

$$q(x_t j x_{t \ 1}) = N (x_t; \sqrt{1 \ \ _t} x_{t \ 1}; \ _t I) \tag{14}$$

where $N (; ; \ ^2)$ denotes a Gaussian distribution with mean and variance ^{2}, and I is the identity matrix.

The noise scales $_t$ are typically chosen to follow a prede ned schedule, such as a linear or cosine schedule, that increases the amount of noise added at each timestep. Figure 8 shows an example of a cosine schedule for the noise scales.

Reverse Di usion Process : The reverse di usion process is the generative process that learns to denoise the noisy data and generate samples. It is de ned by

Abbildung 8: Cosine Schedule for Noise Scales

a sequence of denoising functions $f_s(x_t; t)$, parameterized by a deep neural network with parameters .

At each timestep t, the denoising function $f\ (x_t; t)$ takes the noisy data x_t and the timestep t as input and predicts the noise that was added at that timestep. The predicted noise is then subtracted from the noisy data to obtain the denoised data $\hat{x}_{t\ 1}$:

$$\hat{x}_{t\ 1} = \sqrt{\frac{1}{1\ \ _t}}\ x_t\ \ \sqrt{\frac{_t}{1\ \ _t}} f\ (x_t; t) \tag{15}$$

where $_t = \prod_{s=1}^{t} (1\ \ _s)$ is the cumulative product of the noise scales.

The denoising function $f\ (x_t; t)$ is typically implemented as a deep neural network, such as a U-Net (74) or a residual network (3). The network takes the noisy data x_t and the timestep t as input and outputs the predicted noise.

Training and Sampling

During training, the di usion model is optimized to minimize the weighted sum of the denoising losses at each timestep:

Abbildung 9: Training (left) and Sampling (right) Algorithms for Di usion Models (35)

$$L = \sum_{t=1}^{T} {}_t E_{x_0;} \quad j f \ (x_t; t) \quad j^2 \tag{16}$$

where $_t$ are the weights assigned to each timestep, and is the noise term sampled from a standard Gaussian distribution.

To generate samples, the di usion model starts from random noise x_T and iteratively applies the reverse di usion process for T timesteps. At each timestep, the denoising function $f \ (x_t; t)$ is used to predict and subtract the noise, gradually denoising the data until reaching the nal generated sample x_0.

The architecture of di usion models allows them to capture complex data distributions by learning to denoise the data step by step. The U-Net or residual network architecture used for the denoising function enables the model to capture multi-scale features and generate high-quality samples. Figure 9 illustrates the training and sampling algorithms for di usion models, highlighting the iterative nature of both processes.

Recent advancements in di usion model architectures include the use of attention mechanisms (86), conditional generation (21), and hierarchical architectures (36) to further improve the quality and controllability of the generated samples. These improvements build upon the core algorithms shown in Figure 9, enhancing various aspects of the training and sampling processes.

3.6 Knowledge-Guided Generative Models

Knowledge-guided generative models represent an emerging paradigm that integrates structured domain knowledge, typically encoded in knowledge graphs, into the generative process of deep learning models. By leveraging the rich semantic relationships and conceptual information captured by KGs, these models aim to enhance the coherence, contextual relevance, and controllability of the generated outputs. The incorporation of KGs has been explored across various generative frameworks, such as GANs (88), VAEs, and language models.

In the context of GANs, KGs have been employed to guide the image generation process towards speci c semantic concepts and relationships, enabling the creation of visually coherent and semantically meaningful images. Similarly, knowledge-guided language models have demonstrated improved factual accuracy and contextual consistency in the generated text by aligning the generation process with the domain knowledge encoded in the graphs (95).

However, the integration of KGs with more advanced generative models, such as di usion models (DMs), remains a relatively underexplored area. The proposed research aims to bridge this gap by developing a novel framework that leverages KGEs to guide the generation process of DMs, enabling ne-grained control over the generated samples while ensuring adherence to the semantic constraints and factual knowledge represented in the input KGs (60).

This approach holds promise for enhancing the interpretability, controllability, and domain-speci city of generative models, paving the way for more reliable and knowledge-grounded generative applications in AI.

3.7 LLaVA Model

The Large Language and Vision Assistant (LLaVA) (48) represents a signi cant advancement in multimodal AI models, combining Large Language Model (LLM) with visual understanding capabilities. LLaVA is designed to process and generate

Abbildung 10: Classi er guided di usion model

responses based on both textual and visual inputs, making it a powerful tool for a wide range of applications.

3.7.1 Architecture and Training

LLaVA is built upon a LLM foundation, typically using models like GPT-3.5 or GPT-4, and integrates a vision encoder to process image inputs. The model is trained using a two-stage approach:

1. Pretrained Image Encoder: LLaVA utilizes a pretrained vision transformer (ViT) (22) as its image encoder. This encoder is capable of extracting rich visual features from input images.

2. Multimodal Projector: A trainable linear layer is used to project the visual features into the language model's embedding space, enabling seamless integration of visual and textual information.

The model is then ne-tuned on a diverse dataset of image-text pairs, which includes visual question-answering tasks, image captioning, and general multimodal

dialogue. This training process allows LLaVA to develop a deep understanding of the relationship between visual and textual content.

3.7.2 Capabilities and Applications

LLaVA demonstrates impressive capabilities across various multimodal tasks:

1. Visual Question Answering: The model can accurately answer questions about the content of images, understanding complex visual scenes and their components.

2. Image Captioning: LLaVA can generate detailed and contextually relevant captions for a wide range of images.

3. Visual Reasoning: The model shows the ability to perform complex reasoning tasks based on visual inputs, such as comparing objects, identifying relationships, and making inferences.

4. Multimodal Dialogue: LLaVA can engage in open-ended conversations that involve both textual and visual context, seamlessly integrating information from both modalities.

5. Zero-shot Learning: The model demonstrates strong zero-shot performance on various vision-language tasks, indicating its ability to generalize to new scenarios without speci c training.

3.7.3 Relevance to Knowledge-Guided Generative Models

The success of LLaVA in integrating visual and textual information provides valuable insights for the development of knowledge-guided generative models:

1. Multimodal Integration: LLaVA's approach to combining visual and textual embeddings could inform strategies for integrating KGE with generative models.

2. Contextual Understanding: The model's ability to understand and reason about complex visual scenes aligns with the goal of using KGs to enhance the contextual relevance of generated content.

3. Fine-grained Control: LLaVA's success in speci c visual tasks suggests potential for achieving ne-grained control in generative models guided by structured knowledge.

4. Scalability: The model's architecture demonstrates how large-scale language models can be e ectively adapted for multimodal tasks, which could inform approaches to scaling knowledge-guided generative models.

While LLaVA focuses on integrating visual and textual information, our proposed research aims to extend these principles to the integration of KGs with di usion models. By leveraging the strengths of both approaches, we aim to develop generative models that not only understand and respond to multimodal inputs but also generate content guided by structured domain knowledge.

This literature review has highlighted the potential of integrating KGs with generative models, particularly DMs. Building on these insights, the next chapter will detail our novel methodology for incorporating KGE into LDM, aiming to address the identi ed research gap and enable more controlled and semantically consistent image generation.

4 Methodology

In this section, we introduce our KGs-guided DMs. We discuss di erent components of our model, including facial ontology, facial image KGs, embedding facial KGs, and integrating facial feature embedding into guided DMs.

4.1 Ontology Creation

The rst step in our methodology is the creation of an ontology based on the CelebA dataset (51). The CelebA dataset consists of 202,599 facial images of celebrities, each annotated with 40 binary attributes such as hair color, gender, and age. These attributes provide a rich set of semantic information that can be used to construct a domain-speci c ontology for the generation of facial images.

To create the ontology, we follow a systematic approach that involves the following steps:

4.1.1 Concept Identi cation

We start by identifying the main terms in the domain of facial images. These terms correspond to the key concepts and objects relevant to generating facial images. In the case of the CelebA dataset, the main concepts include Face, Hair, Eyes, Nose, Mouth, and Skin. Each of these concepts represents a piece of high-level information that encompasses various attributes and relationships.

4.1.2 Attribute Assignment

We assign relevant attributes to each entity based on the annotations provided in the CelebA dataset. These attributes capture the speci c characteristics and properties of the entities (images). For example, the Hair can have attributes such as:

- Color : e.g., black, blonde, brown

- Style : e.g., straight, wavy, curly

Similarly, the Eyes can have attributes like:

- Color : e.g., blue, brown, green
- Shape: e.g., round, almond, hooded

4.1.3 Relationship Establishment

To capture the semantic relationships between entities, we de ne the connections and hierarchical structure of the ontology. This involves specifying the relationships between entities using ontological constructs such as \is-a" (subsumption) and \has-a" (composition). For instance, we establish relationships like:

- Face is composed of Hair, Eyes, Nose, Mouth, and Skin

We also de ne relationships between attributes, such as:

- \Blonde" is a subclass of \Hair Color"

4.1.4 Ontology Formalization

After identifying the concepts, attributes, and relationships, we formalize the ontology using a standard ontology language such as OWL (55). This involves:

- Creating classes for each entity
- De ning properties for the attributes
- Specifying the relationships using ontological axioms

The formalized ontology provides a structured representation of the domain knowledge that can be used to guide the image generation process.

4.1.5 Ontology Validation

To ensure the quality and consistency of the created ontology, we perform validation checks. This includes:

- Verifying the logical consistency of the ontology
- Checking for any inconsistencies or contradictions in the de ned relationships
- Ensuring that the ontology accurately captures the domain knowledge

The resulting ontology serves as a structured representation of the semantic knowledge contained in the CelebA dataset. It provides a foundation for guiding the image generation process by incorporating domain-speci c concepts, attributes, and relationships. The ontology can be used to:

- Control the generation of facial images with desired characteristics
- Ensure semantic consistency in the generated samples

Abbildung 11: Visual representation of the facial ontology

4.2 Knowledge Graph Construction

After creating the ontology based on the CelebA dataset, the next step is to construct a KGs that represents the structured knowledge in a graph-based format. The KGs will serve as a formalized representation of the ontology, enabling e cient reasoning, querying, and integration with the generative model. This knowledge graph structure builds upon the ontology visualized in Figure 11, translating the conceptual model into a graph-based representation. The process of constructing the KGs involves the following steps:

4.2.1 Entity Mapping

We start by mapping the entities (images) to the concepts de ned in the ontology and represent them as the KGs with both ontology and instance levels. Each concept, such as Face, Hair, Eyes, Nose, Mouth, and Skin, becomes a node in the graph. These nodes represent the key concepts and objects in the domain of facial image generation.

4.2.2 Attribute Representation

We represent the attributes associated with each entity as properties of the corresponding nodes in the KGs. The attributes capture the speci c characteristics and features of the entities. For example, the Hair node can have properties like:

- Color : e.g., \Blonde"
- Style : e.g., \Straight"
- Length : e.g., \Long"

These attributes provide additional semantic information about the entities.

4.2.3 Relationship Representation

To capture the semantic relationships between entities, we represent the relationships de ned in the ontology as edges in the KGs. The edges connect the nodes based on the speci ed relationships, such as \is-a" (subsumption) and \has-a" (composition). For instance:

- An edge can be created between the Face node and the Hair node to represent the compositional relationship \Face has Hair"
- Edges can be created to represent hierarchical relationships between attributes, such as \Blonde is a subclass of Hair Color"

4.2.4 Triple Generation

The KGs is typically represented using a triple-based format, such as RDF (45). Triples consist of a subject, predicate, and object, representing the relationship between two entities. We generate triples based on the concepts, attributes, image nodes, and relationships de ned in the ontology. For example:

- A triple can be created as (Face, hasHair, Hair), indicating that a face has hair.
- Triples can be generated for attribute values, such as (Hair, hasColor, \Blonde").

4.3 Knowledge Graph Embedding Creation

After constructing the KGs, the next step is to create KGEs that capture the semantic information in a dense, continuous vector space. KGEs enable e cient computation, similarity measurement, and integration with machine learning models. In this work, we utilize PyKEEN (4), a Python library for KGEs, and speci cally employ the TransE model (11) for embedding creation.

The process of creating KGEs using PyKEEN and TransE involves the following steps:

4.3.1 Data Preparation

Before creating the embeddings, we need to prepare the KGs data in a format compatible with PyKEEN. PyKEEN expects the KGs to be represented as a set of triples (subject, predicate, object) in a tabular format, such as CSV or TSV. We export the KGs triples generated in the previous step into a suitable le format, ensuring that the entities and relationships are properly encoded.

4.3.2 PyKEEN Con guration

PyKEEN provides a exible and con gurable framework for training KGE models. We start by con guring PyKEEN with the necessary settings, such as specifying the dataset, the embedding model (TransE in this case), and the hyperparameters for training. TransE is a translation-based embedding model that represents relationships as translations in the embedding space, aiming to learn embeddings such that $h + r \quad t$ for a triple $(h; r; t)$.

4.3.3 Model Training

With the con gured PyKEEN pipeline, we proceed to train the TransH model on the KGs dataset. During training, the model learns to embed the entities and relationships into a continuous vector space while preserving the semantic structure of the KGs. The objective of TransH is to minimize the distance between the translated head entity embedding $(h + r)$ and the tail entity embedding (t) for valid triples, while maximizing the distance for corrupted triples.

4.3.4 Embedding Evaluation

After training the TransH model, we evaluate the quality of the generated embeddings. PyKEEN provides various evaluation metrics to assess the performance of

the embedding model in link prediction tasks:

Let T be the set of test triples, and for each triple $(h; r; t) \in T$, let $\text{rank}(e)$ be the rank of the correct entity e (either h or t) among all entities when predicting it given the other two elements of the triple.

Mean Rank : Mean Rank is the average rank of the correct entities in the test set.

Explanation: For each test triple, we rank all entities based on their predicted score for completing the triple. The rank of the correct entity is recorded, and the mean of these ranks is computed across all test triples.

Formula:

$$\text{Mean Rank} = \frac{1}{2|T|} \sum_{(h;r;t) \in T} (\text{rank}(h) + \text{rank}(t)) \tag{17}$$

Lower values indicate better performance.

Hits@k : Hits@k measures the proportion of correct entities that appear in the top k ranked candidates.

Explanation: For each test triple, we check if the rank of the correct entity is within the top k. The proportion of test triples where this is true is the Hits@k score.

Formula:

$$\text{Hits@k} = \frac{1}{2|T|} \sum_{(h;r;t) \in T} (I[\text{rank}(h) \quad k] + I[\text{rank}(t) \quad k]) \tag{18}$$

where $I[\,]$ is the indicator function. Higher values indicate better performance.

Mean Reciprocal Rank (MRR) : Mean Reciprocal Rank (MRR) is the average of the reciprocal ranks of the correct entities.

Explanation: For each test triple, we compute the reciprocal (1/rank) of the rank of the correct entity. The mean of these reciprocal ranks across all test triples is the MRR.

Formula:

$$\text{MRR} = \frac{1}{2|T|} \sum_{(h,r,t) \in T} \left(\frac{1}{\text{rank}(h)} + \frac{1}{\text{rank}(t)} \right) \tag{19}$$

Higher values indicate better performance.

Note: In all formulas, we average over both head and tail predictions (hence the factor of $\frac{1}{2|T|}$), as is common practice in KGE evaluation.

These metrics measure the model's ability to predict missing or unseen relationships in the KGs based on the learned embeddings. We evaluate the TransH embeddings using these metrics to ensure their e ectiveness in capturing the semantic information.

4.3.5 Embedding Extraction

Once the TransH model is trained and evaluated, we extract the learned embeddings for the entities and relationships in the KGs. PyKEEN allows us to access the embedding matrices, which contain the dense vector representations of the entities and relationships. We can export these embeddings into suitable formats, such as:

- NumPy arrays
- CSV les

These formats allow for further use in the generative model.

The resulting KGEs capture the semantic information of the entities and relationships in a compact and computationally e cient manner. These embeddings can be used to guide the image generation process by providing semantic representations that encode the attributes, relationships, and constraints of the facial domain.

By integrating the KGEs into the generative model, we can:

- Leverage the structured knowledge to control the generation process
- Ensure the semantic consistency and coherence of the generated facial images

4.3.6 Encoding Multiple Features Embeddings

To enhance computational e ciency and reduce the complexity of the KGEs, we employ an autoencoder-based dimensionality reduction technique. Starting with 4000-dimensional KGEs, we train an autoencoder to compress these embeddings into a more compact 100-dimensional representation. The process involves the following steps:

1. Input Preparation: We use a Bernoulli distribution to create a random binary vector of length 40, corresponding to the 40 binary attributes in the CelebA dataset (51). This vector is then used to randomly select which of the 40 features will be active for each input sample. The Bernoulli distribution for each element b_i of the vector b is de ned as:

$$P(b_i = k) = \begin{cases} p & \text{if } k = 1 \\ 1 - p & \text{if } k = 0 \end{cases} \tag{20}$$

   where p is the probability of success. This random selection helps in creating diverse input samples, potentially improving the robustness of the learned embeddings.

2. Autoencoder Architecture: The autoencoder consists of an encoder that maps the 4000-dimensional input to the 100-dimensional bottleneck layer, and a decoder that attempts to reconstruct the original input from this compressed representation.

3. Training: The autoencoder is trained to minimize the reconstruction error between the input and output embeddings. This process encourages the model to learn a compressed representation that retains the most salient features of the original KGEs.

4. Embedding Compression: Once trained, we use the encoder part of the autoencoder to compress the 4000-dimensional KGEs into 100-dimensional embeddings.

5. Semantic Preservation Analysis: To evaluate whether the semantic properties of the embeddings are preserved after compression, we plot both the input (4000-dimensional) and output (100-dimensional) embeddings in a 2D space. This visualization is typically achieved using dimensionality reduction techniques such as t-SNE or UMAP.

The 2D visualization allows us to compare the spatial relationships between embeddings before and after compression. If the semantic properties are well-preserved, we expect to see similar clustering patterns and relative distances between embeddings in both the input and output plots. This comparison helps us assess the quality of our dimensionality reduction process and ensure that the compressed embeddings retain the essential semantic information of the original KGs.

By successfully compressing the KGEs while preserving their semantic properties, we signi cantly reduce the computational resources required for subsequent processing in the KGGLDM pipeline, without sacri cing the richness of the knowledge representation. The use of the Bernoulli distribution in input preparation adds an element of stochasticity that may contribute to the robustness and generalization capability of the learned embeddings.

4.4 Knowledge Graph Embedding Integration into Di usion Model

To integrate the KGEs into the di usion model, we propose a novel approach called the KGGLDM. KGGLDM extends the class-guided di usion model (21) by replacing the class guidance with the KGEs. This modi cation allows the DMs to leverage the semantic information captured by the KGs to guide the image generation process.

The key idea behind KGGLDM is to condition the di usion process on the KGEs, enabling the model to generate images that adhere to the semantic constraints and relationships encoded in the KGs. The integration of KGEs into the DMs involves the following steps:

4.4.1 Embedding Conditioning

In KGGLDM, we modify the DMs architecture to incorporate the KGEs as conditioning information. Speci cally, we concatenate the KGE vectors with the noise representation at each timestep of the di usion process. This allows the model to learn the association between the semantic information and the generated images.

Let z_t denote the latent representation at timestep t, x_t denote the noisy image at timestep t, and e denote the KGE vector. The modi ed di usion process in KGGLDM can be formulated as:

$$z_t = f\ (x_t; t; e) \tag{21}$$

where f is the DMs parameterized by , and e is the KGE vector that guides the di usion process.

4.4.2 Training and Inference

During training, KGGLDM learns to generate images that are both realistic and semantically consistent with the KGEs. The model is trained end-to-end using the MSE as a objective function.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n} (x_i - \hat{x}_i)^2 \tag{22}$$

where x_i and $\hat{x}_i$ are the i-th elements of the input and reconstructed vectors, respectively.

At inference time, given a KGE vector e, KGGLDM generates an image x by sampling from the learned di usion process conditioned on e. The generated image will exhibit the desired semantic properties and relationships encoded in the KGE.

By integrating KGEs into the DMs, KGGLDM enables ne-grained control over the image generation process, allowing for the creation of semantically consistent

and meaningful images. The KGEs guide the di usion process, ensuring that the generated images align with the speci ed semantic constraints and relationships. This approach combines the generative power of DMs with the structured knowledge captured by KGs, enabling more controlled and interpretable image generation.

5 Experiments

This section describes the experimental setup, including the dataset, evaluation metrics, and implementation details, used to assess the performance of the proposed KGGLDM for facial image generation.

5.1 Dataset

We conduct our experiments using the CelebA dataset (51), which is widely used for facial image generation and attribute manipulation tasks. The CelebA dataset contains 202,599 facial images of celebrities, each annotated with 40 binary attributes. These attributes are: 5_o_Clock_Shadow, Arched_Eyebrows, Attractive, Bags_Under_Eyes, Bald, Bangs, Big_Lips, Big_Nose, Black_Hair, Blond_Hair, Blurry, Brown_Hair, Bushy_Eyebrows, Chubby, Double_Chin, Eyeglasses, Goatee, Gray_Hair, Heavy_Makeup, High_Cheekbones, Male, Mouth_Slightly_Open, Mustache, Narrow_Eyes, No_Beard, Oval_Face, Pale_Skin, Pointy_Nose, Receding_Hairline, Rosy_Cheeks, Sideburns, Smiling, Straight_Hair, Wavy_Hair, Wearing_Earrings, Wearing_Hat, Wearing_Lipstick, Wearing_Necklace, Wearing_Necktie, and Young.

Each attribute is binary, where 1 indicates the presence of the attribute and 1 indicates its absence. These attributes cover a wide range of facial characteristics, including hair color and style, facial features, accessories, and demographic information.

The dataset is split into three subsets:

- Training set: 162,770 images
- Validation set: 19,867 images
- Testing set: 19,962 images

This split allows for a fair evaluation of the model's performance and generalization ability.

To construct the KGs for KGGLDM, we utilize the attribute annotations provided in the CelebA dataset. Each image is associated with a set of attributes that describe the facial characteristics present in the image. These attributes serve as the basis for creating the ontology and KGs, as described in the methodology section.

Preprocessing steps applied to the CelebA dataset include:

- Resized all images to 32 32 pixels to ensure uniform input size
- Normalized pixel values to the range [1; 1]
- Augmented the dataset with random horizontal ips to increase diversity
- Balanced the dataset with respect to key attributes to mitigate bias

These steps ensure consistency and compatibility with the KGGLDM framework, while also addressing the imbalanced distribution of certain attributes in the dataset.

The CelebA dataset, along with its attribute annotations, provides a rich and diverse foundation for training and evaluating the KGGLDM framework. The dataset's large scale and variety of facial attributes make it suitable for assessing the model's capability to generate high-quality facial images while adhering to the semantic constraints encoded in the KGs.

By leveraging the CelebA dataset and its associated KGs, KGGLDM aims to generate semantically consistent and visually realistic facial images that align with the speci ed attributes and relationships. The experiments conducted on this dataset will provide insights into the e ectiveness of integrating KGE into the DMs for controlled and interpretable facial image generation.

5.2 Experimental Setup

5.2.1 Model Con gurations

KGGLDM is built upon a modi ed U-Net architecture, which is adapted to incorporate knowledge graph embeddings. The model is implemented as follows:

image_id	Male	Smiling	Black_Hair	Blond_Hair	Wearing_Hat
000001.jpg	1	1	1	1	1
000002.jpg	1	1	1	1	1
000003.jpg	1	1	1	1	1
000004.jpg	1	1	1	1	1

Tabelle 1: Sample feature data for CelebA dataset with 5 features.

Abbildung 12: Unet Architecture

- Base Architecture: The core of KGGLDM is a U-Net structure, which is well-suited for image generation tasks due to its ability to capture multi-scale features.
- Dimensionality: The model uses a base dimension $dim = 64$, which can be adjusted as a hyperparameter. The actual dimensions used in di erent parts of the network are multiples of this base dimension, following the pattern $(1; 2; 4; 8; 8)$ dim.
- Embedding Integration: Knowledge graph embeddings, with a dimension of 64, are integrated into the model at multiple levels. This is achieved through a custom KGConditioning layer that projects the embeddings to match the spatial dimensions of the feature maps at each level of the U-Net.
- Attention Mechanisms: The model incorporates both standard attention and linear attention mechanisms to enhance its ability to capture long-range dependencies in the data.
- Residual Connections: Residual blocks are used throughout the network to facilitate gradient ow and enable the training of deeper networks.
- Time Embedding: A sinusoidal position embedding is used to encode the timestep information, which is crucial for the di usion process.
- Input and Output: The model is designed to work with Red, Green, Blue (RGB) images of size 32 32 pixels, though this can be adjusted as needed.

The di usion process in KGGLDM uses a cosine beta schedule for 1000 timesteps, allowing for a gradual and controlled denoising process. The model is trained using the Adam optimizer with a learning rate of $1e$ 4 and a batch size of 128.

This con guration allows KGGLDM to e ectively incorporate knowledge graph information into the di usion process, enabling semantically guided image generation.

Abbildung 13: KGGLDM architecture

5.2.2 Training Settings

KGGLDM is trained on the CelebA dataset using the following con guration:

- Batch Size: The global batch size is set to 128, adjusted for multi-Graphics Processing Unit (GPU) training using the formula: GLOBAL_BATCH_SIZE = BATCH_SIZE num_replicas_in_sync.
- Epochs: The model is trained for 50 epochs by default, which can be adjusted using the command-line argument-epochs .
- Optimizer: Adam optimizer is used with following settings:
  - Optimizer: Adam with $_1 = 0:9$, $_2 = 0:999$
  - Initial learning rate: 1e 4
  - Learning rate schedule: Cosine annealing with warm restarts

- Weight decay: $1e-5$
- Gradient clipping: Maximum norm of 1.0

- Loss Function: The loss function is defined as the mean squared error between the real noise and the predicted noise, computed over the entire batch.

The training process for KGGLDM is designed to leverage modern distributed computing architectures, enabling efficient utilization of multiple GPUs. This approach not only accelerates the training process but also allows for scaling to larger datasets and more complex models.

- Distributed Training Strategy: We employ a data-parallel distribution strategy, where the model is replicated across multiple GPUs, and each GPU processes a different batch of data. This strategy is implemented using TensorFlow's tf.distribute.MirroredStrategy, which automatically handles the distribution of computations and the aggregation of gradients.

- Gradient Aggregation: During each training step, gradients are computed independently on each GPU and then aggregated across all devices using an all-reduce operation. This ensures that the model parameters are updated consistently across all replicas.

- Training Loop: The training loop is structured to include both training and validation steps:

  - Training Step: In each training step, a batch of data is processed, the loss is computed, and the model parameters are updated using the Adam optimizer.
  - Validation Step: Periodically (e.g., after every epoch), the model is evaluated on a held-out validation set to monitor its generalization performance.
  - Metrics Tracking: We track several metrics during both training and validation:

    * Loss: The primary optimization objective (e.g., the di usion loss).
    * Mean Squared Error (MSE): Provides a direct measure of the reconstruction quality.
    * Additional metrics: FID and Structural Similarity Index Measure (SSIM) are computed at regular intervals to assess the quality of generated images.

- Checkpointing: Model checkpoints are saved at regular intervals, allowing for training resumption and later analysis.

- Learning Rate Scheduling: We implement a cosine decay learning rate schedule to improve convergence and nal model performance.

- Early Stopping: To prevent over tting, we employ an early stopping mechanism based on the validation loss.

In particular, the semantic consistency loss mentioned in the original text is not explicitly implemented in the code snippet provided. Instead, the knowledge graph embeddings are directly incorporated into the model architecture through the KGConditioning layers, guiding the generation process implicitly.

5.3 Evaluation Strategy

To comprehensively assess the e ectiveness of our KGGLDM, we employ a multi-faceted evaluation strategy. This approach allows us to measure both the quality of the generated images and their semantic consistency with the knowledge graph guidance. We utilize the following metrics:

- Fechet Inception Distance (FID) (33): FID measures the distance between the distribution of generated images and the distribution of real images in the feature space of a pre-trained Inception network. Lower FID values indicate better quality and diversity of the generated images. The FID is calculated as:

(23) $$\text{FID} = j \quad {}_r \quad {}_g j^2 + \text{Tr}(\quad {}_r + \quad {}_g \quad 2(\quad {}_r \quad {}_g)^{1=2})$$

where $_r$ and $_g$ are the mean feature representations of real and generated images respectively, and $_r$ and $_g$ are their covariance matrices. Tr denotes the trace of a matrix. We choose FID because it is a widely accepted metric in the eld of generative models, providing a robust measure of both image quality and diversity. It captures higher-level features of the images, making it sensitive to mode collapse and other common issues in image generation.

- Structural Similarity Index (SSIM) (90): SSIM assesses the perceptual similarity between the generated images and the real images. It considers the structural information, luminance, and contrast of the images. Higher SSIM values suggest better perceptual quality of the generated images. The SSIM between two image patches x and y is de ned as:

$$\text{SSIM}(x; y) = \frac{(2\ _x\ _y + c_1)(2\ _{xy} + c_2)}{(\ _x^2 + \ _y^2 + c_1)(\ _x^2 + \ _y^2 + c_2)} \tag{24}$$

where $_x$, $_y$ are the average pixel intensities, $_x$, $_y$ are the standard deviations, $_{xy}$ is the covariance of x and y, and c_1, c_2 are constants to avoid division by zero. We include SSIM to complement FID, as it provides a di erent perspective on image quality. While FID focuses on the overall distribution of images, SSIM captures local structural similarities, which is particularly relevant for assessing the delity of generated facial features.

- Attribute Accuracy: To evaluate the semantic consistency of the generated images with respect to the speci ed attributes, we train a separate attribute classi er on the CelebA dataset. The classi er predicts the presence or absence of each attribute in the generated images. The attribute accuracy is calculated as:

(25)

$$\text{Attribute Accuracy} = \frac{\text{Number of Correctly Predicted Attributes}}{\text{Total Number of Attributes}} \quad 100\%$$

This metric is crucial for assessing how well our KGGLDM incorporates the knowledge graph guidance into the generated images. It directly measures the model's ability to generate images that adhere to the speci ed semantic attributes.

- Human Evaluation: While quantitative metrics provide valuable insights, they may not fully capture the subjective quality and semantic consistency of the generated images. Therefore, we conduct a user study to assess the visual quality and semantic consistency from a human perspective. Participants are presented with generated images and asked to rate their realism and adherence to the speci ed attributes on a Likert scale from 1 to 5. The human evaluation score is calculated as:

  (26) $$\text{Human Evaluation Score} = \frac{1}{N} \sum_{i=1}^{N} \text{Rating}_i$$

  where N is the total number of ratings and Rating_i is the individual rating for each image. Higher scores indicate better perceptual quality and semantic alignment. This qualitative evaluation complements our quantitative metrics, providing insights into the subjective quality of the generated images and their perceived alignment with the speci ed attributes.

By combining these diverse metrics, we aim to provide a comprehensive evaluation of our KGGLDM approach, assessing both the visual quality of the generated images and their semantic consistency with the knowledge graph guidance. This multi-faceted evaluation strategy allows us to compare our model's performance against baselines and state-of-the-art methods across various dimensions of image generation quality and controllability.

5.4 Baselines

We compare KGGLDM against the following baselines:

- Unconditional Di usion Model (35)
- Classi er-Guided Di usion Model (21)

These baselines were chosen to represent state-of-the-art methods in both di usion models and GANs for conditional image generation.

5.5 Hardware and Software Specifications

All experiments were conducted on a system with the following specifications:

- GPU: 4 × NVIDIA Tesla A10 (40GB VRAM each)
- Central Processing Unit (CPU): Intel Xeon Gold 6248R @ 3.00GHz (48 cores)
- RAM: 384GB DDR4
- Software: TensorFlow 2.13.0, PyTorch 1.9.0, CUDA 10.1, PyKEEN 1.6.0

6 Results

This chapter presents the experimental results of our KGGLDM. We first analyze the statistics of the knowledge graph, then compare different knowledge graph embedding techniques, and finally present quantitative and qualitative results.

6.1 Knowledge Graph Statistics

Table 2 presents summary statistics of the knowledge graph used in the KGGLDM framework. The knowledge graph consists of 202,639 entities and 8,103,960 relationships, which form a total of 8,103,960 triples. The knowledge graph forms a single connected component, meaning that there are no isolated subgraphs and all entities are reachable from any other entity through a sequence of relationships.

Statistics	Value
Number of Entities	202,639
Number of Relationships	8,105,560
Number of Triples	8,105,660

Tabelle 2: Statistics of the Knowledge Graph

6.2 Comparison of Knowledge Graph Embedding Methods

We evaluated several KGE techniques to determine which would be most effective in integrating structured knowledge into our latent diffusion model. The embedding methods compared were:

- TransE (11)
- TransH (99)
- TransD (40)
- ComplEx (82)

ˆ ConvE

The performance of these embedding methods was evaluated using standard link prediction metrics: Mean rank, hits@10, and MRR. Lower values are better for Mean Rank, while higher values are better for Hits@10 and MRR.

Method	Mean Rank #	Hits@10 "	MRR "
TransE	243	0.472	0.294
TransH	190	0.565	0.380
TransD	229	0.506	0.325
ComplEx	212	0.528	0.342
ConvE	200	0.540	0.355

Tabelle 3: Performance comparison of knowledge graph embedding methods

The performance of various knowledge graph embedding methods was rigorously evaluated on our facial attribute knowledge graph derived from the CelebA dataset. As evidenced in Table 3, TransH (Translating on Hyperplanes) consistently outperformed other state-of-the-art embedding techniques across all key metrics.

TransH achieved the lowest Mean Rank of 190, signi cantly better than its closest competitor, ConvE, which scored 200. This indicates that TransH more accurately positions correct entities closer to the top of the ranked list during link prediction tasks. The superior performance is further corroborated by the Hits@10 score, where TransH achieved 0.565, meaning that for 56.5% of the test cases, the correct entity was ranked within the top 10 predictions. This is a substantial improvement over the second-best performer, ConvE, which managed 0.540.

Perhaps most tellingly, TransH excelled in the MRR metric, scoring 0.380. This metric is particularly important as it provides a balanced view of the model's performance across all ranks. The notably higher MRR score suggests that TransH not only performs well in top-k predictions but also maintains high accuracy across a broader range of rank positions.

The superior performance of TransH can be attributed to its unique approach of modeling relations as hyperplanes in the embedding space, rather than as trans-

lations in a single vector space. This allows TransH to capture complex and diverse relations more e ectively, which is particularly crucial in our facial attribute domain where relationships between entities can be multifaceted and context-dependent.

Moreover, TransH's ability to project entities onto relation-speci c hyperplanes before performing translations enables it to handle various relation properties such as re exivity, symmetry, and transitivity more adeptly. This is especially bene cial in our knowledge graph, where attributes like imilar hair style or comparable age may exhibit such properties.

The consistent superiority of TransH across all metrics underscores its robustness and reliability in capturing the intricate semantic structures present in our facial attribute knowledge graph. This performance advantage directly translates to enhanced guidance for our KGGLDM, enabling more accurate and semantically rich image generation.

In light of these results, we have chosen TransH as the primary embedding method for our KGGLDM framework. This decision is expected to signi cantly contribute to the model's ability to generate high-quality, attribute-consistent facial images, thereby advancing our research objectives in knowledge-guided image generation.

6.3 Latent Space Analysis of Knowledge Graph Embeddings

To analyze and compare the di erent knowledge graph embedding methods, we employed a two-step dimensionality reduction process. First, we used an autoencoder to compress the knowledge graph embeddings into a lower-dimensional latent space. Then, we applied UMAP to further reduce the dimensionality for visualization purposes.

6.3.1 Autoencoder for Dimensionality Reduction

We designed an autoencoder to compress the knowledge graph embeddings into a 100-dimensional latent space. The autoencoder architecture is as follows:

- Encoder: Input dimension! 4000! 100
- Decoder: 100! 4000! Output dimension

The autoencoder was trained to minimize the reconstruction loss, ensuring that the 100-dimensional latent space retains the essential information from the original embeddings. This step allows us to compare embedding methods with di erent original dimensions on an equal footing.

6.4 UMAP Visualization of CelebA Dataset Features

To gain insights into the latent space representation of facial attributes, we applied UMAP to visualize the 100-dimensional embeddings produced by our autoencoder. UMAP was chosen for its ability to preserve both local and global structure of the data, as well as its e ciency in handling large datasets.

We visualized several key attributes from the CelebA dataset to understand how they are represented in the latent space. Figures 14 through 18 show these visualizations.

These UMAP visualizations reveal several interesting patterns in how facial attributes are captured in the latent space:

- Gender Figure 14: The visualization shows a clear separation between male and female attributes, with a smooth transition in between. This suggests that the latent space has captured the spectrum of gender expression.
- Age Figure 15: Age is represented as discrete category, indicating that the model has learned a nuanced representation of age-related features.
- Hair Color Figure 16: Di erent hair colors form distinct clusters, with logical positioning (e.g., black hair between black and other). This demonstrates the model's ability to distinguish and organize various hair colors.

Abbildung 14: UMAP visualizations for gender attribute

Abbildung 15: UMAP visualizations for age attribute