future internet

MDPI

Review

The Power of Generative AI: A Review of Requirements, Models, Input–Output Formats, Evaluation Metrics, and Challenges

Ajay Bandi *, Pydi Venkata Satya Ramesh Adapa and Yudu Eswar Vinay Pratap Kumar Kuchi

School of Computer Science and Information Systems, Northwest Missouri State University, Maryville, MO 64468, USA; s555173@nwmissouri.edu (P.V.S.R.A.); s555080@nwmissouri.edu (Y.E.V.P.K.K.)
* Correspondence: ajay@nwmissouri.edu

Abstract: Generative artificial intelligence (AI) has emerged as a powerful technology with numerous applications in various domains. There is a need to identify the requirements and evaluation metrics for generative AI models designed for specific tasks. The purpose of the research aims to investigate the fundamental aspects of generative AI systems, including their requirements, models, input–output formats, and evaluation metrics. The study addresses key research questions and presents comprehensive insights to guide researchers, developers, and practitioners in the field. Firstly, the requirements necessary for implementing generative AI systems are examined and categorized into three distinct categories: hardware, software, and user experience. Furthermore, the study explores the different types of generative AI models described in the literature by presenting a taxonomy based on architectural characteristics, such as variational autoencoders (VAEs), generative adversarial networks (GANs), diffusion models, transformers, language models, normalizing flow models, and hybrid models. A comprehensive classification of input and output formats used in generative AI systems is also provided. Moreover, the research proposes a classification system based on output types and discusses commonly used evaluation metrics in generative AI. The findings contribute to advancements in the field, enabling researchers, developers, and practitioners to effectively implement and evaluate generative AI models for various applications. The significance of the research lies in understanding that generative AI system requirements are crucial for effective planning, design, and optimal performance. A taxonomy of models aids in selecting suitable options and driving advancements. Classifying input–output formats enables leveraging diverse formats for customized systems, while evaluation metrics establish standardized methods to assess model quality and performance.

Keywords: generative AI survey; AIGC; AIGC models; ChatGPT; GPT-3; GPT-4; generative AI models; generative adversarial networks; transformers; user experience

Citation: Bandi, A.; Adapa, P.V.S.R.; Kuchi, Y.E.V.P.K. The Power of Generative AI: A Review of Requirements, Models, Input–Output Formats, Evaluation Metrics, and Challenges. *Future Internet* **2023**, *15*, 260. https://doi.org/10.3390/fi15080260

Academic Editor: Guan Gui

Received: 29 June 2023
Revised: 18 July 2023
Accepted: 26 July 2023
Published: 31 July 2023

1. Introduction

Generative artificial intelligence (AI) has emerged as a prominent field of study, revolutionizing various domains, such as computer vision, natural language processing, and creative arts. This research aims to delve into the fundamental aspects of generative AI, including requirements, models, generative types, and evaluation metrics, to gain a comprehensive understanding of this evolving discipline [1]. The field of generative AI focuses on developing algorithms and models capable of generating synthetic data that closely resemble real-world data. The ability to generate realistic and novel data has immense implications across multiple industries, including entertainment, healthcare, finance, etc. Generative AI has opened up new avenues for applications such as image synthesis, text generation, music composition, and even human-like chatbots [2].

The increasing availability of large-scale datasets, coupled with advancements in deep learning techniques, has propelled the rapid development of generative AI. The

ability to generate data that mimics real-world characteristics has the potential to address various challenges, including data augmentation, anomaly detection, and creative content generation. By understanding the requirements, models, generative types, and evaluation metrics within generative AI, researchers, and practitioners can make informed decisions when designing and implementing generative systems. Recent statistics highlight the growing interest and impact of generative AI. Precedence Research has reported that the worldwide market for generative AI was worth USD 10.79 billion in 2022. It is projected to reach approximately USD 118.06 billion by 2032, with a compound annual growth rate (CAGR) of 27.02% during the period from 2023 to 2032 [3]. This surge in market demand reflects the recognition of generative AI as a powerful tool with immense potential for various industries.

Generative AI encompasses a wide range of applications, and two famous examples that have made significant contributions to the field are StyleGAN and OpenAI's GPT. StyleGAN [4], developed by NVIDIA, has revolutionized image generation by producing highly realistic and diverse images. It employs a style-based approach, manipulating specific visual attributes and enabling artists to explore new dimensions of creativity in digital art. On the other hand, OpenAI's GPT series [5], with GPT-3 as its flagship model, has reshaped natural language processing. GPT-3's massive scale and transformer architecture enable it to generate human-like text with impressive fluency and coherence, making strides in tasks such as question answering, essay writing, and conversational responses. These examples exemplify the remarkable potential of generative AI in transforming creative industries, content generation, and human–machine interaction, paving the way for further advancements in image synthesis, text generation, and beyond.

Generative AI represents an integral part of the evolving landscape of Web 3.0, characterized by advancements in technology and user experiences. Table 1 compares the attributes of Web 1.0, Web 2.0, and Web 3.0 and provides a comprehensive overview of the evolution of the web, highlighting the significant changes in user interaction, content creation, technology, data management, communication, innovation, data access, computation resources, storage capacity, and examples from Web 1.0 to Web 3.0.

Table 1. Comparison of Web 1.0, Web 2.0, and Web 3.0.

Attribute	Web 1.0	Web 2.0	Web 3.0
Time line	1990s to 2002	2002 to 2022	2022 to 2042
User Interaction	Static websites with limited user interactivity.	Dynamic websites with enhanced user interactivity.	Intelligent and personalized user experiences.
Content Creation	Professional generated content (PGC), mainly generated by developers and content creators in the static websites.	User-generated content (UGC) became prominent in dynamic and interactive websites (blogging, social media).	Context-aware and intelligent content creation. User-generated and machine-generated content (AI-generated content (AIGC), smart devices).
Technology	Server-side processing using HTML, basic scripting, limited multimedia.	Client-side scripting with rich internet applications (JavaScript, Flash, AJAX).	Advanced technologies (AI, ML, NLP, blockchain, IoT).
Data Management	Centralized data storage and limited data sharing.	Decentralized data sharing and collaboration.	Distributed and interoperable data storage and sharing.

Table 1. *Cont.*

Attribute	Web 1.0	Web 2.0	Web 3.0
Communication	Basic email and discussion forums.	Social media platforms and instant messaging.	Advanced semantic communication and real-time collaboration. Seamless communication across platforms and devices.
Innovation	Limited innovation, focus on information.	Rapid innovation and collaboration.	Emphasis on AI, automation, and emerging technologies.
Computation resources	Limited computational and processing power on user devices. Focus on server-side computation.	Increased computational power on user devices. Focus on client-side computing.	Distributed computation and edge computing.
Storage capacity	Limited storage capacity on servers.	Increased cloud storage and scalable data storage solutions.	Expanded storage capacity. Distributed and scalable storage solutions.
Data access	Passive data consumption.	Active participation and data sharing.	Intelligent data access and personalized recommendations.
Examples	Early websites, static informational pages.	Social media platforms (Facebook, Twitter), blogging platforms.	Virtual collaborative assistants, AI-powered applications, blockchain platforms.

1.1. Research Purpose

The purpose of the research is to comprehensively analyze the requirements, models, generative types, and evaluation metrics in generative AI. By exploring the existing literature and synthesizing research findings, this study aims to contribute valuable insights to the field and provide guidance for researchers, practitioners, and enthusiasts. It investigates the fundamental aspects of generative AI systems, including requirements, models, input–output formats, and evaluation metrics. Categorizing implementation requirements into hardware, software, and user experience, the research presents a taxonomy of generative AI models based on architectural characteristics and explores various model types. Additionally, it proposes a classification system for input and output formats and discusses commonly used evaluation metrics. The ultimate goal is to advance the field by facilitating effective implementation and evaluation of generative AI models for diverse applications.

1.2. Research Questions

The motivation for this research stems from the growing importance of generative AI systems and the need for a deeper understanding of their fundamental aspects. By investigating the requirements necessary for implementing generative AI systems, exploring the different types of generative AI models, identifying specific input and output formats, and discussing commonly used evaluation metrics, we aim to provide researchers, developers, and practitioners with comprehensive insights and guidance in the field. The goal is to enhance the implementation, selection, customization, and evaluation of generative AI models for various applications, ultimately fostering advancements in generative AI.

RQ1 What are the requirements necessary for implementing generative AI systems?

Ans: To address this question, we present three distinct categories of requirements for AIGC: hardware, software, and user experience requirements.

RQ2 What are the different types of generative AI models described in the literature?

Ans: To explore this, we present a taxonomy of AIGC models based on their architecture, including VAEs, GANs, diffusion models, transformers, language models, normalizing flow models, and hybrid models.

RQ3 What specific input and output formats are used for different prescribed tasks in generative AI systems?

Ans: We provide a comprehensive classification of input and output formats for AIGC tasks, along with specific tasks and the corresponding models used in the literature, presented in a tabular format.

RQ4 What evaluation metrics are commonly employed to validate the output generated by generative AI models?

Ans: We propose a classification system based on the output types of generative AI and discuss commonly used evaluation metrics in the field.

1.3. Contributions and Research Significance

The research questions and corresponding answers presented in this study have significant implications for generative AI. Firstly, by investigating the requirements necessary for implementing generative AI systems, encompassing hardware, software, and user experience considerations, this research provides valuable insights for researchers, developers, and practitioners. Understanding these requirements is essential for effective planning and designing generative AI systems, ensuring their successful implementation and optimal performance. Additionally, exploring different types of generative AI models described in the literature contributes to a comprehensive understanding of the field. By presenting a taxonomy of models based on their architectural characteristics, such as VAEs, GANs, diffusion models, transformers, language models, normalizing flow models, and hybrid models, this research guides researchers and practitioners in selecting appropriate models for specific applications. This knowledge fosters advancements in generative AI by promoting informed decision-making and facilitating the development of more sophisticated and tailored models.

Moreover, this study delves into the specific input and output formats used for different prescribed tasks in generative AI systems. The comprehensive classification of these formats, along with the associated tasks and models utilized in the literature, is a valuable resource for researchers and practitioners. It enables them to understand and leverage the diverse input and output formats available, promoting the development of effective and customized generative AI systems. Furthermore, the proposal of a classification system based on output types and the discussion of commonly used evaluation metrics contribute to the establishment of robust evaluation frameworks. These frameworks enhance the credibility and applicability of generative AI models by providing standardized methods for assessing their quality and performance. In summary, this research offers valuable insights into generative AI systems' requirements, models, input–output formats, and evaluation metrics. These findings contribute to advancements in the field, guiding researchers, developers, and practitioners in effectively implementing and evaluating generative AI models for various applications.

1.4. Organization of the Paper/Reading Map

The structure of the paper is given in Figure 1.

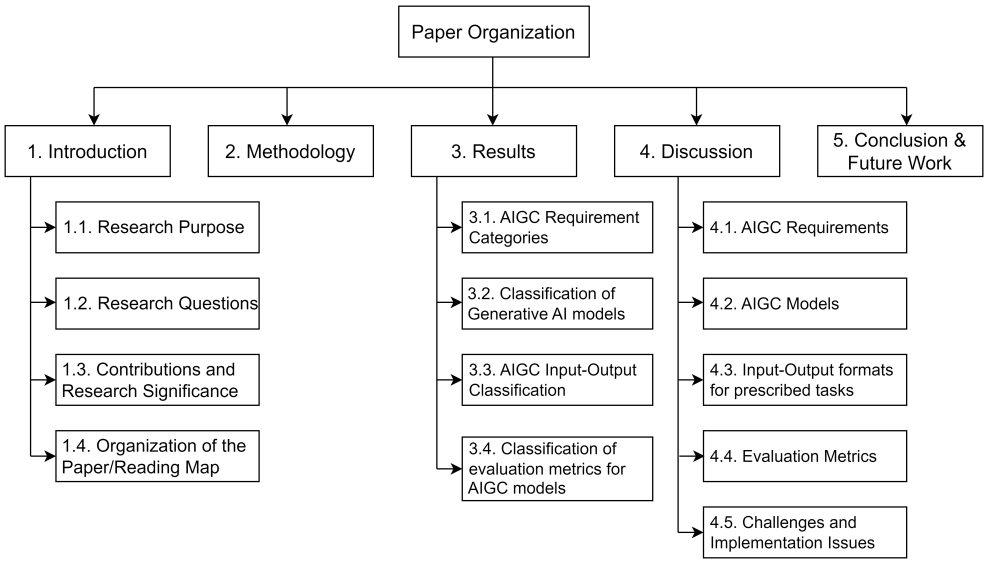

Figure 1. Organization of this review paper.

2. Methodology

To conduct our research, we conducted an extensive literature search across multiple scholarly databases, including Google Scholar, Semantic Scholar, ACM Digital Library, and IEEE Xplore. Our search was based on a carefully crafted search string "("Generative AI" OR "AIGC" OR "Generative Adversarial Networks" OR "GANs" OR "Generative Models") AND ("survey" OR "review" OR "overview" OR "summary" OR "literature review")" designed to capture relevant studies pertaining to our research questions. We meticulously reviewed the search results and included studies that addressed our research inquiries directly. As part of our inclusion criteria, we focused on papers published in English, ensuring consistency in language and accessibility. We also took measures to eliminate duplicate papers, carefully screened the abstracts and conclusions of the selected papers, and evaluated their suitability based on qualitative or quantitative analysis, surveys, or review papers. Additionally, we identified valuable secondary references through survey papers to augment our research findings. Subsequently, we extracted and synthesized pertinent data attributes aligned with our research questions, enabling us to present comprehensive and insightful results.

The search process results are as follows: The publication years range from 2014 to the present, with 90% of the papers published from 2017 to the present. The year-wise publication count is given in Table A1. This indicates that more relevant papers have been published in recent times. We have extracted data from 122 full-text papers, which consist of peer-reviewed publications in conferences and journals and accepted articles archived in the arXiv database. Among these articles, 49% are from conference proceedings, while 51% are from journals and archives. The distribution of the publication type is given in Table A2. Notably, we found 15 papers at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), followed by ten papers from the International Conference on Machine Learning and eight papers published in the Advances in Neural Information Processing Systems and seven papers in IEEE International Conference on Computer Vision (ICCV). The remaining papers are in the following order: European Conference on Computer Vision, International Conference on Multimedia, International Conference on Learning Representations (ICLR), International Conference on Neural Information Processing Systems, Conference on Empirical Methods in Natural Language Processing, etc. The other venues are listed in Table A3.

Additionally, we identified 29 survey or literature review papers apart from the aforementioned papers. We have separated the survey papers and analyzed them based on our research questions, comparing their contributions with our own in this paper. We

have labeled the contributions as L, M, H, or NA, representing low, medium, high, and not applicable, respectively. We observed that the majority of the surveys did not explicitly focus on the requirements and evaluation metrics of AIGC. We summarized the important survey papers in Table 2 and compared them with our paper.

The table provides a comprehensive overview of various papers and their contributions to the field of generative AI. It includes papers published between 2017 and 2023, as well as the contributions of each paper in terms of AIGC requirements taxonomy, AIGC models, AIGC input–output classification, and AIGC evaluation metric classification. Most papers reviewed (labeled with "L") focus on different aspects of AIGC, such as the applications of GANs across various domains and the progress made in computer vision using GANs. Some papers (labeled with "M" or "H") provide more in-depth analyses of AIGC models and their variants, particularly in the fields of drug discovery, material science, and speech synthesis. A few papers (labeled with "NA") either do not contribute significantly to the specified categories or focus on broader opportunities and impacts of AIGC in business, education, and society. Our paper, labeled "this paper", is distinguished by its comprehensive coverage of all four areas of AIGC: AIGC requirement categorization, AIGC models, input–output classification, and evaluation metrics. It highlights the significant contributions made by the authors in these areas.

Table 2. Summary of important surveys on AIGC (L—Low, M—Medium, H—High, NA—Not Applicable).

Reference	Year Published	AIGC Requirements Taxonomy	AIGC Models	AIGC Input–Output Classification	AIGC Evaluation Metric Classification	Remarks
[6]	2017	L	M	M	NA	A review of generative adversarial networks (GANs) and their applications across various domains.
[7]	2019	L	M	M	M	This highlights the background, evaluation metrics, and training processes of GANs.
[8]	2019	NA	M	M	L	A review of GANs, including comparisons, performance evaluations, and their applications in computer vision (CV).
[9]	2020	NA	M	M	L	A review of GANs, their training processes, evaluation indices, and applications in CV and NLP.
[10]	2020	NA	L	L	NA	Overview of GAN architecture, along with the state of the art in the security domain.
[11]	2020	L	H	L	M	Presented overview of generative AI model classifications.
[12]	2020	NA	L	L	NA	Focuses on the application of GANs in architecture and urban design.
[13]	2020	L	M	M	NA	A review of the progress made with GANs and their applications in computer vision (CV).
[14]	2021	L	L	M	NA	This highlights the various applications of GANs and their impact across different domains.
[15]	2021	L	M	L	L	Focusing on the application of GANs in finance research.
[16]	2021	L	H	H	L	A survey of GANs and their variants across various research fields.

Table 2. *Cont.*

Reference	Year Published	AIGC Requirements Taxonomy	AIGC Models	AIGC Input–Output Classification	AIGC Evaluation Metric Classification	Remarks
[17]	2021	L	L	L	L	An overview of GANs in the field of digital pathology.
[18]	2021	L	L	L	M	A survey on GANs for NLP tasks, including available datasets and evaluation metrics.
[19]	2021	L	H	L	M	Focuses on AIGC models for drug discovery.
[20]	2022	L	M	L	M	An overview of the main enhancements, variations of GAN models, and their evaluation metrics.
[21]	2022	L	M	L	NA	A survey on generative AI models used in drug discovery.
[22]	2023	L	L	L	NA	A comprehensive survey on the underlying technology and applications of text-to-3D conversion.
[1]	2023	H	H	H	L	An overview of the history, components, recent advances, and challenges in AI-guided combinatorial chemistry.
[23]	2023	NA	NA	NA	NA	Focuses on the opportunities that AIGC presents in business, education, and society.
[24]	2023	NA	L	NA	L	Generative AI models in the domain of computer-aided drug design.
[25]	2023	L	M	H	NA	Focusing on applications that involve input–output classification.
[26]	2023	L	M	H	L	Focusing on applications that involve input–output classification.
[27]	2023	L	H	L	L	Progress of AIGC models, their challenges, and applications in material science.
[28]	2023	L	L	L	NA	A review of ChatGPT's role across various research fields.
[29]	2023	L	M	M	M	Focuses on AIGC models in molecule, protein, and material science.
[30]	2023	L	M	L	NA	Focuses on ChatGPT, its underlying architecture, and its applications across various domains.
[31]	2023	L	L	M	M	A survey on recent progress of AIGC models in speech synthesis.
[32]	2023	M	L	M	M	A review of text-to-image diffusion models.
[2]	2023	M	H	M	L	A review of AIGC models across various research fields.
This paper	2023	H	H	H	H	Our paper focuses on all four areas and contributes significant results such as AIGC requirements categorization, AIGC models, input–output classification, and evaluation metrics.

3. Results

3.1. AIGC Requirement Categories

This section begins explaining the implementation phases of generative AI models, such as variational autoencoders (VAEs), generative adversarial networks (GANs), and transformers, have emerged as powerful tools for generating synthetic data samples. However, the process of building generative AI models requires various stages that must be addressed in a systematic way to obtain the required results. While the exact terminologies and steps may vary depending on the specific approach and context, as illustrated in Figure 2, the common phases involved in generative AI are problem definition, data collection and preprocessing, model selection, model training, model evaluation, model fine-tuning, deployment, and monitoring and maintenance.

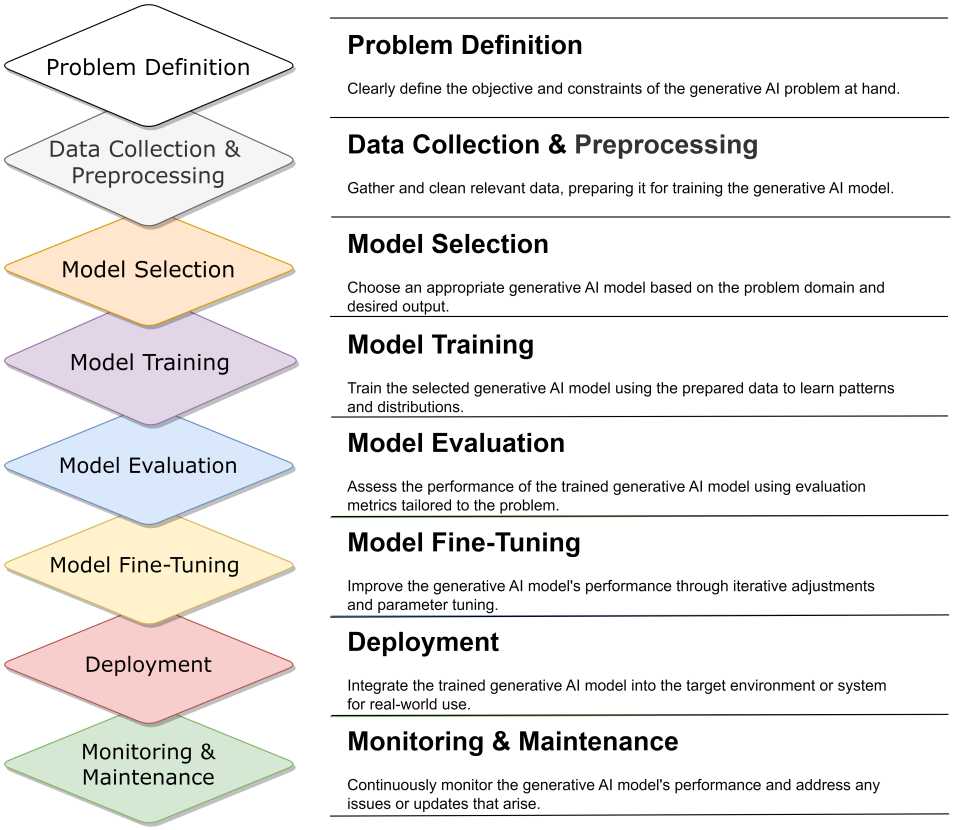

Figure 2. Implementation phases of generative AI.

The initial phase revolves around clearly defining the problem that the generative AI model aims to address. This encompasses identifying the desired outcomes, data requirements, and any constraints or limitations, thus establishing a solid foundation for subsequent stages. Accurate problem definition facilitates targeted data collection and effective model selection, streamlining the overall implementation process. The subsequent phase focuses on data collection, which entails gathering a large and representative dataset that encapsulates the patterns and characteristics that the generative model intends to learn using appropriate devices for capturing data, such as web scraping tools [33–35], microphones, cameras [36], or sensors. There are various datasets from researchers to facilitate research and benchmarking for various types of data: images—MSCOCO [37], Flickr [38]; text—Colossal Clean Crawled Corpus (C4) [39]; or audio—FSD50K [40], AudioCaps [41]. Data should be diverse and comprehensive to capture the underlying patterns effectively. Following data collection, the model selection phase begins, where the most suitable generative model architecture is chosen. This phase involves considering popular options such as VAEs, GANs, transformers, or diffusion models. The selection of an appropriate model architecture aligns with the problem requirements and paves the way for subsequent steps.

Once the generative model architecture is determined, the model training phase commences. This stage involves training the selected model using the collected or available dataset. Through this process, the model learns the underlying patterns and statistical relationships within the data. Training generative models frequently necessitates substantial computing resources, particularly for large-scale datasets and sophisticated models. To expedite training, high-performance hardware like graphics processing units (GPUs) [42] or tensor processing units (TPUs) [43] are typically employed. The specific training algorithm varies depending on the chosen model, with GANs, for instance, training a generator network to produce realistic samples while concurrently training a discriminator network to differentiate between real and generated samples [44]. Hyperparameter tuning constitutes a significant aspect of the model training phase. Various hyperparameters, including

learning rate, batch size, network architecture, and regularization techniques, influence the behavior and performance of the generative model [45]. Fine-tuning these hyperparameters is crucial for optimizing the model's convergence and overall performance [33]. Once the model training is completed, the subsequent phase involves evaluating and validating its performance. Evaluation metrics are tailored to the specific task or domain. In the case of image generation, metrics such as inception score, Frechet inception distance (FID) [46], or visual inspection can be utilized to assess the quality and diversity of the generated samples. Post-processing and refinement may be necessary in certain cases to enhance the quality or adhere to specific constraints of the generated outputs. Techniques such as image smoothing, text correction, or style transfer can be employed in this phase to refine the generated samples based on domain-specific requirements. Upon successful training and validation, the generative model is ready for deployment to generate new samples. This phase involves providing an input to the model, such as a random noise vector or a partial input and obtaining an output that aligns with the distribution of the training data. Multiple samples can be generated to explore the variety and creativity exhibited by the generative model.

The rest explores the essential requirements for generative AI, focusing on hardware, software, and user aspects. The categories of AIGC requirments are shown in Table 3. When it comes to hardware, the collection of data for generative AI tasks involves leveraging cameras, microphones, sensors, and existing datasets curated by researchers for specific purposes. For the training, fine-tuning, and hyperparameter optimization stages, powerful hardware configurations like Tesla V100 16 GB, RTX 2080Ti, NVIDIA RTX 3090 with 24 GB, and TPUs are commonly employed. However, for smaller-scale models, a GTX 1060 6 GB of DDR5 can suffice. Sample generation, which is an integral part of the generative AI process, can be achieved using more basic configurations like a CPU with an i7 3.4 GHz clock speed and a GPU such as the GTX970. On the software side, various tools and frameworks play a crucial role in different phases of generative AI. Data collection and preprocessing rely on frameworks like web scraping frameworks, Pandas, Numpy, scikit-image, torch-audio, torchtext, and RDKit. Additionally, specialized tools for data acquisition, audio recording, and motion capture are employed. To train generative models effectively, deep learning frameworks, such as TensorFlow, PyTorch, scikit-learn, and SciPy, provide comprehensive support for various model architectures and optimization algorithms. These frameworks are also instrumental in evaluating and validating the models. Furthermore, post-processing and model refinement can be facilitated using libraries like OpenCV-Python and NLTK. By understanding and fulfilling these hardware and software requirements, researchers and practitioners are well-equipped to delve into generative AI research and development. These requirements lay the foundation for creating sophisticated and high-quality generative models, enabling advancements in this exciting field of artificial intelligence.

User experience requirements for generative AI models described in Table 3 are critical in ensuring user satisfaction and successful outcomes. High-quality and realistic outputs are expected, along with customization and control options to align the generated content with user preferences. Diversity and novelty in outputs, as well as performance and efficiency, are important considerations. Interactivity and responsiveness to user input, along with ethical considerations, such as fairness and data privacy, are significant requirements. Seamless integration with existing systems and compatibility with programming languages are also valued for easy adoption. By addressing these requirements, developers and researchers can create generative AI models that meet user expectations and deliver enhanced experiences.

Table 3. AIGC requirement categories.

Category	Description
Hardware requirements	Data can be collected using cameras [36], microphones, sensors and can use datasets that are released by researchers for specific tasks [37–39]. To train, fine-tune and optimize hyperparameters—Tesla V100 16 GB [47], RTX 2080Ti [48], NVIDIA RTX 2080Ti, NVIDIA GeForce RTX 3090 with 24 GB [49], TPU [50], etc., are generally used, while GTX 1060 6 GB of DDR5 [51] can also be used to train a small-scale model. Sample generation can be performed on basic configuration like CPU—i7 3.4 GHz and GPU—GTX970 [52] or the configuration required by the generative model.
Software requirements	For data collection and preprocessing, tools such as Web scraping frameworks [33–35], Pandas [48,52,53], Numpy [53–55], scikit-image [48,55,56], torch-audio, torchtext [48], RDKit [57], data acquisition tools, audio recording software, motion capture software. To train the models, deep learning frameworks like TensorFlow [52,58,59], PyTorch [60–62], scikit-learn [52,60,63], SciPy [53,63] are used, which provide support for various generative model architectures and optimization algorithms. PyTorch [64], TensorFlow [65], scikit-learn [60]: these libraries are also used to evaluate and validate the model. For post-processing and refinement of models, libraries like opencv_python [55,66], NLTK [59,67] are used.
User-experience requirements	Key considerations for user aspects are high quality, accuracy [68] and realistic outputs [69], customization and control, diversity [70] and novelty, performance and efficiency, interactivity and responsiveness to user input, ethics [71] and data privacy [72,73], and seamless integration with existing systems.

3.2. Classification of Generative AI Models

Generative AI Model Architecture: This is the model's basic structure or design. It includes how its layers or neural networks and components are arranged and organized. The model's architecture determines how it processes and generates information, which makes it a critical aspect of its functionality and suitable for specific tasks. Table 4 describes the architecture components and training methods that are used in the generative AI models.

The classification of generative models based on architecture provides insights into the specific components and training methods that define each model as shown in Figure 3. These architectural choices have significant implications for how the models generate new data points and learn from the available data. By understanding these distinctions, researchers and practitioners can choose the most suitable generative model for their specific task or explore hybrid approaches that combine different models to leverage their respective strengths. Variational autoencoders (VAEs) have an encoder–decoder architecture and use variational inference for training. They learn compressed representations of input data and generate new samples by sampling from the learned latent space. Generative adversarial networks (GANs) consist of a generator and a discriminator. They are trained adversarially, with the generator generating synthetic samples to fool the discriminator. GANs excel at generating realistic and diverse data. Diffusion models involve a noising step followed by a denoising step. They iteratively refine noisy inputs to generate high-quality samples. Training involves learning the dynamics of the diffusion process. Transformers employ an encoder–decoder architecture and utilize self-attention mechanisms for capturing global dependencies. They are commonly used in tasks like machine translation and generate coherent sequences through supervised training. Language models, often based on recurrent neural networks (RNNs), generate sequences by predicting the next token. They are trained through supervised learning and excel at generating natural language sequences. Normalizing flow models use coupling layers to transform data while preserving density. They learn complex distributions by transforming a simple base distribution, trained via maximum-likelihood estimation. Hybrid models combine different architectures and training methods to leverage their respective strengths. They offer flexibility and tailored generative capabilities by integrating elements from multiple models.

Table 4. Architecture components and training methods used in generative AI models.

Model	Architecture Components	Training Method
Variational Autoencoders	Encoder–Decoder	Variational Inference [19]
Generative Adversarial Networks	Generator–Discriminator	Adversarial [44]
Diffusion Models	Noising (Forward)–Denoising	Iterative Refinement [31]
Transformers	Encoder–Decoder	Supervised [74]
Language Models	Recurrent Neural Networks	Supervised [75]
Normalizing Flow Models	Coupling Layers	Maximum-Likelihood Estimation [76]
Hybrid Models	Combination of Different Models	Varied

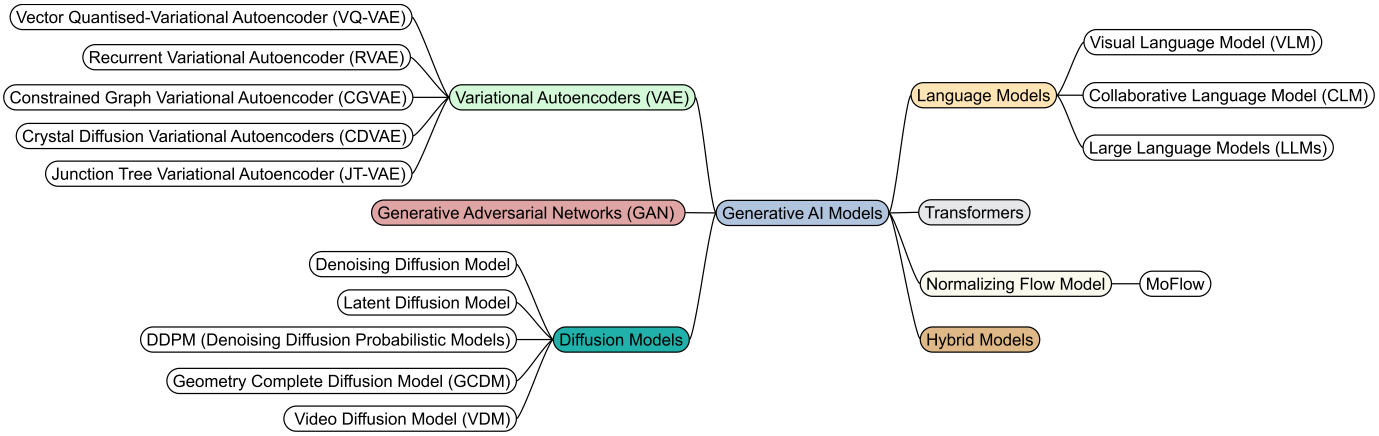

Figure 3. Classification of the generative AI models based on the architecture.

3.2.1. Variational Autoencoders (VAE)

A variational autoencoder (VAE) is a type of autoencoder that combines variational inference with an encoder–decoder architecture. Autoencoders consist of an encoder network that maps high-dimensional data to a low-dimensional representation and a decoder network that reconstructs the original input from the representation [19]. However, traditional autoencoders lack the ability to generate new data points.

In Figure 4, in a VAE, the encoder network maps the input data (x) to the parameters of a probability distribution in a latent space (z) using input layer and hidden layer composed of neural network units, such as dense or convolutional layers. This distribution is often modeled as a multivariate Gaussian with mean and covariance parameters [27] achieved in mean, variance layers. Samples are drawn from this latent space distribution in sampling layer, generated by the encoder, to produce new data points using the decoder network (y) with hidden and output layers. By sampling from the approximate posterior distribution in the latent space, VAEs can generate diverse outputs resembling the training data.

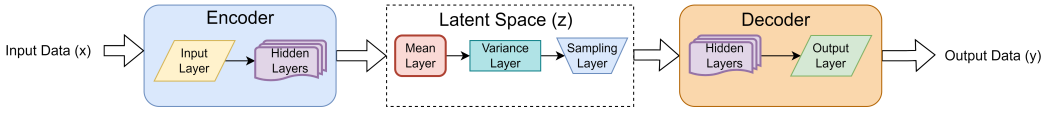

Figure 4. Typical structure of variational autoencoder (VAE).

Neural networks, such as fully connected networks or convolutional neural networks (CNNs), are commonly used as encoders and decoders in VAEs. The specific architecture

depends on the data and its complexity. For images or grid-like data, CNNs or deconvolutional neural networks (also known as convolutional transpose networks) are often employed as decoders. Training a VAE involves optimizing the model parameters by minimizing a loss function comprising a reconstruction loss and a regularization term. The reconstruction loss measures the discrepancy between the input data and the reconstructed output, while the regularization term computes the Kullback–Leibler (KL) divergence between the approximate posterior distribution and a chosen prior distribution in the latent space. This term promotes smoothness and regularization. The training process of a VAE includes selecting the network architecture, defining the loss function, and iterating through batches of input data. The encoder processes the data, latent space points are sampled, and the decoder reconstructs the input. The total loss, combining the reconstruction loss and regularization term, is computed, and gradients are used to update the model parameters through backpropagation.

While VAEs offer generative modeling and capture complex latent representations, they may suffer from issues such as blurry reconstructions and challenges in evaluating the quality of generated samples. Researchers have proposed various improvements to VAEs to address these concerns, such as vector-quantized variational autoencoder (VQ-VAE) [77], which introduces a discrete latent space by quantizing encoder outputs, leading to a more structured and interpretable latent representation; recurrent variational autoencoder (RVAE) [78] to sequential data by incorporating recurrent architectures, allowing for sequence generation and anomaly detection; constrained graph variational autoencoder (CGVAE) [79] models graph-structured data using graph neural networks, enabling generation while preserving structural properties; crystal diffusion variational autoencoders (CDVAE) [80], generating crystal structures in materials science, combining VAE with a diffusion process to learn a continuous representation of crystal structures; junction tree variational autoencoder (JT-VAE) [57], leveraging junction trees, a type of graphical model, to capture complex dependencies between variables in structured domains like natural language processing or bioinformatics.

3.2.2. Normalizing Flow Models

Normalizing flow models are deterministic and invertible transformations between the raw data space and the latent space [21]. Unlike other generative models such as GANs or VAEs, which introduce latent variables and transform them to generate new content, normalizing flow models directly solve the mapping transformation between two distributions by manipulating the Jacobian determinant [27]. In Figure 5, normalizing flow applies a sequence of invertible transformations to a simple probability distribution (z) to model more complex probability distributions using an affine coupling layer in the encoder (flow). The decoding (inverse) function is designed to be the exact inverse of the encoding function using same affine coupling layers and quick to calculate, giving normalizing flows the property of tractability [29].

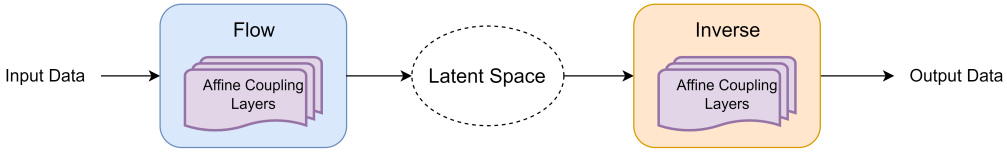

Figure 5. Typical structure of normalizing flow model.

Coupling layers play a crucial role in normalizing flow models. They are used to perform reversible transformations on the input data and latent variables. Affine coupling transformations, a specific type of coupling layer, are commonly used in normalizing flows. These transformations model complex relationships between variables while maintaining invertibility. By using element-wise multiplication and addition, the Jacobian determinant can be efficiently computed. In a coupling layer, the input data are split into fixed and transformed parts. The fixed part is typically passed through unchanged, while the

transformed part undergoes a transformation based on a function of the fixed part. This approach allows the model to focus on modeling complex relationships while preserving certain aspects of the input.

The design of an invertible function with expressive structures and efficient computation of the Jacobian determinant is a challenging task in normalizing flows. Affine coupling transformations address these challenges by providing a flexible and efficient way to model complex relationships and compute the Jacobian determinant [76]. By applying a sequence of invertible transformations, normalizing flows can model complex probability distributions. These transformations are designed to be reversible, allowing for tractable likelihood computation. The encoder–decoder functions in normalizing flows are exact inverses of each other, enabling efficient calculations and maintaining tractability.

Normalizing flows offer the advantage of providing an exact likelihood evaluation and efficient sampling from complex probability distributions, enabling flexible generative modeling. However, a drawback of normalizing flows is the computational expense associated with training deep architectures, particularly for large-scale datasets. Additionally, achieving satisfactory performance may necessitate a significant amount of data during the training process. MoFlow, a flow-based graph generative model to learn invertible mappings between molecular graphs and their latent representations [76]. In MoFlow, each component flow in the mixture is responsible for capturing different aspects of the data distribution. By combining these flows, MoFlow can better model diverse samples from complex data distributions. The mixture of flows can be learned using a gating mechanism that assigns weights or probabilities to each component flow, allowing the model to dynamically select the most appropriate flow for each input sample.

3.2.3. Generative Adversarial Networks (GAN)

Generative adversarial networks or GANs were first introduced by Ian Goodfellow in 2014 [44]. The GAN is based on the minimax two-person zero-sum game, in which one player profits only when the other suffers an equal loss. The two players in GAN are the generator and the discriminator. The generator's purpose is to trick the discriminator, while the discriminator's goal is to identify whether a sample is from a true distribution. The discriminator's output is a probability that the input sample is a true sample. A higher probability suggests that the sample is drawn from real-world data. In contrast, the closer the probability is to zero, the more probable the sample is a fake. When the probability approaches one-half infinity, the optimal answer is reached because the discriminator finds it difficult to check fake samples [7].

Typically, generator (G) and discriminator (D) are implemented using deep neural networks, working as latent function representations. The architecture of the GAN, illustrated in Figure 6, involves the G learning the data distribution from real samples and mapping it to a new space (generated samples) using dense/convolutional layers accompanied by its corresponding probability distribution. The primary objective of the GAN is to ensure that this probability distribution closely resembles the distribution of the training samples. The D receives input data, which can be either real data (x) from the training set or generated data produced by the generator. The discriminator then outputs a probability using dense/convolutional layers or scalar value that indicates whether the input is likely to come from the real data distribution.

GAN (generative adversarial network) training faces several challenges, including gradient disappearance, difficulty in training, and poor diversity. These problems arise from the loss function used in GANs, which involves measuring and minimizing the distance between the real data distribution (Pr) and the generated data distribution (Pg).

During training, the discriminator aims to minimize cross-entropy by differentiating between real and generated samples. The optimal discriminator (D) takes the form given below.

$$D(x) = Pr(x)/(Pr(x) + Pg(x))$$

On the other hand, the generator (G) seeks to minimize a generator-specific loss function that includes an independent item to ensure diversity.

The loss function for the generator can be written as,

$$V(G) = KL(Pg||Pr) - 2JSD(Pr||Pg)$$

where KL is the Kullback–Leibler divergence and JSD is the Jensen–Shannon divergence. Minimizing the JS divergence helps the generated samples resemble real ones. However, if there is little or no overlap between Pr and Pg, the JS divergence becomes a constant, leading to gradient vanishing and disappearance [8].

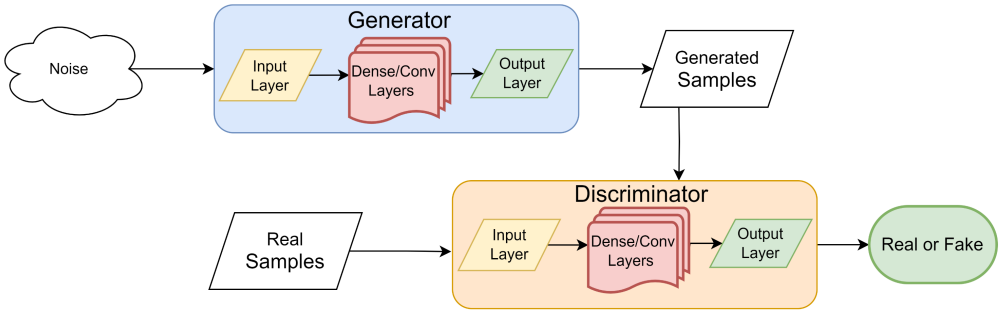

Figure 6. Typical structure of generative adversarial networks (GAN).

Additionally, training GANs can be challenging because the feedback from the discriminator can be close to zero when it is trained optimally, slowing down convergence. Moreover, determining when the discriminator is properly trained is difficult since there is no indicator for it.

Another problem is the poor diversity in the generated samples. The generator loss function V(G) can be reformulated to address this issue. Minimizing this loss function is equivalent to minimizing the KL divergence and the JSD, leading to more diverse generated samples. Several new models have been introduced to address these limitations of the original GAN, including issues like gradient disappearance, unstable training, and poor diversity. These new GAN models aim to enhance stability and improve the quality of the generated outputs.

Conditional generative adversarial networks (CGANs) have emerged as a solution to enhance the control and convergence speed of GANs in complex or large-scale datasets. By incorporating conditional variables, such as category labels, textual descriptions, or specific generated targets, CGANs provide guidance to the data generation process. This allows for supervised learning, targeted generation, and the ability to generate images with specific categories or labels. Moreover, CGANs can utilize image features as condition to generate corresponding word vectors, enabling effective cross-modal generation illustrated in Figure 7.

Some of the GANs that incorporate this technique are conditional generative adversarial networks (CGAN) [81], CGAN with Pix2Pix framework [82], conditional tabular GAN (CTGAN) [83], conditional generative adversarial networks with text (TAC-GAN, TAGAN) [67,84].

Wasserstein generative adversarial networks (WGANs) offer a novel approach to address the challenges faced by traditional GANs. By introducing the Wasserstein distance as a metric, WGANs provide a more stable training process and better gradient flow. The discriminator in WGANs, known as the "critic", assigns scores representing the distance between the real and fake data distributions [20]. This distance is measured using the Wasserstein distance instead of the Jensen–Shannon divergence or Kullback–Leibler divergence used in other generative models. WGANs mitigate the issue of mode collapse, where GANs fail to capture the full diversity of the data, by effectively learning the underlying data distribution, even for complex and high-dimensional datasets [85]. The generator and

discriminator in WGANs are trained to minimize the Wasserstein distance, encouraging the generator to generate samples that closely resemble real data. This enables WGANs to produce more diverse and realistic outputs.

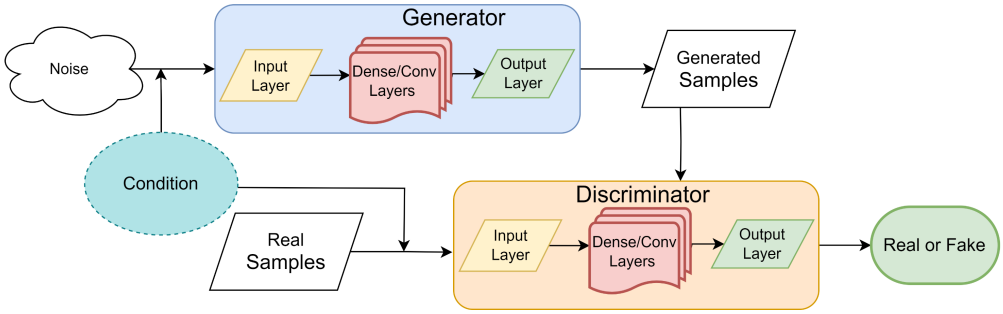

Figure 7. Typical structure of conditional GAN.

WGANs have found applications in various domains, such as image synthesis, text generation, and data augmentation. Their effectiveness in addressing mode collapse and providing a more reliable training process has made them a popular choice for researchers and practitioners working with generative models.

Deep convolutional generative adversarial networks (DCGANs) are a variant of GANs that leverage deep convolutional neural networks (CNNs) to enhance the quality of generated samples, particularly in the domain of image synthesis. DCGANs have proven to be highly effective in generating realistic and high-resolution images [86]. DCGANs utilize convolutional layers in both the generator and discriminator networks, allowing them to capture spatial dependencies and patterns in the data. DCGANs introduce several key design principles, including the use of convolutional and transposed convolutional layers, batch normalization, and ReLU activation functions. These principles contribute to the stability of the training process, mitigate issues like mode collapse and allow for the generation of diverse and high-quality samples.

The benefits of DCGANs extend beyond image synthesis, with applications in areas such as image-to-image translation, style transfer, and data augmentation. The combination of deep convolutional architectures and adversarial training has propelled DCGANs as a go-to choice for generating visually appealing and realistic images in the field of deep learning.

Generative adversarial networks (GANs) have revolutionized various domains of computer vision and machine learning. They can be classified into different categories based on their specific tasks and applications. Image-to-image translation GANs focus on translating images between domains, with subcategories such as CycleGAN [87], DiscoGAN [88], and DTN [89]. Super-resolution GANs [90] enhance the resolution of low-resolution images, including SRGAN and VSRResFeatGAN [91]. Text-to-image GANs generate images from textual descriptions, exemplified by AttnGAN [92] and StackGAN [93]. Tabular data GANs generate synthetic tabular data, with examples like CTGAN [83] and TGAN [94]. Defense and security GANs address security-related applications, including defense against adversarial attacks and steganography, such as defense GANs [95] and SSGAN [96]. Style-based GANs capture and manipulate artistic styles, including StyleGAN [97] and StyleCLIP [98]. Other GAN types encompass diverse applications like BigGAN [43] for high-resolution images, ExGANs [99] for variation generation, and SegAN [100] for semantic segmentation and various other GANs and are listed below. These categories demonstrate the versatility and advancement of GANs in various domains, enabling tasks such as image translation, super-resolution, text-to-image synthesis, data generation, security applications, style manipulation, and more. GANs continue to drive innovation and push the boundaries of generative models in the field of artificial intelligence.

3.2.4. Diffusion Models

Diffusion models are a type of generative model that operates by progressively introducing noise into data until it conforms to a desired distribution. The main idea behind diffusion models is to learn the process of reversing this diffusion, enabling the generation of valid samples [31]. In Figure 8 the forward pass of a diffusion model, Gaussian noise is iteratively added to the data in a series of steps. This noise corrupts the original data, gradually degrading its quality. As the noise level increases with each step, the images become increasingly distorted or destroyed. The objective of the diffusion model is to learn the dynamics of this diffusion process. By observing the corrupted data and the corresponding noise levels, the model learns to estimate the conditional probability distribution that describes the relationship between the corrupted data and the noise levels. Once the diffusion process is learned, the model can then perform the reverse pass, starting from the corrupted data and progressively removing the noise in each step [32]. This process of denoising leads to the generation of valid and realistic samples that resemble the original data distribution.

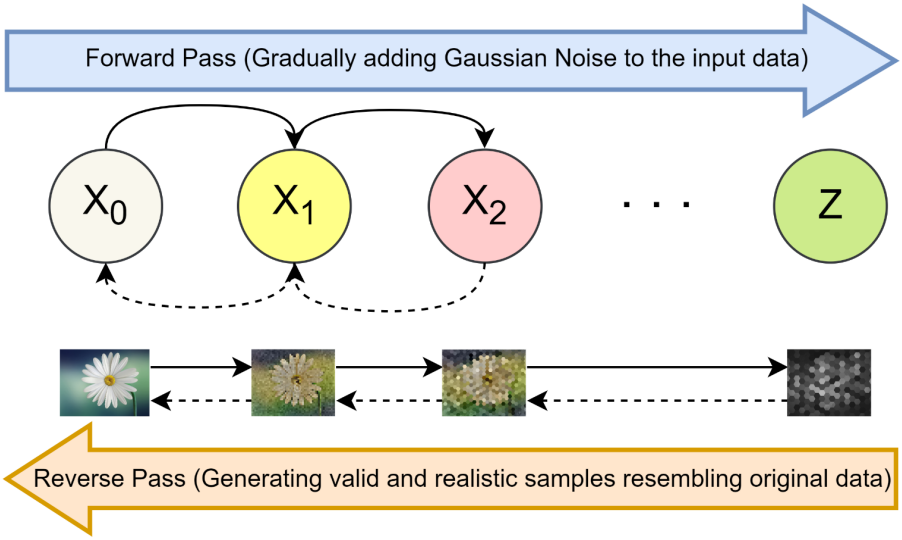

Figure 8. Typical structure of diffusion model.

There are three sub-types that differ in their implementation of the forward and backward diffusion pass. These sub-types are denoising diffusion probabilistic models (DDPMs), score-based generative models (SGMs), and stochastic differential equations (SDEs) [29].

Denoising Diffusion Probabilistic Models (DDPMs): DDPMs, also known as denoising score-matching models, incorporate a two-step process for diffusion [101]. They apply Markov chains to progressively corrupt data with Gaussian noise and then reverse the forward diffusion process by learning Markov transition kernels. DDPMs focus on modeling the diffusion process and its associated reversibility.

Score-based Generative Models (SGMs): SGMs, also referred to as score-matching models, work directly with the gradient of the log density (score function) of the data [32]. They perturb the data with noise at multiple scales and jointly estimate the score function of all noisy data distributions using a neural network conditioned on different noise levels. This decoupling of training and inference steps enables flexible sampling.

Stochastic Differential Equations (SDEs): SDEs generalize diffusion models into continuous settings. They formulate noise perturbations and denoising processes as solutions to stochastic differential equations [1]. By leveraging the probabilistic flow of these equations, the reverse generation process can be modeled. Probability flow ordinary differential equations (ODEs) can also be utilized to represent the reverse process.

Diffusion models employ neural network architectures to capture the complex dependencies and patterns in the data. These architectures can consist of various layers, such as convolutional layers for image data or recurrent layers for sequential data. The network is trained to learn the conditional probability distribution that describes the relationship between the corrupted data and the noise levels. The training objective of diffusion models is typically based on maximum-likelihood estimation or other probabilistic frameworks. The model parameters are optimized to minimize the discrepancy between the generated samples and the original data distribution. Various techniques such as gradient descent and backpropagation are employed to train the model effectively.

Diffusion models, such as the deep diffusion generative models (DDGM), have gained prominence as strong generative models in recent years. They take a novel technique to modeling complicated data distributions by diffusing a given input iteratively towards a target distribution. However, to address specific difficulties or increase performance in specific scenarios, variations in diffusion models are necessary. The latent diffusion model (LDM) [102] is a variant of the diffusion model that operates in latent space. It is a generative model that aims to learn the underlying data distribution by applying a diffusion process to the latent variables instead of the observed data. The latent diffusion model can develop more meaningful representations and capture the underlying structure of the data distribution by acting in latent space [103]. It enables the efficient and effective generation of high-quality samples with desired attributes. The latent diffusion model has been used to produce varied and realistic samples in a variety of fields, including image generation [49], text generation, video generation [104], and audio synthesis.

The geometry complete diffusion model (GCDM) is an extension of the diffusion model that incorporates geometric constraints and priors into the diffusion process [105]. It leverages the underlying geometric structure of the data to guide the diffusion process, resulting in improved generation quality and better preservation of geometric properties. The GCDM takes into account geometric relationships such as distances, angles, and shape characteristics, allowing for more precise and controlled generation of samples.

The video diffusion model (VDM) is a specific type of diffusion model designed for generating videos. It extends the diffusion process to the temporal dimension, allowing for the generation of coherent and dynamic sequences of frames [106]. The VDM progressively corrupts the video frames with noise perturbations and then learns to denoise and generate realistic video sequences. It captures the temporal dependencies and dynamics of the data distribution, enabling the generation of videos with smooth transitions and realistic motion.

3.2.5. Language Models

Language models (LMs) have undergone a significant transformation in recent years, evolving from their traditional role of generating or evaluating fluent natural text to becoming powerful tools for text understanding. This shift has been achieved through the utilization of language modeling as a pre-training task for feature extractors, where the hidden vectors learned during language modeling are leveraged in downstream language understanding systems [75]. LMs have proven instrumental in a wide range of applications, enabling tasks such as answering factoid questions, addressing commonsense queries, and extracting factual knowledge about entity relations. At its core, a language model is a computational framework that aims to understand and generate human-like text. It operates based on the fundamental principle of probabilistic prediction, where it learns patterns and dependencies in sequences of words to estimate the likelihood of a particular word given the preceding context. By capturing statistical regularities in language, LMs can generate coherent and contextually relevant text. This is achieved by training the model on vast amounts of text data, allowing it to learn the distribution of words, phrases, and syntactic structures in each language [107].

The components of a language model consist of the training data, the architecture of the model itself, and the inference mechanism used for generating text. The training data serve as the foundation for learning the underlying patterns and probabilities in language.

The architecture of the model encompasses various neural network architectures, such as recurrent neural networks (RNNs), transformers, or a combination of both, which enable the model to capture long-range dependencies and contextual information. The inference mechanism involves utilizing the trained model to generate text based on input prompts or predicting missing words in each context. In Figure 9, the RNN architecture, the input sequence X is processed step by step, where $X(t)$ represents the input at each time step. The goal is to predict an output sequence y. At each time step, the RNN takes the current input $X(t)$ and the previous hidden state $h(t-1)$ as inputs. The hidden state $h(t)$ represents the network's memory and is computed using a set of learnable parameters and activation functions. In some cases, cell state is used alongside the hidden state, as seen in long short-term memory (LSTM) and gated recurrent unit (GRU) variants. The cell state acts as a long-term memory component. The hidden state $h(t)$ is then used to generate the output sequence $y(t)$ [19], which can be used for tasks like sequence-to-sequence [108] predictions.

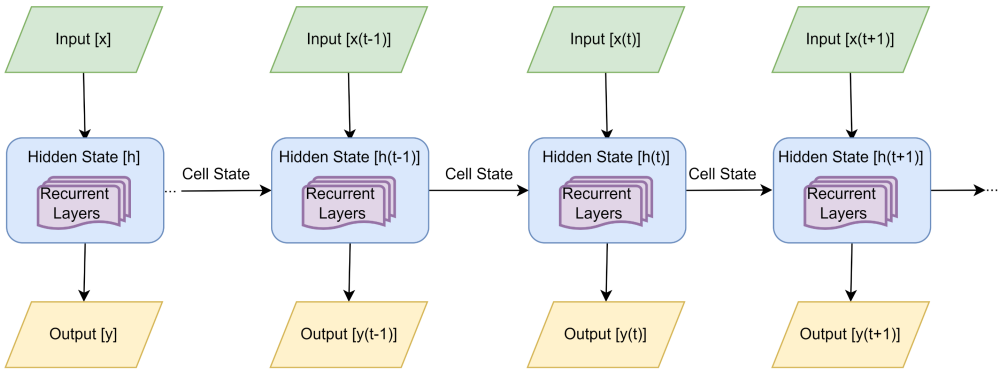

Figure 9. Recurrent Neural Network Architecture.

Language models are used for a variety of tasks, which are supported by different types of language models such as the visual language model (VLM) [109], which combines textual and visual information to understand and generate language in the context of visual data. By leveraging visual input, such as images or videos, VLMs can accurately interpret the content and generate captions, answer questions, and perform other language-related tasks. A collaborative language model (CLM) [34] is developed through the collective effort of multiple individuals or organizations. The collaborative nature of CLMs incorporates diverse perspectives and insights to enhance the quality and reliability of their language generation capabilities. By leveraging the collective wisdom of contributors and subject matter experts.The large language model (LLM) [107] represents language models that have been trained on extensive textual data and possess many parameters. With billions of parameters, LLMs, like GPT-3, demonstrate the ability to generate sophisticated and human-like text across a wide range of topics and writing styles. These language model variants play crucial roles in natural language processing and have the potential to enhance various applications and systems reliant on human-like text generation.

3.2.6. Transformers

The transformer model has revolutionized the field of natural language processing (NLP) by replacing traditional recurrent neural networks (RNNs) with a self-attention mechanism. This model has achieved state-of-the-art performance on various language tasks while being computationally efficient and highly parallelizable. The core component of the transformer model is the self-attention mechanism, which allows the model to focus on different parts of the input sequence simultaneously when making predictions. Unlike RNNs that process sequential information step by step, the transformer considers the entire input sequence at once, effectively capturing dependencies between tokens [74]. Transformer architecture consists of an encoder and a decoder, both comprising multiple layers of self-attention and feed-forward neural networks. The encoder processes the input

sequence, while the decoder generates the output sequence. The self-attention mechanism in the transformer enables the model to selectively attend to relevant parts of the input sequence, facilitating the capture of long-range dependencies and improving translation quality, among other tasks.

The attention module in the transformer adopts a multi-head design. The self-attention is formulated as a scaled dot-product [2], where the input queries (Q), keys (K), and values (V) are combined to calculate the attention weights. The scaling factor of $\sqrt{d_k}$ is applied to normalize the dot-product scores. The resulting attention weights are then multiplied with the values and summed up to produce the final output.

$$\text{Attention}(\text{Q}, \text{K}, \text{V}) = \text{softmax}(\text{QK}^{\text{T}} / \sqrt{\text{d}_\text{k}})\text{V}$$

The transformer model employs multiple layers of self-attention and fully connected point-wise layers in both the encoder and decoder components illustrated in Figure 10. This architecture allows the model to effectively capture and process the complex relationships and dependencies within the input and output sequences.

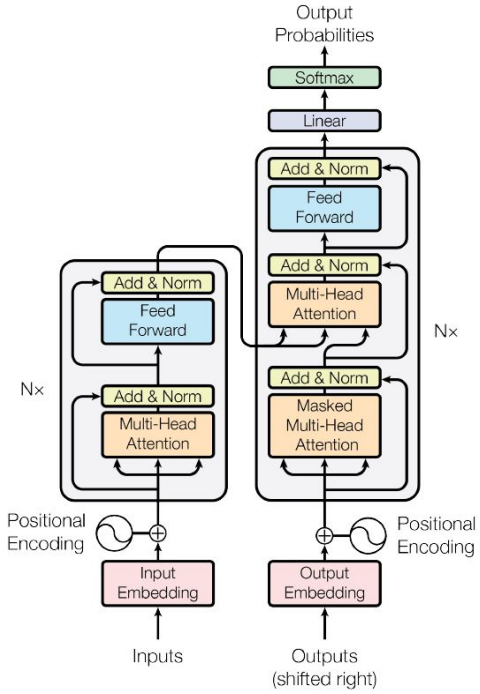

Figure 10. Transformer architecture (Figure obtained from [74]).

Transformers vary in their architectures, specific network designs, and training objectives depending on the application and input data.

BERT (Bidirectional Encoder Representations from Transformers): BERT consists of a multi-layer bidirectional transformer encoder. It employs a masked language modeling (MLM) objective during pre-training. It randomly masks words in the input text and trains the model to predict the masked words based on their context [110]. BERT also uses a next sentence prediction (NSP) task, where it learns to predict if two sentences are consecutive in each document. BERT is pre-trained on a large corpus of text, such as Wikipedia and Book Corpus. It utilizes unsupervised learning and large-scale transformer architectures to capture general language representations. After pre-training, BERT can be fine-tuned on specific downstream tasks using supervised learning with task-specific datasets.

GPT (Generative Pre-trained Transformer): GPT employs a multi-layer transformer decoder. GPT is trained using an autoregressive language modeling objective [111]. It predicts the next word in a sequence based on the previous context, enabling the generation of fluent and contextually relevant text. GPT is pre-trained on a large corpus of text, such

as web pages and books. It learns to generate text by conditioning on the preceding context. Fine-tuning of GPT can be performed on specific tasks by providing task-specific prompts or additional training data.

T5 (Text-to-Text Transfer Transformer): T5 employs a transformer architecture like BERT but follows a text-to-text framework [112]. It can handle various NLP tasks using a unified approach. T5 is trained using a text-to-text format, where both input and output are text strings. It leverages a combination of unsupervised and supervised learning objectives for pre-training and fine-tuning.

The field of transformers has witnessed remarkable progress, leading to the development of several influential models for various natural language processing (NLP) tasks. One prominent model is the adaptive text-to-speech (AdaSpeech) [113] system, which focuses on generating highly realistic and natural-sounding synthesized speech. It employs advanced techniques to overcome limitations in traditional text-to-speech systems, enabling more expressive and dynamic speech synthesis.

For code-related tasks, researchers have introduced specialized transformer models such as code understanding BERT (CuBERT) [110], CodeBERT [114], CODEGEN [50], and CodeT5 [115]. CuBERT is specifically designed for code comprehension, leveraging the power of transformers to understand and analyze source code. CodeBERT, on the other hand, performs code-related tasks like code generation, bug detection, and code summarization. CODEGEN focuses on generating code snippets given natural language descriptions, facilitating the automation of programming tasks. CodeT5, inspired by the T5 architecture, excels in various code-related tasks, including code summarization, translation, and generation. The feed-forward transformer (FFT) [42] model is a versatile transformer architecture that has demonstrated exceptional performance across multiple NLP tasks. It leverages a feed-forward neural network to process and transform input sequences, enabling effective modeling of complex language patterns and semantic relationships. The GPT language model (Codex) [116], based on the GPT-3 architecture, has gained significant attention for its ability to generate coherent and contextually relevant text. It excels in tasks such as text completion, question answering, and text generation. InstructGPT (GPT-3) [111] is another powerful language model that can understand and generate human-like text based on specific prompts. It has been extensively used in various conversational AI applications, virtual assistants, and creative writing assistance. Grapher [117] is a transformer model designed to process and understand graphical data. It leverages graph neural networks and self-attention mechanisms to capture dependencies and relationships within structured data, enabling tasks such as graph classification, node-level prediction, and link prediction. Language models for dialog applications (LaMDA) [33] are transformer-based models specifically tailored for conversational tasks. They enhance dialogue understanding and generation by capturing context, nuances, and conversational dynamics. LaMDA models have shown promise in improving conversational agents, chatbots, and virtual assistants. In the realm of multimodal tasks that involve both text and visual information, transformer-based models have also made significant contributions. MotionCLIP [118] focuses on understanding and generating textual descriptions of videos, bridging the gap between language and visual understanding. Muse explores the connection between text and image, enabling tasks such as text-based image retrieval and image captioning. The pre-trained language model (PLM)/visual GPT [65] is a multimodal model that combines text and visual information to generate coherent and contextually relevant captions for images. Other notable transformer models include T5X [119], text-to-text transfer transformer (T5) [39], TFix [45], w2v-BERT (Word2Vec and BERT) [120], and WT5 (Why, T5?) [121]. T5X extends the T5 architecture to handle even more complex NLP tasks and demonstrates superior performance in tasks such as machine translation and text summarization. TFix focuses on addressing issues related to fairness, transparency, and explainability in transformer models. w2v-BERT combines Word2Vec and BERT to enhance the representation of word semantics within the transformer framework. WT5 focuses on training text-to-text models to explain their predictions. It builds upon the architecture of the text-to-text transfer

transformer (T5) model. The primary objective of WT5 is to enhance the interpretability and explainability of the T5 model by providing insights into the reasoning behind its predictions.

3.2.7. Hybrid Models

Hybrid generative AI models are models that combine multiple generative AI techniques or architectures to leverage their respective strengths and produce improved results. These models aim to overcome limitations or enhance the capabilities of individual generative models by integrating different approaches.

Adversarial autoencoder (AAE): AAE is a type of generative model that combines elements of both autoencoders and generative adversarial networks (GANs). It is designed to learn a compact latent representation of input data while generating realistic samples from that latent space. The autoencoder is integrated with a GAN framework in an adversarial autoencoder. The autoencoder acts as a generator network, taking in random noise and creating samples in the latent space. Instead of attempting to discriminate between actual and false samples, the discriminator network seeks to distinguish between samples from the true latent distribution and samples produced by the autoencoder in Figure 11.

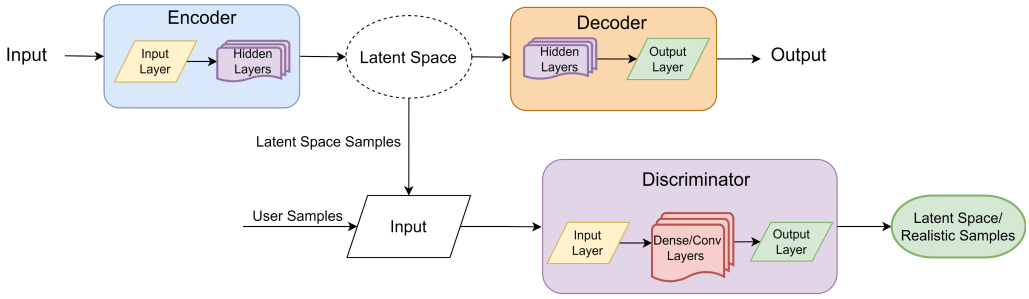

Figure 11. Adversarial autoencoder architecture (AAE).

An AAE's training consists of two major stages, the reconstruction stage where the autoencoder is trained to correctly reconstruct the input data. It reduces the reconstruction loss between the input and output, which encourages the autoencoder to develop a meaningful representation. Coming to the second stage which is the adversarial stage, where the discriminator is trained to differentiate samples derived from the actual latent distribution from samples produced by the autoencoder. The generator (autoencoder) seeks to produce samples that deceive the discriminator. This adversarial training pushes the autoencoder to generate realistic latent space samples. The AAE may learn a compact latent representation that captures the main features of the input data while generating realistic samples from that latent space by combining the reconstruction and adversarial phases. Adversarial training prevents mode collapse and makes the generator explore its entire latent space. Adversarial autoencoders have been employed in a wide range of applications, including image generation [122], anomaly detection, and data synthesis.

PixelCNN: PixelCNN is a type of generative model that belongs to the family of autoregressive models and is specifically tailored for generating images pixel by pixel. It utilizes convolutional layers to capture spatial dependencies within the image. PixelCNN models the conditional probability distribution of each pixel given its preceding context. By modeling this conditional distribution, PixelCNN can generate images that exhibit realistic textures and local coherence.

During training, PixelCNN is typically trained using maximum-likelihood estimation. The model takes an image as input and is trained to maximize the likelihood of generating that image. PixelCNN employs a process called autoregression for generating new images. It starts with an empty canvas and generates the pixels one by one, conditioning each prediction on the previously generated pixels. This autoregressive process allows the model to capture complex dependencies and generate coherent images. PixelCNN has

demonstrated success in tasks such as image completion, super-resolution, and image synthesis [123].

Variational Autoencoder with Generative Adversarial Networks (VAE-GAN): This hybrid model combines the generative capabilities of variational autoencoders (VAEs) and generative adversarial networks (GANs). The VAE component helps encode and decode input data, while the GAN component enhances the realism and diversity of the generated samples. Introspective adversarial networks [124] and Mol-CycleGAN [125] are examples of this combination. In introspective adversarial networks, there are other techniques to improve its performance, such as multiscale dilated convolution blocks and orthogonal regularization. These techniques help the model to capture long-range dependencies in the image, prevent overfitting, and generate images that are more realistic and coherent. Mol-CycleGAN extends the CycleGAN framework to molecular embeddings in the latent space of JT-VAE utilizes the latent space of JT-VAE (junction tree variational autoencoder) as the embedding representation. The latent space is created by a neural network during the training process. The advantage of using the latent space embedding is that the distance between molecules can be defined directly in this space, enabling the calculation of the loss function. VAE-GANs have been successfully applied in various domains, including image synthesis, text generation, and music composition.

Generative Adversarial Networks (GAN) with Dense Convolutional Neural Networks (DenseNet) or Residual Neural Networks (ResNet): Dense convolutional neural networks (DenseNet) are known for their dense connections, which facilitate feature reuse and enhance the flow of gradients throughout the network. DenseNet architectures have shown remarkable performance in image classification tasks by capturing intricate patterns and representations in the data. When combined with generative adversarial networks (GANs), DenseNet can contribute to the generator component of the GAN framework. By utilizing DenseNet as the generator, the hybrid model benefits from its powerful feature learning capabilities and the ability to capture complex patterns and details in the data. ResNet is also used in similar way but there is a slight difference between them ResNet's skip connections enable training of very deep networks, while DenseNet's dense connectivity promotes parameter efficiency and better information flow. This combination of models is done in CycleGAN [54], PGGAN [126]. CycleGAN is a powerful framework for unsupervised image translation by leveraging the concept of cycle consistency and utilizing architectures such as ResNet and PatchGAN to achieve impressive results in various image-to-image translation tasks. The PGGAN discriminator, formed by combining PatchGAN and G-GAN, provides fine-grained evaluation of local image patches and incorporates gradient penalty regularization, enhancing the training stability and diversity of generated samples in the PGGAN framework.

Generative Adversarial Networks (GAN) with Recurrent Neural Networks (RNN) or Convolutional Neural Networks (CNN): Combining RNNs or CNNs with GANs, it becomes possible to generate sequences that possess both coherence and realism. The RNN component provides the ability to model sequential dependencies, ensuring that the generated sequences flow naturally and exhibit contextual understanding. The GAN component, on the other hand, improves the diversity and quality of the generated sequences by leveraging the adversarial training process. In the RTT-GAN [62], the generator of the GAN employs a hierarchical structure and attention mechanisms to retain contextual states at various levels and a hierarchical structure and attention mechanisms to retain contextual states at various levels. The hierarchy is formed by a paragraph-level recurrent neural network (RNN), a sentence-level RNN, and a word-level RNN, along with two attention modules. The paragraph RNN encodes the current paragraph state by considering preceding sentences. The spatial–visual attention module selectively focuses on semantic regions, guided by the current paragraph state, to generate the visual representation of the sentence. Consequently, the sentence RNN can encode a topic vector for the newly generated sentence. The discriminator LSTM RNN takes the sentence embeddings of all preceding sentences as inputs. It computes the topic smoothness value of the current constructed paragraph description at each recurrent step, assessing the coherence of topics across the generated sentences.

With these multi-level assessments, the model can generate long yet realistic descriptions, maintaining both sentence-level plausibility and topic coherence. In the CNN-GAN model, a convolutional encoder–decoder network is utilized for generating new content by jointly training it with adversarial networks. This training setup aims to ensure coherence between the generated pixels and the existing ones. These CNN-based methods have demonstrated the ability to generate realistic and plausible content in highly structured images, such as faces, objects, and scenes [127].

Generative Adversarial Networks (GAN) with Denoising Diffusion Probabilistic Models (DDPM) and Transformers: Combination DDPMs, GANs, and transformers can create a hybrid generative AI model with enhanced capabilities. This combination allows for the generation of diverse and high-quality samples while leveraging the strengths of each component. DiffGAN-TTS [128] and ProDiff [53] implement this combination of models. DiffGAN-TTS is a novel text-to-speech (TTS) model that achieves high-fidelity and efficient speech synthesis. It takes inspiration from the denoising diffusion GAN model and models the denoising distribution using an expressive acoustic generator. This generator is trained adversarially to match the true denoising distribution, ensuring high-quality output spectrograms. DiffGAN-TTS ability to allow large denoising steps during inference. This reduces the number of denoising steps required and accelerates the sampling process. To further enhance sampling efficiency, DiffGAN-TTS incorporates an active shallow diffusion mechanism. ProDiff utilizes generator-based parameterization, where the denoising model directly predicts clean data using a neural network [53]. This approach has shown advantages in accelerating sampling from complex distributions. By directly predicting clean data, ProDiff avoids the need to estimate gradients and achieves faster synthesis.

Transformer with Recurrent Neural Network (RNN): The combination of transformers and RNNs can leverage the strengths of both architectures, allowing for improved modeling of sequential data with long-term dependencies and global context understanding. This combination is useful for tasks such as speech recognition, time series forecasting, and video processing, where both local temporal dependencies and global context are crucial for accurate predictions. MolT5 [58] implements three baseline models for the tasks of molecule captioning and molecule generation. The first baseline is a four-layer GRU recurrent neural network with a bidirectional encoder. This model leverages the sequential nature of the data and captures contextual information from both past and future. The second baseline is based on the transformer architecture, consisting of six encoder and decoder layers. Transformers utilize self-attention mechanisms to capture global dependencies and have been successful in various sequence-to-sequence tasks. The third baseline is based on the T5 model, a pre-trained sequence-to-sequence model. Three T5 checkpoints, namely small, base, and large, are fine-tuned for molecule captioning and molecule generation. T5 models have shown strong performance in natural language processing tasks.

Transformer with Graph Convolutional Network (GCN): For tasks that require graph-structured data, this hybrid model combines the strength of transformers and GCNs. Transformers excel at sequence-to-sequence tasks and have demonstrated success in natural language processing and image processing. GCNs, on the other hand, are especially intended to handle graph-structured data and capture node relationships. This hybrid model can effectively capture both the sequential dependencies of the data and the graph-based relationships by combining transformers and GCNs, enabling enhanced modeling and representation learning in graph-based tasks such as node classification, link prediction, graph generation and molecule structure generation [60].

Transformer with Long Short-Term Memory (LSTM): The transformer architecture with long short-term memory (LSTM) is a type of recurrent neural network known for its ability to capture long-term dependencies in sequential data. Transformers are powerful models for sequence processing, leveraging self-attention mechanisms to capture dependencies across the sequence. The GTR-LSTM [129] encoder provides a graph-based approach to encoding triples, considering the structural relationships between entities in a knowledge

graph. By incorporating attention mechanisms and entity masking, the model aims to generate coherent and meaningful output sequences based on the input graph.

Vision Transformers with Residual Neural Networks (ResNet): Vision transformers leverage the self-attention mechanism of transformers to capture long-range dependencies and enable effective modeling of image data. The combination of ResNet and vision transformers can benefit from both the local feature extraction capabilities of ResNet and the global context understanding of vision transformers, resulting in improved image understanding and representation [130].

Diffusion probabilistic models with Contrastive Language-Image Pretraining (CLIP): Diffusion modeling is a powerful technique for modeling complex data distributions and generating high-quality samples. CLIP, on the other hand, is a state-of-the-art method for learning visual representations from images and corresponding textual descriptions. DiffusionCLIP [131] combines the power of diffusion modeling and the guidance of CLIP to enable precise and controlled image manipulation. It leverages pretrained diffusion models and the CLIP loss to fine-tune the diffusion model and generate samples that align with a target textual description, which opens new possibilities for image generation and manipulation tasks.

Convolutional Neural Network (CNN) with Bidirectional Encoder Representations from Transformers (BERT): CLAP (contrastive learning for audio and text pairing) [47] is a model that jointly trains an audio encoder and a text encoder to learn the similarity or dissimilarity between audio and text pairs. The goal is to enable zero-shot classification by computing embeddings for audio and text and using cosine similarity to measure their similarity. The model takes audio and text pairs as input, which are separately processed by the audio encoder and text encoder. The encoders extract meaningful representations from the audio and text inputs. These representations are then projected into a joint multimodal space using linear projections.

Convolutional Sequence-to-Sequence Learning (ConvS2S): This is a neural network architecture that was introduced for sequence-to-sequence tasks, such as machine translation or speech recognition. It leverages convolutional neural networks (CNNs) to process input sequences and generate output sequences, providing an alternative to the commonly used recurrent neural networks (RNNs) Unlike RNN-based models that rely on sequential processing, ConvS2S [132] applies parallel convolutions across the input sequence. This enables more efficient computation and allows for better utilization of parallel processing capabilities, leading to faster training and inference times. The use of convolutions also helps capture local dependencies in the input sequence, which can be beneficial for tasks where context is primarily determined by nearby elements. The architecture of ConvS2S typically consists of an encoder and a decoder. The encoder is composed of several layers of 1D convolutional filters followed by non-linear activation functions. These filters capture different patterns and features in the input sequence, allowing for effective representation learning. The decoder, on the other hand, employs similar convolutional layers but with additional techniques like attention mechanisms to generate the output sequence. Unlike RNN-based models that rely on sequential processing, ConvS2S applies parallel convolutions across the input sequence. This enables more efficient computation and allows for better utilization of parallel processing capabilities, leading to faster training and inference times. The use of convolutions also helps capture local dependencies in the input sequence, which can be beneficial for tasks where context is primarily determined by nearby elements. The architecture of ConvS2S typically consists of an encoder and a decoder. The encoder is composed of several layers of 1D convolutional filters followed by non-linear activation functions. These filters capture different patterns and features in the input sequence, allowing for effective representation learning. The decoder, on the other hand, employs similar convolutional layers but with additional techniques like attention mechanisms to generate the output sequence.

3.3. AIGC Input–Output Classification

The field of AI content generation encompasses a wide array of aspects, leading to the development of numerous methods. These methods can be categorized into different groups based on the nature of the input and output involved. In this section, we will explore various techniques that enable the generation of AI-driven content by transforming different types of input into desired output. The classification of the input–output types of AIGC is given in Table 5.

Table 5. Generative AI Models for Various Input–Output Transformations.

Input	Output	Prescribed Task	Technique/ Technology/ Model	Ref.
Image	3D Image	Text-guided 3D object generation: generating 3D objects based on textual descriptions	DREAMFUSION	[133]
	Image	Blind motion deblurring of a single photograph	DeblurGAN	[61]
			DeblurGAN-v2	[56]
	Image	Generate highly realistic and diverse synthetic images	StyleGAN	[4]
	Image	Blending two Images	Gaussian-Poisson Generative Adversarial Network (GP-GAN)	[134]
	Image	Image compositing or image blending	Geometrically and Color Consistent GANs (GCC-GANs)	[64]
	Image	Filling absent pixels in an image or image inpainting	Exemplar GANs (ExGANs)	[99]
			Contextual Attention Generative Adversarial Networks (CA-Generative Adversarial Networks (GAN))	[127]
			PGGAN	[126]
	Image	Face aging: generating images that depict a hypothetical future appearance of a person.	Age-cGAN	[135]
			Conditional Adversarial Autoencoder (CAAE)	[122]
			Identity-Preserved Conditional Generative Adversarial Networks (IPCGANs)	[136]
	Image	Image editing	Introspective Adversarial Network (IAN)	[124]
	Image	Medical image analysis: segmenting objects or regions in an image	SegAN	[100]
	Image	Converting low-resolution images to high resolution	Multi-Level Densely Connected Super-Resolution Network (mD-CSRN)	[137]
	Image	Synthesizing a photorealistic frontal view from a single face image	Two-Pathway Generative Adversarial Network	[138]
	Image	To increase the resolution of an image	Super Resolution GAN (SRGAN)	[90]
	Image	Generates high-quality face samples at a resolution of 128 × 128 pixels	Boundary Equilibrium GAN (BEGAN)	[139]
	Image	To increase the resolution of an image better than SRGAN	Enhanced Super Resolution GAN (SRGAN)	[140]
	Image	To convert the image content from one domain to another	Conditional Generative Adversarial Networks (CGAN)	[81]
	Image	Style transfer, image-to-image translation, domain adaptation, data augmentation	cycle generative adversarial networks (CycleGAN)	[87]
	Image	Style transfer, image synthesis, image-to-image translation, and domain adaptation	Discover Cross-Domain Relations with Generative Adversarial Networks (DiscoGAN)	[88]
	Image	Method for training generative neural networks for efficient texture synthesis	Markovian Generative Adversarial Networks (MGANs)	[141]
	Image	Specifically designed for spatial data and related tasks such as image generation, editing, manipulation, data augmentation, and style transfer	Spatial Generative Adversarial Networks (Spatial GAN)	[142]
	Image	Creating tileable textures for 3D models, generating repeating backgrounds or surfaces for digital art or design, or synthesizing periodic visual elements for games or virtual environments	Periodic Spatial Generative Adversarial Networks (Spatial GAN)	[143]
	Image	To generate high-quality, high-resolution, and diverse synthetic images that resemble real-world images	Big Generative Adversarial Networks (BigGAN)	[43]
	Image	Cyber intrusion and malware detection	Defense-Generative Adversarial Networks	[95]
	Image	Generating images in a target domain from a different source domain	Domain Transfer Network (DTN)	[89]
	Text	Generate textual descriptions for given images	Recurrent Topic-Transition Generative Adversarial Network (RTT-Generative Adversarial Networks (GAN))	[62]
	Text	Image-to-text generation or image captioning	Show and Tell: Neural Image Captioning	[59]
	Text	Generating handwritten characters in a target font style or creating new fonts or handwritten font generation	DenseNet Cycle Generative Adversarial Networks (GAN)	[54]
	Text	Answers questions based on image input	Visual Language Model - Flamingo	[109]

Table 5. *Cont.*

Input	Output	Prescribed Task	Technique/ Technology/ Model	Ref.
Text	3D Image	Generate 3D images using textual descriptions	Magic3D	[103]
	3D Animated Avatar	Generate text-driven 3D avatar with animations	AvatarCLIP	[66]
	3D Faces	Generate personalized, animatable 3D faces using text guidance	DreamFace	[102]
	3D Human Avatar	Generate 3D human avatars with identities and artistic styles using a text prompt	AvatarCraft	[55]
	3D Human Motion	Generate 3D motion using text descriptions	MotionCLIP	[118]
	Animated Character	Generate animated characters from text	Progressive Structure-conditional GANs (PSGAN)	[144]
	Audio	Generate audio using text	w2v-BERT (Word2Vec and BERT)	[120]
	Music	Generate music from lyrics	Jukebox	[145]
	Code	Generate valid programming code using natural language descriptions	CodeBERT	[114]
			CODEGEN	[50]
			CodeT5	[115]
			Codex	[116]
		It assists in code completion, bug detection, and code summarization	Code Understanding BERT (CuBERT)	[110]
		Generate competition-level code	Pre-trained Transformer-Based Language Model - Alphacode	[35]
	Knowledge graph	Generate a knowledge graph (KG) using textual inputs	Grapher	[117]
	Image	Generate images using text	Text Conditioned Auxiliary Classifier Generative Adversarial Network (TAC-GAN)	[84]
		Generate Steganographic images to hide secret information	Steganographic Generative Adversarial Networks model (SGAN)	[146]
			Secure Steganography Based on Generative Adversarial Networks (SS-GAN)	[96]
		Manipulate/edit images using textual descriptions	Text-Adaptive Generative Adversarial Network (TAGAN)	[67]
		Generate images based on textual instructions	Denoising Diffusion Probabilistic Models (DDPM)	[101]
			Guided Language to Image Diffusion for Generation (GLIDE)	[147]
			Imagen	[148]
			Attentional Generative Adversarial Networks (AttnGAN)	[92]
			CogView	[77]
			Auxiliary Classifier GANs (AC-GAN)	[149]
			Stacked Generative Adversarial Networks (StackGAN)	[93]
			alignDRAW (Deep Recurrent Attention Writer)	[78]
			Deep Convolutional Generative Adversarial Networks (DCGAN)	[86]
			Muse	[150]
			Text Conditioned Auxiliary Classifier GAN (TAC-GAN)	[67]
	Image	Generate more complex images using captions	Generative Adversarial CLIPs (GALIP)	[151]
	Image	Generate original, realistic images and art using a text prompt	Contrastive Language Image Pre-training (CLIP)	[130]
	Molecule	Text-based de novo molecule generation, molecule captioning	MolT5 (Molecular T5)	[58]
	Molecule Structure	Generate or retrieve molecular structures using textual description	Text2Mol	[60]
	Speech	Synthesize custom voice speech using text	Adaptive Text to Speech (AdaSpeech)	[113]
		Convert text to human-like speech	Denoising Diffusion Model for Text-to-Speech (Diff-TTS)	[63]
			Grad-TTS	[152]
			ProDiff	[53]
			DiffGenerative Adversarial Networks (GAN)-TTS	[128]
			Pixel Convolutional Neural Network - Wavenet	[123]
		Generate speech using text	Feed-Forward Transformer (FFT)	[42]
		Generate high-quality, synthetic musical audio clips	Generative Adversarial Networks Synth (GANSynth)	[153]
	Text	To translate text from one language to another	Text-to-Text Transfer Transformer (T5)	[39]
			Convolutional Sequence to Sequence Learning (ConvS2S)	[132]
			Sequence to Sequence (Seq2Seq)	[108]
	Text	Generate handwritten characters in a target/new font style using text	GlyphGAN	[154]
	Text	Generate accurate and meaningful corrections for code issues	TFix	[45]
	Text	Explains the given input statements	WT5 (Why, T5?)	[121]
	Text	Perform tasks like translation, question answering, classification, and summarization using input texts	Text-To-Text Transformer (T5)	[39]
	Text	Generate or crack passwords	PassGAN	[155]
	Text	Chat with users, answer follow-up questions, challenge incorrect premises, and reject inappropriate requests	InstructGPT (GPT-3)	[111]
	Text	Operate as a conversational AI system to chat with users and answer follow-up questions	Language Models for Dialog Applications (LaMDA)	[33]

Table 5. *Cont.*

Input	Output	Prescribed Task	Technique/ Technology/ Model	Ref.
	Text	Write drafts, add suggestions, propose edits and provide explanations for its actions	PEER (Plan, Edit, Explain, Repeat)	[34]
	Text	Password cracking	Improved Wasserstein GAN (IWGAN)	[85]
	Text	Predict future markets using historical data	GAN-FD	[156]
			Stochastic Time-series Generative Adversarial Network (ST-GAN)	[157]
			Multiple Time-series Generative Adversarial Networks (MTSGAN)	[158]
	Video	Generate text-guided videos	MAKE-A-VIDEO	[159]
			IMAGEN VIDEO	[160]
			Tune-A-Video	[104]
2D Structure Molecule	3D Structure Molecule	Generating 3D molecular structure	Geometry Complete Diffusion Model (GCDM)	[105]
3D Image	3D Image	Performing inpainting on 3D images	Point Encoder GAN	[161]
		Generation of realistic human poses	GAN-Poser	[51]
Audio	Text	Generating captions for audio	Contrastive Language-Audio Pretraining (CLAP)	[47]
Chemical Properties	Molecule	Designing molecules/drugs with desired properties	Mol-Cycle Generative Adversarial Networks (GAN)	[125]
	Molecular Graphs	Creating molecular graphs or designing molecule graphs	Junction Tree Variational Autoencoder (JT-VAE)	[57]
	Molecular Graphs	Designing molecule graphs from chemical properties	MoFlow	[76]
Data	Text	Generating natural language from structured data	Text-To-Text Transformer (T5)	[112]
Gesture	Text	Gesture recognition	DCGAN (Deep Convolutional Generative Adversarial Network) with CNN (Convolutional Neural Network)	[36]
Graphs	Molecule Structure	Molecule generation	Constrained Graph Variational Autoencoder (CGVAE)	[79]
	Graph	Generates the periodic structure of materials	Crystal Diffusion Variational Autoencoders (CDVAE)	[80]
Image+Text	Image	A text-guided image manipulation method	LDEdit	[49]
		Generating steganographic images (hiding messages in an image)	SteganoGAN	[162]
		Performing text-driven image manipulation/editing	Style Contrastive Language-Image Pre-training (StyleCLIP)	[98]
		Describing and editing the given image based on the text prompt	Pre-trained Language Model (PLM) - Visual GPT	[65]
Knowledge Graph	Text	Converting knowledge graph-based RDF triples to text	GTR-Long Short-Term Memory (LSTM)	[129]
Music	Text	Generating captions for music audio	MusCaps	[48]
Road Network	Road Network	Synthesizing road networks	StreetGAN	[97]
Speech	Speech	Speech enhancement	SEcGAN	[82]
Tabular Data	Tabular Data	Synthesize fake tables that are statistically similar to the original table	Table-GAN	[52]
		Generate a synthetic dataset that is statistically similar to the original data	Tabular GAN (TGAN)	[94]
		Generate synthetic data for tabular datasets	Conditional Tabular GAN (CTGAN)	[83]
Text+Image	Image	Performing text-based realistic image synthesis/generation	Semantic Image Synthesis via Adversarial Learning (SISGAN)	[163]
		Text-based image manipulation	DIFFEDIT	[164]
			DiffusionCLIP	[131]
	Video	Generating video from text prompt and input image	PHENAKI	[119]
Text+Shape	3D Avatar	Generating 3D avatars guided by text and shape	DreamAvatar	[165]
Video	Video	Converting low-resolution videos to higher-resolution videos	VSRResFeatGAN	[91]
Video+Text	Video	Editing videos based on text input and animating images based on input (image+text)	Video Diffusion Model (VDM) - Dreamix	[106]

3.3.1. Text to Text

In the field of natural language processing, the ability to transform text into various textual outputs has been revolutionized by generative AI techniques. This section explores the field of text-to-text generation, where diverse tasks are accomplished by leveraging advanced models. Table 5 provides a comprehensive overview of the wide range of tasks that can be achieved using text-to-text generation approaches. At the core of these techniques lies the process initiated by the user input or 'prompt'. This input is processed through an encoder, which not only interprets the text but also converts it into a series of 'hidden states'. The decoder then takes these hidden states and performs further processing to generate a response that is contextually relevant to the user's prompt. The response can vary depending on the task: translation, answering questions, suggesting code corrections, or generating text in different fonts, the possibilities are vast. ChatGPT is an impressive text-to-text technique that enhances the GPT-3 [5] architecture to engage in dynamic conver-

sations. In addition to ChatGPT, other influential models like T5 [39], ConvS2S [132], and Seq2Seq [108] have made significant contributions in the field. These models excel in tasks such as language translation. InstructGPT [111] specializes in handling follow-up questions, challenging incorrect assumptions, and rejecting inappropriate requests, while TFix [45] assists in code corrections and suggests meaningful improvements. The underlying process of text-to-text generation encompasses both transformer-based architectures and models that harness the power of generative adversarial networks (GANs). For example, models like PassGAN [155] employ GANs to generate or crack passwords, and GlyphGAN [154] utilizes GANs to create handwritten characters in different font styles based on text input. This fusion of transformer-based approaches and GAN-based models expands the horizons of text-to-text generation, unlocking exciting possibilities in natural language processing.

3.3.2. Text to Image

Text-to-image techniques have evolved significantly, enabling the generation of images from textual descriptions. Previously, image captioning, an image-to-text approach, was more prevalent. However, with the emergence of notable applications like DALL-E [166] and Midjourney, along with other existing models, text-to-image synthesis has gained prominence. Figure 12 was generated from DALL-E with a given prompt and Figure 13 was generated by Midjourney. Various architectures, such as GAN, diffusion, VAE, and transformers, are employed to facilitate text-to-image generation. In order to provide a comprehensive overview of the field, Table 5 presents prescribed tasks and various techniques/models utilized in text-to-image synthesis. Among the diffusion-based models, GLIDE [147] and Imagen [148] have gained popularity for their impressive results. StackGAN [93] has introduced a two-stage image generation process using GAN, involving Stage-I GAN and Stage-II GAN, which enables the generation of high-quality images. AttnGAN [92] is another prominent technique that employs multi-stage refinement, significantly improving the quality of the generated images. Additionally, novel approaches like SGAN [146] and SSGAN [96] have proven helpful in the domain of stenography. Figure 14 was obtained from Imagen for the given prompt "Sprouts in the shape of text 'Imagen' coming out of a fairytale book".

Figure 12. Prompt is "An astronaut dance party on the surface of Mars". (Image was obtained from DALL-E.)

Figure 13. Prompt is "3D oil painting tulip". (Image was obtained from Midjourney.)

Figure 14. Prompt is "Sprouts in the shape of text 'Imagen' coming out of a fairytale book". (Image was obtained from Imagen.)

3.3.3. Text to Audio/Speech

Text-to-speech (TTS) and text-to-audio techniques both involve converting text into human-like speech and generating music with various vocal styles, respectively. TTS is commonly used in applications such as voice assistants, voice navigation systems, and audiobooks. One notable approach in this field is AdaSpeech [113], which is an efficient custom voice synthesis technique. It utilizes two acoustic encoders: one for extracting the sequence of phonemes and another for extracting utterance-level information. By incorporating conditional layer normalization in the mel-spectrogram decoder, AdaSpeech [113] enhances the quality of the synthesized voice. Several diffusion-based models, such as Diff-TTS [63], Grad-TTS [152], and ProDiff [53], have been developed to achieve more human-like speech synthesis using diffusion techniques. Additionally, a hybrid model called DiffGAN-TTS [128] combines both diffusion and GAN architecture to further enhance the quality of speech synthesis. Another interesting technique is GANSynth [153], which focuses on generating high-quality musical clips from text. Jukebox [145], on the other hand, is a VQ-VAE based model that can generate music from lyrics, thereby accomplishing the lyrics-to-singing (LTS) task. In the domain of audio classification, Microsoft's CLAP [47] is a notable pioneer. It utilizes audio and text encoders to classify audio samples, such as identifying sounds like audience claps, bird sounds, and other environmental audio cues. This technique can be considered an example of audio-to-text conversion. Additionally, there are techniques like MusCaps [48], which aid in captioning music by genre, and SecGAN [82], a speech enhancement technique that uses conditional GAN to achieve speech-to-speech conversion. These techniques showcase the versatility and broad range of applications within the realm of audio and speech processing.

3.3.4. Text to Code and Code to Text

In the world of software development, the task of manually writing or replicating code patterns can be a time-consuming process. However, there is an innovative solution known as text to code and code to text, which offers significant benefits. Text to code enables us to generate entire source code for specific business problems, resolve issues within existing code by providing suggestions for corrections, while code to text empowers us to generate documentations for the code. Moreover, these cutting-edge advancements leverage generative AI techniques, allowing us to effortlessly complete simple functions. Notably, GitHub has recently launched GitHub Copilot, which utilizes Open AI Codex [116]. It allows for tasks such as generating repetitive code patterns or even entire functions by simply providing natural language comments in the editor. Figure 15 demonstrates the generation of source code using the textual prompt. Another noteworthy model is CodeBERT [114], which is based on a bimodal transformer [1] technique. CodeBERT supports natural-language (NL)-to-programming-language (PL) applications, enabling tasks such as code search, code documentation (code to text), and code review. Additionally, CodeBERT [114] has been further pretrained for CodeReviewer [167], CodeExecutor [168](for tracking execution traces), GraphCodeBERT [169] (for code refinement and translation between programming languages), and UniXcoder [170] (for code generation

tasks). CODEGEN [50] is an autoregressive-transformer-based large language model (LLM) that specializes in program synthesis based on input and expected output or even natural language descriptions. CodeT5 [115], on the other hand, is a pre-trained encoder–decoder transformer model capable of tasks such as code understanding, code generation, and converting source code between programming languages. Additionally, AlphaCode [35], another encoder–decoder transformer-based model, excels in understanding algorithms and generating competition-level source code for given problem statements. This model has been effectively evaluated on the Codeforces platform.

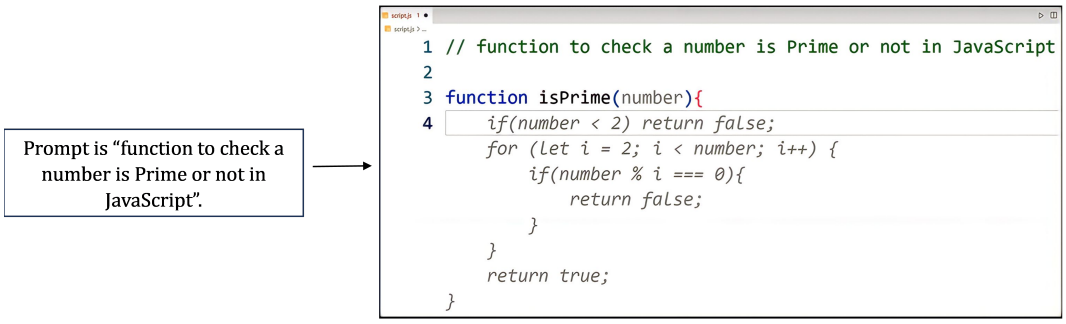

Figure 15. Source code for isPrime() function generated by Github Copilot using Codex.

3.3.5. Image to Text

The content of an image can be described using AIGC techniques, which leverages the power of computer vision and natural language processing. One notable model in this field is the Show and Tell: Neural Image Caption (NIC) [59] developed by the Google research team, which serves as the foundation for image captioning applications.

The architecture process of NIC [59] involves the utilization of convolutional neural networks (CNN) and long short-term memory (LSTM), a type of recurrent neural network (RNN). The CNN acts as an encoder, extracting visual features from the image. These features capture the salient information necessary for generating meaningful captions. The extracted features are then passed on to the LSTM, which functions as a decoder. The LSTM generates a sequence of words or a description based on the visual features obtained from the CNN.

Subsequently, attention mechanisms were introduced to further improve image captioning. One such model is the recurrent topic-transition GAN (RTT-GAN) [62], which enhances the caption generation process by incorporating attention-based techniques. The architecture of RTT-GAN [62] includes a generator and two discriminators. The generator recurrently produces sentences using semantic regions, allowing it to focus on different aspects of the image and generate more contextually relevant captions. The two discriminators assess the quality of the sentences generated by the generator, ensuring that the captions are accurate and coherent.

Another noteworthy model within the realm of visual attention is show, attend and tell [171]. This model also incorporates an visual attention mechanism, specifically the soft attention mechanism. It allows the model to dynamically focus on different regions of the image while generating captions. By attending to relevant image regions, the model can generate more accurate and detailed descriptions. VisualGPT [65] based on pre-trained language model (PLM) which employs an encoder–decoder attention mechanism to generate a suitable caption for an given image.

Overall, these models demonstrate the advancements in image-to-text generation by combining the power of computer vision, natural language processing, and attention mechanisms.

3.3.6. Approaches for Visual Content Generation

Visual content generation techniques have evolved significantly in recent years, offering diverse approaches for creating compelling visual media. This section explores three key methods: text to video, text+video to video, and video to video. These techniques enable the generation of visual content based on text descriptions, combined text and video inputs, and existing video data, respectively. By referring to Table 5, which outlines model names and their associated tasks, we can obtain a comprehensive understanding of the diverse potential and progress within the field of visual content generation.

Image to Image

In the realm of image processing and synthesis, image-to-image techniques offer a wide array of tasks and models that empower us to manipulate and transform images in diverse ways, including the synthesis of highly realistic images. One such powerful technique in generating highly realistic and diverse synthetic images is StyleGAN [4]. Its architecture includes a generator network that produces synthetic images based on a learned mapping from a latent space to the image space. An intriguing aspect of StyleGAN [4] is the introduction of adaptive instance normalization, which grants control over different aspects of the image's style and appearance. Consequently, this approach leads to the generation of highly realistic and visually diverse synthetic images.

Another task within image-to-image techniques is image editing, which allows for precise modifications to images. The Neural Photo Editor with Introspective Adversarial Network [124], a hybrid model combining the power of generative adversarial networks (GANs) and variational autoencoders (VAEs), enables us to edit various aspects of an image, such as color or even hair color in a portrait. This model's architecture leverages the strengths of GANs and VAEs.

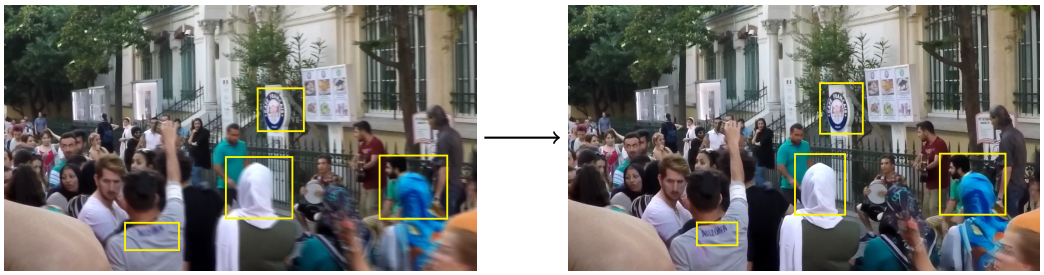

Figure 16. Deblurring the images using DeblurGAN-v2 (DeblurGAN-v2 model downloaded from https://github.com/VITA-Group/DeblurGANv2, (accessed on 25 July 2023) and used our image).

Image deblurring aims to enhance the clarity and sharpness of blurred images. Models like DeblurGAN [61], which is based on conditional GANs, and DeblurGAN-v2 [56], a conditional GAN with a double-scale discriminator, effectively address this challenge. Figure 16 demonstrates the deblurring task on images using DeblurGAN-v2. Style transfer is another fascinating task within image-to-image techniques, enabling the application of artistic styles to images. Models like CycleGAN [87] leverage the power of adversarial training and cyclical consistency loss to learn mappings between different visual domains. This allows for the translation of images between domains while preserving important style characteristics, opening up new possibilities for creative expression. Image compositing or blending involves the seamless combination of two images. Geometrically and color consistent GANs (GCC-GANs) [64] specialize in this task. The architecture of GCC-GANs includes four sub-networks: a transformation network, a refinement network, a discriminator network, and a segmentation network. The transformation and refinement networks work together to generate the composited image, while the discriminator and segmentation networks play a crucial role in increasing the realism of the blended image by leveraging geometric and color consistency constraints. By incorporating these sub-networks, GCC-GANs [64] ensure visually appealing and realistic image blending. To fill absent pixels in

an image or perform image inpainting, Exemplar GANs (ExGANs) [99] come into play. By leveraging exemplar-based techniques, ExGANs [99] effectively fill in the missing areas, resulting in visually plausible and coherent images.

For face aging, there are models like Age-cGAN [135], conditional adversarial autoencoders (CAAE) [122], and identity-preserved conditional generative adversarial networks (IPCGANs) [136]. Age-cGAN [135] employs an architecture that includes a generator network responsible for generating aged faces based on input images and a discriminator network that provides feedback on the realism and age progression of the generated faces. IPCGANs [136] also adopt a similar architecture, with a focus on preserving the identity of the person while generating the hypothetical future appearance. By incorporating conditional inputs and adversarial training, these models generate realistic and convincing aged face images.

Furthermore, image segmentation plays a vital role in various domains, particularly in medical image analysis. SegAN [100], a model combining GANs and convolutional neural networks (CNNs), excels in accurately segmenting objects or regions within medical images. By leveraging its generator network for producing segmentation maps and discriminator network for feedback, SegAN contributes to advancements in medical imaging and diagnosis. Boundary equilibrium GAN (BEGAN) [139] focuses on generating high-quality face samples at a resolution of 128 $\times$ 128 pixels. It employs an architecture that achieves equilibrium between generator and discriminator networks, enabling stable training and the synthesis of visually appealing facial images. To tackle the challenge of converting low-resolution images to high-resolution, two powerful techniques come to the forefront: the multi-level densely connected super-resolution network (mDCSRN) [137] and the super resolution GAN (SRGAN). The mDCSRN takes a unique approach by employing a densely connected architecture with skip connections. By progressively upsampling the input image through multiple levels, it effectively enhances image details while ensuring the preservation and propagation of important image features. On the other hand, SRGAN [90] adopts a different strategy by combining a generator network and a discriminator network. The generator network utilizes deep residual blocks to extract and reconstruct high-frequency image details, resulting in remarkable resolution enhancement. Lastly, domain transfer networks (DTN) [89] specialize in generating images in a target domain based on different source domains. These models facilitate the seamless translation of visual content across domains, enabling applications such as style transfer, image synthesis, and data augmentation.Exploring these image-to-image techniques not only unlocks new possibilities for image manipulation and transformation but also drives advancements in the field of computer vision. These models and tasks provide invaluable tools for tasks such as image editing, deblurring, resolution enhancement, style transfer, domain transfer, and image segmentation, ultimately contributing to the progress and innovation of image processing and synthesis.

Text-to-Video

Text-to-image (T2I) models have achieved remarkable progress in generating visual content based on text descriptions. Building upon this advancement, researchers have now turned their attention to text-to-video (T2V) generation. In the early works of this field, one notable technique that emerged was temporal GANs conditioning on captions (TGANs-C). TGANs-C, based on generative adversarial networks (GANs), focused on generating video sequences from textual descriptions. Expanding to TGANs-C [172], subsequent research has led to the development of various diffusion-based models for text-to-video generation. One such technique is "Make-A-Video" by Meta AI, which directly extends the diffusion-based T2I advancements to T2V. This approach leverages paired text-image data to capture the visual appearance and descriptions of the world, while utilizing unsupervised video footage to understand motion dynamics. In the "Make-A-Video" [159] technique, the text input undergoes processing using a decoder to create image embeddings. These generated images are then interpolated to influence the frames per second in the resulting video.

Spatiotemporal layers are employed to produce high-resolution video output. Another noteworthy model in this domain is IMAGEN VIDEO [160], introduced by Google. It is a diffusion-based model that goes beyond generating simple text-guided videos, enabling the creation of text animations with diverse artistic styles. In IMAGEN VIDEO [160], textual inputs are encoded into textual embeddings using the T5 text encoder. The video diffusion model is then employed to generate a 16-frame video, which is further refined using spatial super-resolution (SSR) and temporal super-resolution (TSR) techniques. Tune-a-video [104], yet another diffusion-based model, utilizes a single text–video pair to train the T2V generator, a technique known as one-shot video tuning. This model holds immense potential for various applications, including object editing (e.g., replacing a zebra with a horse in the provided video input), background change, and style transfer. PHENAKI [119] is a video generation model that utilizes a bidirectional transformer architecture. By leveraging textual descriptions as input, it has the capability to generate video sequences. The model excels in generating videos that correspond to different time-varying text prompts, allowing for dynamic and diverse output.

Text+Video to Video

Video-to-video generative AI techniques have opened up possibilities for object editing in videos. One such technique is Dreamix [106], which is based on a video diffusion model. Dreamix [106] takes natural language descriptions and videos as inputs. The process begins by adding severe noise to the video and downscaling it to a low resolution. Then, a video diffusion model (VDM) is applied, which preserves the style of the original video while ensuring temporal consistency and fidelity in the output video. This framework has been extended to support the animation of sequences of images into videos or even the generation of animated videos from a single image.

Video to Video

Significant progress has been made in the field of video resolution enhancement, addressing the challenge of converting low-resolution video into high-quality video. One noteworthy approach, VSRResFeatGAN [91], leveraging the capabilities of generative adversarial networks (GANs) and incorporating perceptual losses, VSRResFeatGAN [91] exhibits potential in achieving remarkable video super-resolution outcomes. This model seamlessly integrates adversarial training and perceptual loss functions, resulting in enhanced video quality.

Image+Text to Image

Image+text-to-image synthesis is an approach that builds upon the foundations of image-to-text and text-to-image techniques. It offers a unique and compelling way to manipulate and generate images by leveraging both visual and textual inputs. By combining the power of an image and its accompanying textual description, this approach enables the creation of new images that align with specific styles, concepts, or artistic visions. LDEdit [49], which utilizes the latent diffusion model (LDM) to enable semantic attribute manipulation and artistic style transfer. By utilizing a common shared latent space between the input image and the target image, the LDM allows for the extraction of latent representations using an encoder. These latent representations can be manipulated and combined with textual prompts to achieve the desired image transformation using forward diffusion and reverse diffusion. Additionally, a notable technique in the field of image+text-to-image synthesis is StyleCLIP [98]. StyleCLIP [98] introduces a unique architecture that combines style transfer and clip-based image generation. It leverages a pre-trained neural network that maps textual prompts to images in a latent space. By providing a text prompt, users can manipulate the style, content, or appearance of the generated images. Figure 17 demonstrates image editing task with a textual prompt using StyleCLIP. Another notable model in the field is DIFFEDIT [164], which operates on the diffusion principle to manipulate images based on text queries. By applying noise to the

input image and masking specific regions based on text queries, DIFFEDIT [164] utilizes DDIM encoders to obtain the latent space. Using the diffusion model's decoder, changes can be applied to both the masked and non-masked regions, resulting in customized image modifications.

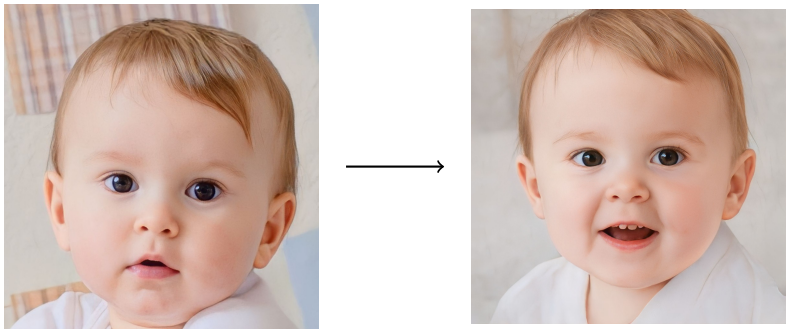

Figure 17. Input to the StyleCLIP (output image is generated using https://replicate.com/orpatashnik/styleclip, accessed on 25 July 2023) is baby image with prompt "A baby boy with smiley face." The output is a smiley baby image.

Additionally, DiffusionCLIP [131], another diffusion-based approach, has demonstrated capabilities in image manipulation, even in zero-shot scenarios. By leveraging forward and reverse DDIM processes, DiffusionCLIP [131] achieves better image transformations. Another noteworthy technique, SISGAN (synthetic image synthesis using conditional GAN) [163], incorporates conditioning both images and text. The encoder encodes the source image and accompanying text description, while the decoder synthesizes the final image based on the combined feature representations. Lastly, SteganoGAN [162], a GAN-based model, focuses on steganography, which involves hiding text within images.

3.3.7. Text-Driven 3D Content Generation

Generative AI models have revolutionized the field of image synthesis, allowing for the generation of visually compelling 2D images based on textual descriptions. However, the potential of these models extends beyond 2D images, as they can also be applied to the domain of 3D content generation. In this section, we will discuss various text-to-3D techniques.

Text to 3D Image

The field of text-to-3D-image generation has witnessed significant advancements in recent years. One notable approach, DreamFusion [133], utilizes a diffusion-based method to generate high-quality 3D images from textual descriptions. By employing a pre-trained image diffusing model and score distillation sampling, DreamFusion [133] achieves impressive results. However, challenges such as low-resolution image synthesis and the slow optimization process of NeRF (neural radiance fields) have been identified. To address these limitations, NVIDIA proposed a two-stage optimization framework called Magic3D [167]. This framework incorporates both low-resolution and high-resolution optimization steps, leading to the generation of detailed and high-resolution 3D images.

Text to 3D Animation

The synthesis of animated 3D images from textual descriptions has also gained significant attention in the research community. One notable model, AvatarCLIP [66], leverages the power of CLIP (contrastive language-image pre-training) to generate animated avatars guided by text prompts. This two-step process involves generating a static avatar based on the textual description and then animating it using motion guidance. Another approach, known as DreamFace [102], focuses on generating animated 3D faces. It involves a three-step process: geometric generation using ICT-FaceKit shape space, texture diffusion, and

animatability empowerment. Textual descriptions are utilized to generate geometric details, hair, face color, and other personalized features. Additionally, the generated faces are further animated using a neural animation approach. Addressing the need for human avatars in virtual reality (VR) and augmented reality (AR) applications, the AvatarCraft [55] model offers a comprehensive solution. It involves a two-stage process: encoding a base mesh and utilizing a text-conditioned diffusion model to generate the desired human avatar. Furthermore, the avatar can be animated using the SMPL (skinned multi-person linear) model, enabling realistic and customizable animations.

3.3.8. Text to Molecule or Molecular Structure

Generative AI techniques have expanded beyond visual content generation and are now being applied to drug discovery as well. Text2Mol [60] combines multimodal molecule retrieval by employing two submodels: a text encoder (SciBERT) and a molecule encoder (MLP-GCN). These submodels work together to create an embedding space for efficient retrieval of relevant molecules. Instead of retrieving information from existing molecules [1], researchers proposed MolT5 (molecular T5) [58], a self-supervised framework designed for molecule captioning and text-based de novo molecule generation. MolT5 utilizes a transformer architecture and achieves its desired results by scaling the pretrained data. Generative AI has pushed boundaries with the introduction of JT-VAE (junction tree variational autoencoder) [57]. This approach allows for the generation of molecular graphs with specific chemical properties. By leveraging the power of variational autoencoders and junction trees, it was designed on two encoders and two decoders which are used to decompose and encode the latent space and generate new moleulcar graphs. Furthermore, MoFlow [76] presents another approach to molecular graph generation. This technique focuses on modeling the underlying distribution of molecular graphs and employs flow-based generative models to generate new molecules. The goal is to optimize the chemical structure generation process and enhance the efficiency of drug discovery. In terms of optimizing chemical structures, researchers have proposed Mol-CycleGAN [125]. This model utilizes the CycleGAN architecture to perform cyclic transformations between two domains: real and generated molecules. By leveraging the power of generative adversarial networks, Mol-CycleGAN [125] aims to refine and optimize chemical structures, thereby aiding in the drug discovery process.

3.3.9. Tabular Data to Tabular Data

Tabular GANs [94] have emerged as powerful generative AI models for working with tabular data. These models excel in generating synthetic data that closely resembles the underlying data distribution. TGANs, or tabular GANs [94], focus on generating realistic tabular data by learning from the original data distribution. They utilize adversarial training to optimize the generator and discriminator networks, resulting in generated samples that closely match the statistical properties of the real data. On the other hand, a CTGAN, or conditional tabular GAN [83], enhances the generation process by incorporating conditional information. It allows for the generation of tabular data conditioned on specific attributes or classes, enabling more targeted and controlled data synthesis. These tabular GAN models have numerous applications, including data augmentation, privacy-preserving data sharing, and generating synthetic datasets for training machine learning models in scenarios where real data may be limited or sensitive. By leveraging the power of generative AI, tabular GANs provide valuable tools for data scientists and researchers working with tabular data.

3.3.10. Text to Knowledge Graph and Knowledge Graph to Text

Text-to-knowledge-graph and knowledge-graph-to-text are two tasks that bridge the gap between natural language understanding and knowledge representation. Text to knowledge graph involves transforming unstructured textual data into a structured knowledge graph, capturing the semantic relationships and entities within the text. One

technique used for this purpose is Grapher [117]. Grapher [117] consists of two stages. In the first stage, it leverages the power of pre-trained language models (PLMs), such as T5, to generate the nodes of the knowledge graph. PLMs have the ability to understand and extract information from text, allowing them to identify and represent the entities mentioned in the input text as nodes in the graph. In the second stage, Grapher [117] focuses on generating the edges of the knowledge graph. This is performed using recurrent neural network models like LSTM or GRU. These models take the generated nodes and the original text as input and generate the edges that capture the relationships between the nodes. The LSTM or GRU models are particularly effective in capturing long-range dependencies and contextual information, which are crucial for accurately modeling the relationships within the knowledge graph. On the other hand, in the knowledge-graph-to-text task, the aim is to generate coherent and meaningful textual representations from knowledge graphs. One approach for this task is the GTR-LSTM [129], which is a technique for generating sentences from RDF data. It utilizes a triple encoder architecture consisting of subject, predicate, and object encoders, which capture the semantic information of the RDF triples.

3.3.11. Road Network to Road Network

StreetGAN [97] is a generative AI model designed to synthesize realistic road networks. It employs deep learning and generative adversarial networks (GANs) to generate road layouts that closely resemble real-world road systems. By training on existing road network data, StreetGAN [97] learns the underlying patterns and structures, enabling it to create new synthetic road networks with similar statistical properties. The architecture involves a generator network that produces road network layouts and a discriminator network that distinguishes between real and synthetic road networks. Through iterative training, StreetGAN [97] refines its generator to generate high-quality road networks. The synthesized road networks have various applications, including simulation, testing traffic management strategies, training machine learning algorithms, and aiding urban planning and transportation development.

3.4. Classification of Evaluation Metrics for AIGC Models

Evaluating the performance of generative AI techniques has become increasingly important as these models continue to advance in complexity and capability. With applications ranging from natural language processing to computer vision and creative arts, assessing the quality and effectiveness of generative AI systems has become crucial for ensuring their reliability and usefulness in various domains. In this section, we will discuss various evaluation metrics.

Figure 18 represents categorization to have a clear understanding of different metrics used to assess the effectiveness of generative AI models across various tasks. The abbreviations of the metrics are given in Table A4. The root node represents the output type generated by the generative AI models for the given input. The intensities of the colors in level 1 and level 2 represent the same subtree. Leaf nodes, which are white in color, represent the metrics.

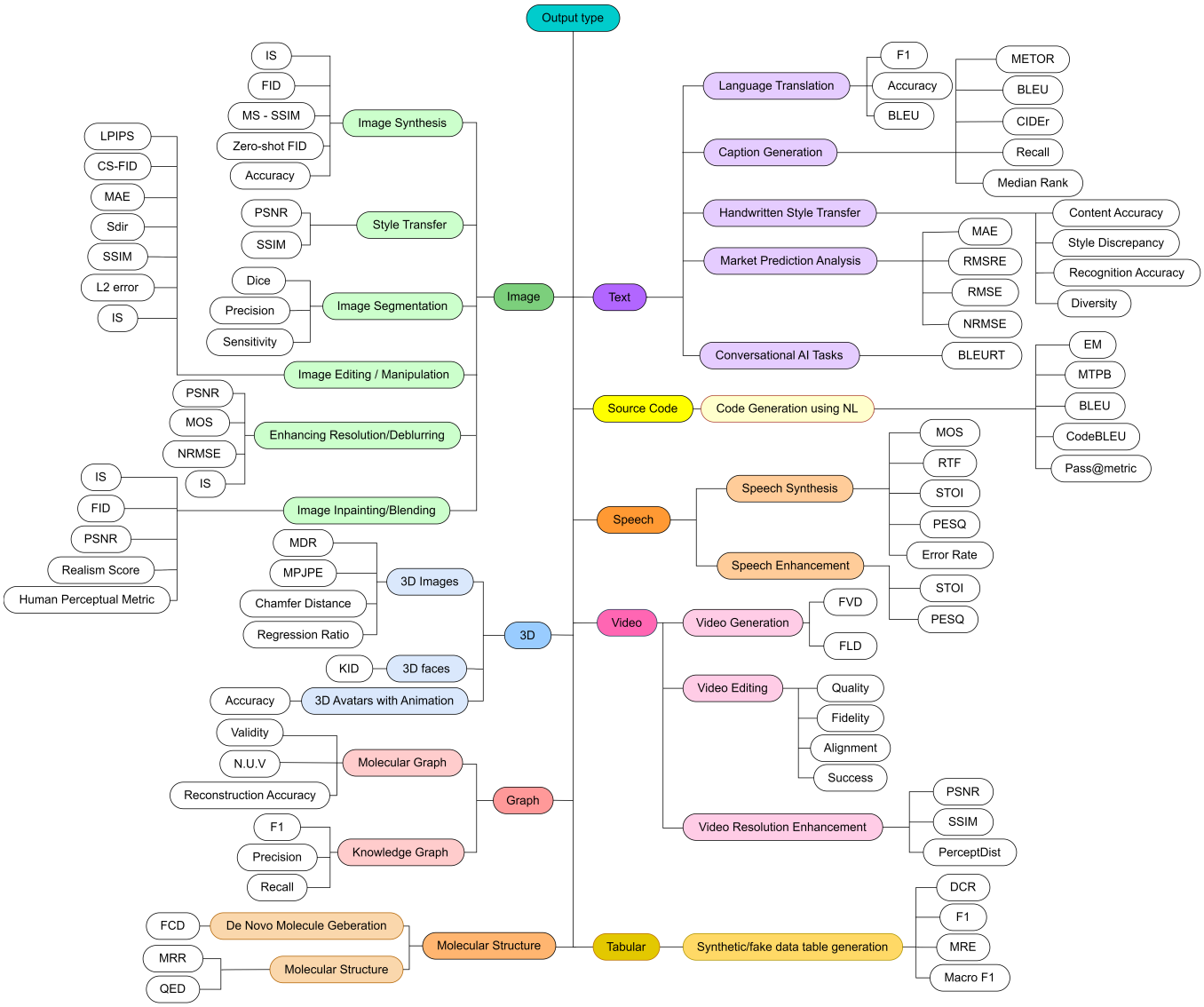

Figure 18. Evaluation metrics classification diagram for various output types.

3.4.1. **Evaluation Metrics for Image Processing:**

Inception Score (IS): The inception score (IS) is a widely used metric for evaluating the quality of generated images in generative adversarial networks (GANs). It measures both the diversity and realness of the generated images. The IS is calculated by utilizing the inception model and computing the KL divergence between the conditional class distribution of the generated samples and the marginal class distribution. In addition to IS, several models mentioned earlier utilize this metric for assessing the quality of generated images. These models include text-conditioned auxiliary classifier generative adversarial network (TAC-GAN) [84], attentional generative adversarial networks (AttnGAN) [92], CogView [77], and stacked generative adversarial networks (StackGAN) [93].

Structural Similarity Index (SSIM): The structural similarity index (SSIM) is a metric for image quality assessment that considers changes in structural information as the primary factor influencing visual quality. Unlike other methods that focus on the errors between the original and compressed image, SSIM considers changes in structural information as the main element affecting visual quality. This metric has found applications in various image and video processing models, including DiffusionCLIP [131], mDCSRN [137], alignDRAW [78], SRGAN [140], and VSRResFeatGAN [91]. These models utilize SSIM to

assess the quality of reconstructed images or videos by comparing them to their respective originals.

FID (Fréchet Inception Distance): The Fréchet inception distance (FID) is a metric used to measure the similarity between two sets of images. It is often used to assess the quality of images generated by generative adversarial networks (GANs), comparing the distribution of generated images to the distribution of real images. The lower the FID, the closer the two distributions, and therefore the better the GAN model. The FID is based on the Fréchet distance between two multivariate Gaussian distributions. Given the two Gaussian distributions of real data (with mean $\mu 1$ and covariance $\Sigma 1$) and generated data (with mean $\mu 2$ and covariance $\Sigma 2$). Various image-related task models, such as DDPM [101], BigGAN [43], and Muse [150], as well as video-generation models, like PHENAKI [119], have utilized FID as a metric for their respective evaluations.

Zero-Shot FID (Fréchet Inception Distance): Zero-shot FID (Fréchet inception distance) is a modified version of the FID metric that extends its application to evaluate the quality of images generated in a zero-shot setting. In the zero-shot scenario, the generated images do not belong to any predefined training classes or categories. Instead, they are produced based on textual descriptions or other forms of input that are distinct from the training data. This metric has been applied in various image synthesis techniques, including GLIDE [147] for image synthesis and IMAGEN Video [160] for video generation.

Multi-Scale Structural Similarity Index Measure (MS-SSIM): The multi-scale structural similarity index measure (MS-SSIM) is a method for comparing the similarity between two images. It is an extension of the structural similarity index measure (SSIM), a popular metric for image quality assessment that is based on the degradation of structural information. While SSIM operates on a single scale, MS-SSIM evaluates similarity at multiple scales, which can provide a more robust and comprehensive measurement. This can be particularly helpful when dealing with images that have variations in size, resolution, or viewing conditions. The end result is a single value ranging from -1 to 1, with 1 indicating perfect similarity, 0 meaning no similarity, and negative values suggesting inverse similarity. Several image synthesis models, including TAC-GAN [84], have utilized MS-SSIM as a means to evaluate the quality of synthesized images.

Accuracy: Accuracy is a metric commonly used in machine learning and classification tasks. It measures the correctness of predictions or classifications made by a model. It is calculated as the ratio of the number of correct predictions to the total number of predictions. Accuracy is often expressed as a percentage. In the context of image-related tasks, accuracy is a commonly used metric to assess the performance of models, such as domain transfer network (DTN) [89] and TGAN [67], for tasks like image classification, image segmentation, or object detection.

F1 Score: The F1 score is a measure of a model's accuracy that takes into account both precision and recall. It is commonly used in machine learning and statistical analysis to evaluate the performance of classification models, particularly in imbalanced datasets where one class may dominate the others. The F1 score is calculated as the harmonic mean of precision and recall: F1 score = 2 * (precision * recall)/(precision + recall).

Learned Perceptual Image Patch Similarity (LPIPS): Traditional metrics like SSIM and PSNR were effective at measuring numerical image similarity, but they are not aligned with human visual perception. To improve this, LPIPS was introduced to better model human visual perception. DiffusionCLIP [131] and DIFFEDIT [164] are two models we discussed in our paper, and we have chosen one of their metrics as LPIPS to evaluate images manipulated by textual prompts.

MAE (Mean Absolute Error): Mean absolute error (MAE) is generally used for regression tasks, but it can also be used as a loss function for image reconstruction, like manipulation of images or editing images. One such model, called DiffusionCLIP [131], uses MAE to evaluate reconstructed image quality. A lower MAE indicates a better reconstruction of the image.

Directional CLIP Similarity (Sdir): This metric is used for evaluating the performance of text-driven image manipulation changes among the two images in a CLIP space. A higher Sdir score indicates the two images were more consistent, and text-based edit instructions have been successfully carried out by the model; DiffusionCLIP [131] is one such model that uses Sdir as one of its evaluation metrics.

L2 Error: L2 error in the context of image manipulation is used as a component of the loss function to measure the difference between the generated image and the target image so that the generated image to be closed as possible to the target image in the pixel space.

Dice: It is also known as the dice similarity coefficient and is used for evaluation of segmentation by generative AI models. One such model, called SegAN [100], which is useful in medical applications to segment the X-ray scans of patients, is now used to evaluate these segmentations. The dice score is 1 for perfect segmentation and 0 for completely non-overlapping masks.

Precision: In the context of image segmentation, precision is the proportion of true positive pixels, which indicates whether the pixels were identified as belonging to a certain class out of all pixels that were identified as belonging to that particular class. A higher precision indicates fewer false positives.

Sensitivity: Is also known as recall or true positive rate, which measures the proportion of true positive pixels to actual positive pixels. SegAN [100] used sensitivity as one of the metrics to evaluate the segmentation performed by the model.

PSNR (Peak Signal-to-Noise Ratio): PSNR is one of the metrics used to measure the quality between the original image and the deblurred image. In the context of image deblurring, DeblurGAN [61] and DeblurGAN-v2 [56] use PSNR as one of their metrics to evaluate the image reconstruction quality.

Normalized Root Mean Square Error (NRMSE): The normalized root mean square error (NRMSE) is a performance metric used to evaluate the accuracy of predictive models, particularly when numerical outputs are involved. It is essentially the root mean square error (RMSE) adjusted for scale, which makes it useful for comparing datasets. NRMSE is computed as the ratio of RMSE to the dispersion of actual data values, which is typically expressed as a percentage. A lower NRMSE indicates that the model's predictions are closer to the actual values. mDCSRN [137] used this metric to evaluate the high-resolution image generated by the model.

Mean Opinion Score (MOS): The mean opinion score (MOS) is a subjective metric often used in image processing tasks like deblurring or resolution enhancement. It gauges the quality of an image based on human perception. Scores are assigned by a group of human evaluators on a predefined scale, such as 1 to 5, with 'bad' to 'excellent' quality. SRGAN [90] uses this metric as one of the evaluation metrics to evaluate the generated image HR Image.

Fully Convolutional Network Score (FCN-Score): The FCN-Score is an evaluation metric used to assess the quality of samples generated by models such as GANs like CGAN [81]. Leveraging the fully convolutional network (FCN) architecture, it extracts features from real and generated images and measures the differences in their distributions. Lower FCN-Scores suggest that the generated images closely resemble the real ones, signifying a well-performing model, while higher scores indicate a deviation from the real data, pointing to a less effective model.

Realism Score: The realism score is a standard for evaluating the quality of images after they have undergone modifications such as inpainting or blending. GP-GAN [134] is one such model that uses this metric to evaluate its output images. This scoring system evaluates how closely these altered images mimic their unmodified, natural counterparts. So, when an image scores highly on the realism score, the modifications have been integrated so seamlessly that it becomes nearly impossible to distinguish the final result from the original image.

Human perceptual metric: Human perceptual metrics are key in image processing and computer vision, offering a benchmark that aligns closely with human visual perception. These metrics quantify an image's quality or distinct features or visual result based on the

human perceptual understanding of colors, shapes, patterns, and intricate details. Models like GCC-GAN's [64], LDEdit [49], and TAGAN [67] use the human perceptual metric as one of their evaluation metrics.

3.4.2. Evaluation Metrics for NLP Tasks

F1 Score: It is a measure of a test's accuracy that considers both precision (the number of correct positive results divided by the number of all positive results) and recall (the number of correct positive results divided by the number of positive results that should have been returned). The F1 score is the harmonic mean of precision and recall. The formula is F1 = 2 * (precision * recall)/(precision + recall). The T5 Model [39] used the F1 score as one of the evaluation metrics for the translation tasks.

*BLEU (Bilingual Evaluation Understudy):*This is a metric for evaluating the similarity between a generated sentence and a reference sentence. A perfect match yields a score of 1.0. It is widely used not only in machine translation tasks but also in text generation applications. Various language translation and image captioning AI models, including T5 [39], ConvS2S [132], Seq2Seq [108], MusCaps [48], "Show and Tell: Neural Image Captioning" [59], WT5 [121] and RTT-GAN [62], have incorporated BLEU as one of their key evaluation metrics.

Recall: Also known as sensitivity or true positive rate (TPR), it is the fraction of the total amount of relevant instances that were actually retrieved. It can be thought of as the ability of a model to find all the relevant cases within a dataset. For image captioning tasks, generative AI models like MusCaps [48] use recall as one of their evaluation metrics.

METEOR (Metric for Evaluation of Translation with Explicit Ordering): It is an evaluation technique designed specifically for machine translation. This method calculates the harmonic mean of unigram precision and recall, where recall carries more weight than precision. What sets METEOR apart from other evaluation metrics is its inclusion of additional features like stemming and synonym matching in addition to the traditional exact word matching. Conversational AI models like GPT-3 and T5 [112] and image captioning models like MusCaps [48], GTR-LSTM [129], and Show and Tell: Neural Image Captioning [59] have used METEOR as their evaluation metrics.

CIDEr (Consensus-based Image Description Evaluation): It is a metric designed to evaluate the quality of image descriptions generated by models like RTT-GAN [62] and MusCaps [48] that capture the consensus among human references.

Median Rank: The median rank is a frequently employed metric in retrieval tasks. In such tasks, a system receives a query and retrieves a list of items ranked according to their relevance. The rank of the first item that is deemed relevant is recorded, and the median of these ranks across a collection of queries is then calculated to determine the Median Rank.

EM-Diff (Exact Match Difference): EM-Diff is an evaluation metric commonly used in conversational AI tasks, particularly in question-answering systems. It measures the difference between the exact match scores of two systems on a given dataset. The exact match score represents the proportion of responses that exactly match the ground-truth answers. EM-Diff provides insights into the improvement or degradation of the exact match performance between different models. Conversational AI models like PEER [34] use EM-Diff as one of their evaluation metrics.

BLEURT (Bilingual Evaluation Understudy for Natural Language Understanding in Translation): BLEURT is an evaluation metric specifically designed for assessing the quality of machine translation systems, question-answering models like InstructGPT [111]. It measures the similarity between human-generated reference translations and machine-generated translations. BLEURT is trained using a combination of deep learning techniques and is capable of providing a continuous score that represents the quality of translations. A higher BLEURT score indicates better translation quality.

Root Mean Squared Relative Error (RMSRE): RMSRE is an evaluation metric used to measure the accuracy of predictions or forecasts for models like GAN-FD [156]. It is a variant of the root mean squared error (RMSE) that takes into account the relative

difference between predicted and actual values. The RMSRE calculates the square root of the average of the squared relative errors between the predicted and actual values. It provides an indication of the average percentage error in the predictions, with a lower RMSRE indicating better accuracy.

Content Accuracy: Content accuracy measures the degree to which the generated handwritten characters match the content or shape of the target font style or new font which are generated by models like DenseNet CycleGAN [54]. It assesses the ability of the model to capture the fundamental structure and characteristics of the characters accurately.

Style Discrepancy: Style discrepancy evaluates the difference or dissimilarity between the generated handwritten characters and the target font style or new font in terms of style variations. It measures the ability of the model to capture the specific stylistic details, variations, and nuances of the font style. DenseNet CycleGAN [54] has used this to evaluate the generated font.

Recognition Accuracy: Recognition accuracy assesses the ability of the generated handwritten characters to be recognized correctly by optical character recognition (OCR) systems or other recognition models. It measures how well the models like GlyphGAN [154] generate characters that resemble the original symbols and can be accurately identified.

Diversity: Diversity measures the range and variation of the generated handwritten characters. It assesses the model's ability to produce distinct and diverse characters within the target font style or new font, avoiding repetitions or excessive similarities.

3.4.3. Performance Metrics for Code Generation Models

CodeBLEU: CodeBLEU is a metric specifically designed to evaluate the quality of machine-generated code in code generation tasks, particularly in the context of natural language description-to-code generation. It extends the BLEU metric, commonly used in machine translation tasks, to measure the similarity between the generated code and the reference code. Notably, models such as CodeBERT [114] and CodeT5 [115] have utilized the CodeBLEU metric to assess how closely the generated code aligns with the reference code.

Exact Match (EM): Exact match is a binary metric commonly used in code generation tasks to determine whether the generated code perfectly matches the desired output or reference code. It provides a measure of accuracy by evaluating if the generated code is an exact match. CodeT5 [115] models, for example, often employ the exact match metric to assess the level of accuracy in generating code that aligns precisely with the reference code.

Pass@metric: Pass@metric is a widely used performance evaluation metric in code generation tasks, particularly for evaluating language models or systems that generate code based on natural language descriptions. The metric measures the percentage of generated code that successfully passes a given set of evaluation tests or test cases. It provides insights into the system's ability to produce functionally correct code that meets the desired specifications. Models like GPT language model (Codex) [116] and pre-trained transformer-based language models like Alphacode [35] have been evaluated using the Pass@metric to assess the success rate of the generated code.

Multi-Turn Programming Benchmark (MTPB): MTPB is an evaluation benchmark employed to assess the performance of code generation systems like CODEGEN [50], specifically in multi-turn programming scenarios. This benchmark involves multiple interactions between a human and the code generation system, where the human provides natural language descriptions, queries, or instructions, and the system generates code accordingly. The quality of the generated code is evaluated based on correctness, efficiency, and adherence to the specified requirements.

3.4.4. Evaluation Metrics for Various Graph Generation Models

Validity Metric: The validity metric is a crucial evaluation measure in graph generation systems. It assesses the extent to which the generated graphs comply with the rules and constraints of a particular domain. For instance, in the context of generating molecular

graphs or designing molecule graphs, validity ensures that the generated structures adhere to the chemical bonding rules and other molecular properties. The junction tree variational autoencoder (JT-VAE) [57] model is an example of a graph generation model that utilizes the validity metric. By leveraging a junction tree representation and variational autoencoder framework, the JT-VAE [57] aims to generate molecular graphs that are structurally valid and consistent within the chemical context it operates in.

Reconstruction Accuracy: Reconstruction accuracy is a significant evaluation metric that measures how faithfully the generated graphs reproduce the original input or reference graphs. It assesses the ability of the graph generation system to capture the essential features and characteristics of the input data. The JT-VAE [57] model, mentioned earlier, not only focuses on generating valid molecular graphs but also places emphasis on accurate reconstruction of the input molecular structures. By striving to capture the molecular features and properties as accurately as possible, the JT-VAE [57] aims to achieve high reconstruction accuracy.

N.U.V (Novel, Unique, and Valid Molecules): The N.U.V metric evaluates the novelty, uniqueness, and validity of generated molecules. It measures the degree to which the generated molecular structures are novel (different from existing molecules in a given dataset), unique (not redundant or similar to other generated molecules), and valid (adhering to the chemical rules and constraints). The MoFlow [76] model is an example of a graph generation system that incorporates the N.U.V metric in its evaluation criteria. MoFlow [76] focuses on generating molecular graphs based on chemical properties and aims to produce molecules that are both novel and unique while also maintaining validity in accordance with the underlying chemical knowledge.

3.4.5. Evaluation Metrics in Molecular Structure Generation

MRR (Mean Reciprocal Rank): Mean reciprocal rank (MRR) is an evaluation metric employed in the context of generating or retrieving molecular structures from textual descriptions. It measures the average reciprocal rank of the correct answer among a set of possible answers. This metric is commonly used in the Text2Mol [60] technique, which focuses on generating or retrieving molecular structures based on textual descriptions. By utilizing MRR, the Text2Mol [60] technique can assess how well the generated structures align with the given textual descriptions, ultimately ranking the correct molecular structure higher among the generated options.

QED (Quantitative Estimate of Drug-Likeness): Quantitative estimate of drug-likeness (QED) is an evaluation metric extensively used in molecular graph generation and design, particularly in the domain of molecular design and drug discovery. QED provides a quantitative measure of the drug-likeness of a generated molecule, indicating its potential to be developed into a safe and effective drug. The CGVAE [79] technique, employed in molecular design and drug discovery tasks, incorporates QED as an evaluation metric. The technique leverages QED to assess the drug-likeness of generated molecules, considering various molecular properties such as stability, solubility, and bioactivity. Higher QED values indicate a higher likelihood of a molecule being considered drug-like, facilitating the prioritization of molecules with greater potential.

FCD (Fréchet ChemNet Distance): Fréchet ChemNet distance (FCD) is an evaluation metric commonly utilized in text-based de novo molecule generation and molecule captioning tasks. FCD quantifies the similarity between the distributions of molecular structures generated by a model and a reference distribution, typically a dataset of known molecules. The MolT5 (Molecular T5) [58] technique, known for text-based de novo molecule generation and molecule captioning, incorporates FCD as an evaluation metric. By leveraging FCD, the MolT5 [58] technique can assess the similarity of generated molecules with the reference distribution in terms of chemical features and properties. A lower FCD value indicates a higher similarity between the generated molecules and the reference distribution, indicating better quality and alignment with the desired chemical characteristics.

3.4.6. Evaluation Metrics for 3D Generation and Animation Techniques

Matching Distance Ratio (MDR): Matching distance ratio is an evaluation metric used in the context of inpainting 3D images. It quantifies the quality of the generated inpainted regions by calculating the ratio of points that are accurately matched between the generated and ground-truth images. The point encoder GAN [161] model, employed for inpainting 3D images, utilizes the matching distance ratio as an evaluation metric to assess the effectiveness and accuracy of the inpainted regions. A higher matching distance ratio indicates a higher level of accuracy in matching the points between the generated and ground truth images, indicating a better quality of inpainted regions.

MPJPE (Mean per Joint Positioning Error): MPJPE, or mean per joint positioning error, is an evaluation metric commonly used in the context of generating realistic human poses. It measures the average positional error between the joints of the generated pose and the ground-truth pose. GAN-Poser [51], a model for generating realistic human poses, employs MPJPE as an evaluation metric to assess the accuracy and realism of the generated poses. A lower MPJPE value indicates a higher level of pose accuracy and alignment with the ground truth.

Chamfer Distance: Chamfer distance is an evaluation metric utilized in text-guided 3D object generation tasks. It quantifies the dissimilarity between the generated 3D objects and the target objects based on their point cloud representations. The DREAMFUSION [133] model employs Chamfer distance as an evaluation metric to measure the dissimilarity between the generated 3D objects and the target objects described in textual descriptions. A lower Chamfer distance value indicates a higher level of similarity and alignment with the desired object.

Regression Ratio: Regression ratio is an evaluation metric used in the context of inpainting 3D images. It quantifies the accuracy of the generated inpainted regions by calculating the ratio of accurately regressed points. The point encoder GAN [161], a model for inpainting 3D images, utilizes regression ratio as an evaluation metric to assess the accuracy and quality of the generated inpainted regions. A higher regression ratio indicates a higher level of accuracy and fidelity in regressing the missing parts of the 3D images.

KID (Kernel Inception Distance): Kernel inception distance (KID) is an evaluation metric employed in the context of generating personalized animatable 3D faces based on text guidance. It measures the distance between the feature representations of the generated faces and the real faces using a pre-trained kernelized inception network. The DreamFace [102] model utilizes KID as an evaluation metric to assess the similarity and realism of the generated 3D faces.

3.4.7. Evaluation Metrics for Synthetic Data Generation in Tabular Datasets

DCR (Distance to the Closest Record): DCR, or distance to the closest record, is an evaluation metric used in the context of synthesizing fake tables that are statistically similar to the original table. It measures the distance between the generated synthetic records and the closest corresponding records in the original table. The table-GAN [52] model employs DCR as an evaluation metric to assess the similarity and proximity of the generated records to the original data.

Macro-F1: Macro-F1 is an evaluation metric that assesses the performance and similarity of generated synthetic datasets to the original data. It calculates the F1 score for each class or label in the dataset and then takes the average across all classes. The Tabular GAN (TGAN) [94] model utilizes macro-F1 as an evaluation metric to assess the overall performance and similarity of the generated synthetic dataset to the original data.

Mean Relative Error (MRE): Mean relative error (MRE) is an evaluation metric employed in synthesizing synthetic data for tabular datasets tasks. It quantifies the average relative error between the generated synthetic data and the original data, taking into account the magnitude of the errors. The table-GAN [52] model utilizes MRE as an one of its evaluation metric to assess the accuracy and similarity of the generated synthetic data to the original dataset.

3.4.8. Evaluation Metrics for Speech and Audio Generation Techniques

RTF (Real-Time Factor): RTF, or real-time factor, is an evaluation metric used in speech synthesis tasks to measure the efficiency of the synthesis process. It quantifies the speed at which speech is synthesized in real-time. The Grad-TTS [152] model incorporates RTF as an evaluation metric to assess the efficiency and speed of the text-to-speech synthesis process.

PESQ (Perceptual Evaluation of Speech Quality): PESQ, or perceptual evaluation of speech quality, is an evaluation metric commonly used in speech enhancement tasks. It quantifies the perceived quality of enhanced speech signals by comparing them to the original, clean speech signals. The conditional generative adversarial networks (CGAN) [82] model with the Pix2Pix framework utilizes PESQ as an evaluation metric to assess the improvement in speech quality achieved through the speech enhancement process, and also ProDiff [53] utilizes this metric to evaluate the generated speech. A higher PESQ score indicates a higher level of perceived speech quality and fidelity.

STOI (Short-Time Objective Intelligibility): STOI, or short-time objective intelligibility, is an evaluation metric used to measure the intelligibility of speech signals. It assesses the degree to which speech can be understood by comparing the enhanced speech signals to the original, clean speech signals. The CGAN [82] model with the Pix2Pix framework also employs STOI as an evaluation metric to evaluate the improvement in speech intelligibility achieved through speech enhancement.

Error Rate: Error rate is an evaluation metric commonly used in speech synthesis tasks, specifically in the generation of speech from text. It measures the accuracy of the generated speech by comparing it to the desired or target speech. The feed-forward transformer (FFT) [42] model utilizes the error rate as an evaluation metric to assess the accuracy and fidelity of the synthesized speech.

4. Discussion

4.1. AIGC Requirements

The hardware requirements for the generative AI aspect plays a vital role in data collection, model training, and sample generation. The findings highlight the diverse options available for hardware, including cameras, microphones, sensors, and existing datasets for data collection. For model training and optimization, powerful hardware configurations like Tesla V100 16 GB [47] and RTX 2080Ti [49] are commonly used, while smaller-scale models can be trained on more modest configurations. Sample generation can be achieved even on basic hardware setups. Regarding software requirements, various tools and frameworks are essential for different phases of generative AI. Web scraping frameworks [33–35], Pandas [48,52,53], Numpy [54], and torch-audio [48] are used for data collection and preprocessing. Deep learning frameworks such as PyTorch [64] and TensorFlow [65] provide support for model training and evaluation, while libraries like opencv Python [55,66], NLTK [59,67] aid in post-processing and model refinement.

Generative AI models must meet diverse user experience requirements to ensure user satisfaction. Users expect high-quality and realistic outputs [69], customizable and controllable generative processes, diverse [70] and novel results, efficient performance, interactive capabilities, ethical considerations [71], data privacy and security [72,73], and seamless integration with existing systems. By addressing these requirements, developers can create generative AI models that deliver exceptional user experiences.

4.2. AIGC Models

Variational autoencoders (VAEs) provide a strong framework for learning compressed representations of incoming data and creating new samples. Their benefits include the capacity to capture data uncertainty, provide a continuous latent space for interpolation, and provide efficient data representation [19]. However, as compared to other models, VAEs may have limits in generating extremely realistic and varied samples, resulting in fuzzy or less detailed outputs. VAEs are especially useful in sectors where data structure

and uncertainty are important, such as image generation [77], data compression, and anomaly detection.

Generative adversarial networks (GANs) are excellent at producing realistic and varied samples, making them excellent for tasks requiring high visual quality. GANs provide versatility in data generation across domains. GANs, on the other hand, can be difficult to train and keep stable, requiring precise hyperparameter adjustment. Mode collapse, in which the generator fails to capture the whole data distribution, can also occur. Despite these problems, GANs are nevertheless quite effective in image synthesis [90,92,93], image-to-image translation [87–89], and providing synthetic training data [83,94] for deep learning models.

Diffusion Models are well-known for producing high-quality samples with fine features and realistic textures. They provide more control over sample quality by modifying the number of diffusion steps, and they are less prone to mode collapse than GANs. Training diffusion models, on the other hand, may be computationally expensive, and tuning hyperparameters is critical for their success. Diffusion Models excel in sectors requiring high-resolution image generation [49], creative image stylization, video generation [104,106], and image inpainting, where the emphasis is on delivering aesthetically stunning and detailed outputs.

Transformers, with their self-attention mechanism, efficiently capture long-term dependencies and have revolutionized natural language processing tasks. By creating coherent and contextually appropriate sequences, they excel at tasks like machine translation [45,110], text generation [39,116], and sentiment analysis [111,121]. However, because of memory limits, transformers can struggle with very lengthy sequences, and training large-scale models can be computationally costly. Transformers excel in sectors where comprehending global context and producing high-quality language sequences are essential.

Language models, which are frequently built on RNNs, have the benefit of producing cohesive and contextually appropriate sequences. They extract grammar, semantics, and style from training data, which makes them useful for applications like text generation [107], chatbots [33], and language understanding [108] . Language models, on the other hand, might be subject to training data biases and may output language that reflects such prejudices. Long sequences can also result in a lack of cohesion and meaningful context. Language models are useful in areas where creating natural language sequences is critical, such as conversation systems [34], text generation [107], and language modeling [109].

Normalizing flow models provide a versatile framework for producing samples and modeling complicated distributions. They are very good at density estimation and enable accurate likelihood calculation [21]. Normalizing flow models, on the other hand, can be computationally costly, especially when dealing with a high number of coupling layers or complicated data distributions. They discover efficiency in disciplines including generative modeling, density estimation, and simulation-based inference, where correct modeling of complicated data distributions is critical [76].

Hybrid models incorporating numerous deep learning architectures have been created to leverage on the strengths of distinct models in various areas. The combination of variational autoencoder and generative adversarial networks (VAE-GAN) gives the capacity to encode and decode data while creating different samples for image generation [122], image editing [124] and generating graphs [125] . GANs with dense convolutional neural network (DenseNet) or residual neural network (ResNet) designs have extensive feature learning capabilities and have demonstrated outstanding performance in image-to-image translation [54,126]. Combining GANs with recurrent neural networks (RNNs) or convolutional neural networks (CNNs) delivers coherent and realistic outputs for sequence generating challenges [127]. GANs paired with denoising diffusion probabilistic models (DDPM) and transformers provide sample generation with both diversity and quality [53]. Transformers in conjunction with RNNs allow for successful sequential data modeling, but transformers in conjunction with graph convolutional networks (GCNs) capture both sequential and graph-based interactions [58,60]. Vision transformers in conjunction with ResNet improve

image comprehension [130], while diffusion probabilistic models in conjunction with contrastive language-image pretraining (CLIP) allow for controlled image editing [131]. In sequence-to-sequence challenges, convolutional sequence-to-sequence learning (ConvS2S) blends CNNs and local dependency capture [132]. These hybrid models address the constraints and problems of separate designs, but efficiency, computing resources, training stability, and optimization must all be taken into consideration.

4.3. Input–Output Formats for Prescribed Tasks

The capabilities of text-to-text generative AI models encompass a wide range of tasks. These models excel in multilingual communication by enabling accurate translation of text from one language to another [39,108,132]. They also offer a unique tool for design and personalization, allowing the generation of handwritten characters in various font styles based on text input [154]. In the domain of programming, these models contribute to code quality and efficiency by providing precise and meaningful corrections for code issues [45]. Additionally, they possess the ability to offer insights and clarifications by explaining given input statements [121]. Leveraging input texts, these models demonstrate exceptional performance in tasks such as translation, question answering, classification, and summarization [39]. While these models provide valuable functionality, caution must be exercised to prevent misuse, such as generating or cracking passwords [155], emphasizing the need for responsible use and ethical considerations. Furthermore, these models function as conversational AI systems, engaging in interactive conversations, answering follow-up questions, challenging incorrect assumptions, and rejecting inappropriate requests [111]. They also streamline the writing process by generating drafts, suggesting edits, and providing explanations for their actions, thus assisting with various writing tasks [34]. Lastly, by leveraging historical data, these models contribute to market analysis and forecasting, enabling the prediction of future market trends [156–158].

A wide range of tasks can be accomplished in the areas of text-to-speech/audio, speech-to-speech, music-to-text, and audio-to-text generation. These tasks include generating audio from text using techniques such as w2v-BERT [120] and music from text using Jukebox (vector quantized variational autoencode, VQ-VAE) [145]. Speech synthesis methods like adaptive text to speech (AdaSpeech) [113] allow for the customization of voices and the conversion of text into human-like speech. Notable approaches like denoising diffusion model for text to speech (Diff-TTS) [63], Grad-TTS [152], ProDiff [53], DiffGenerative adversarial networks (GAN)-TTS [128], and pixel convolutional neural network (Wavenet [123]) offer advancements in text to human-like speech synthesis. Additionally, generative AI models can generate robotic voice speech from text using methods like feed-forward transformer (FFT). Within the domain of music/audio, techniques such as generative adversarial networks synth (GANSynth) [153] enable the creation of high-quality, synthetic musical audio clips. Speech enhancement techniques, such as conditional generative adversarial networks (CGAN) with the Pix2Pix framework [82], can be employed to improve speech quality. Furthermore, there are models like MusCaps [48] for generating captions for music audio and contrastive language-audio pretraining (CLAP) for generating captions for audio [47]. These diverse tasks demonstrate the vast potential and versatility of generative AI models in various audio and speech-related applications.

It is essential to address the range of tasks that generative AI models can accomplish in the field of code generation. These tasks include generating valid programming code using natural language descriptions, which can be achieved through models such as CodeBERT [114], CODEGEN [50], CodeT5 [115], and GPT language model (Codex) [116]. These models excel in converting natural language descriptions into executable code. Additionally, the Alphacode [35] model is specifically designed to generate competition-level code, demonstrating its capability to produce high-quality code solutions. Furthermore, models like code-understanding BERT (CuBERT) [110] contribute to code completion, bug detection, and code summarization, providing valuable assistance to developers in understanding and optimizing their code. The diverse functionalities of generative AI models in

the domain of code generation highlight their potential in automating programming tasks and enhancing code development processes.

Generative AI models offer a wide range of capabilities in the domain of image-to-image tasks. These tasks include blind motion deblurring of a single photograph, achieved through models like DeblurGAN [61] and DeblurGAN-v2 [56]. Generative models such as StyleGAN [4] can generate highly realistic and diverse synthetic images. Image blending can be accomplished using Gaussian-Poisson generative adversarial network (GP-GAN [134]), while image compositing and blending can be achieved with geometrically and color consistent GANs (GCC-GANs) [64]. Models like exemplar GANs (ExGANs) [99], contextual attention generative adversarial networks (CA-GAN) [127], and PGGAN [126] excel in image inpainting or filling absent pixels. For face-related tasks, models like Age-cGAN [135], Conditional Adversarial Autoencoder (CAAE) [122], and identity-preserved conditional generative adversarial networks (IPCGANs) [136] are used for face aging and image editing. Other tasks include medical image analysis with SegAN; image super-resolution using mDCSRN [137] and SRGAN [90]; domain transfer using CGAN [81], CycleGAN [87], and DiscoGAN [88]; efficient texture synthesis with Markovian generative adversarial networks (MGANs) [141]; and cyber intrusion and malware detection using defense-generative adversarial networks [95]. These tasks demonstrate the versatility and effectiveness of generative AI models in various image-to-image applications.

As discussed in Table 5, generative AI models have shown significant advancements in image-to-text tasks, offering a wide range of applications in real-world scenarios. The recurrent topic-transition generative adversarial network (RTT-GAN) [62] has proven effective in generating textual descriptions for images, which can find practical use in content analysis, image indexing, and automated report generation. The "Show and Tell: Neural image captioning" [59] model provides a valuable solution for image-to-text generation and captioning, enabling applications such as automatic image description for visually appealing presentations or enriching image-based search results with descriptive metadata. The DenseNet CycleGenerative adversarial networks (GAN) [54] demonstrate the ability to generate handwritten characters in specific font styles, presenting opportunities for personalized and creative content creation in areas like artistic designs, branding materials, or customized digital assets. Moreover, the visual language model Flamingo [109], with its question-answering capabilities based on image input, holds promise for visual assistance systems, interactive image-based tutorials, or intelligent image search engines. These applications highlight the versatility and benefits of generative AI models in enhancing creativity, accessibility, and user experiences across various domains.

Generative AI models have significantly expanded the possibilities of video-to-video tasks. With models like MAKE-A-VIDEO [159], IMAGEN [160], and Tune-A-Video [104], users can now create text-guided videos that cater to their specific needs. These advancements open up avenues for personalized storytelling, dynamic marketing campaigns, and interactive educational content. Additionally, VSRResFeatGAN [91] offers a solution for enhancing video resolution, which has practical applications in industries such as video generation, surveillance, and online meetings where clear visuals are essential. Another noteworthy model, the video diffusion model (VDM) Dreamix [106], introduces exciting features like video editing based on text input and animated image-text combinations. These capabilities enable users to unleash their creativity and find applications in various industries like advertising, entertainment, and communication.

Image+text-to-image synthesis with generative AI models offers a range of powerful capabilities. These include text-guided image manipulation through methods like LDEdit [49] and StyleCLIP [98], text-based image synthesis with SISGAN [163], and diffusion-based approaches like DIFFEDIT [164] and DiffusionCLIP [131]. Additionally, there are tasks that involve combining images and text for specific purposes. SteganoGAN focuses on generating steganographic images, where messages can be hidden within the image itself. This technique can be useful for secure communication or embedding information in visual content. These techniques enable users to perform tasks such as

text-driven image editing, steganography, realistic image synthesis, and text-based image manipulation.

Generative AI models have demonstrated their capabilities in various tasks related to molecule generation and design. Text-based approaches like MolT5 [58] and Text2Mol [60] allow for the de novo generation of molecules and the retrieval of molecular structures using textual descriptions. These techniques offer several advantages, including accelerated molecule design, molecule captioning, and efficient exploration of chemical space. Furthermore, methods such as Mol-CycleGAN [125], JT-VAE [57], and MoFlow [76] enable the design of molecules with desired properties by leveraging chemical properties and molecular graphs. These advancements have promising applications in the fields of drug discovery.

Generative AI models have made significant strides in the field of 3D content generation based on textual input. These models demonstrate the ability to generate various forms of 3D content guided by text descriptions. Magic3D [103] allows for the detailed synthesis of 3D images, leveraging textual descriptions as input. AvatarCLIP [66] focuses on text-driven 3D avatars with animations, enabling dynamic and interactive virtual characters. DreamFace [102] takes text guidance to create personalized and animatable 3D faces, offering a high degree of customization. AvatarCraft [55] goes a step further by generating 3D human avatars with specific identities and artistic styles based on text prompts. MotionCLIP [118] explores the generation of 3D motion using text descriptions, allowing for the creation of animated sequences. Progressive structure-conditional GANs (PSGAN) [144] specialize in generating animated characters from text, facilitating the development of diverse and engaging virtual personas. DREAMFUSION [133] stands out in text-guided 3D object generation, providing the ability to generate 3D objects based on textual descriptions. DreamAvatar [165] combines text and shape information to generate 3D avatars with customizable features. These advancements not only enable applications such as 3D image inpainting using the point encoder GAN [161], but also offer the generation of realistic human poses using GAN-Poser [51], opening up new possibilities in virtual environments.

Generative AI models, including table-GAN [52], TGAN [94], and CTGAN [83], provide the ability to create synthetic tabular datasets that closely resemble the original data. These models offer valuable resources for tasks such as data augmentation, algorithm testing, and privacy preservation. By leveraging these generative models, researchers and practitioners can generate datasets with similar statistical properties as the original data, which can be used for training machine learning models, conducting simulations, and performing various data-driven analyses. This empowers the exploration and development of novel solutions in fields that rely on tabular data, fostering advancements in machine learning, data science, and decision-making processes.

In the broader context of generative AI techniques, our discussion encompasses various aspects, including the transformation of unstructured text into structured knowledge graphs through approaches like Grapher [117] for text-to-knowledge-graph conversion. Additionally, we explore the conversion of knowledge graphs, represented as RDF triples, into coherent text using techniques such as GTR-LSTM [129] for knowledge-graph-to-text conversion.

4.4. Evaluation Metrics

In evaluating the performance of models across various tasks, a combination of qualitative and quantitative metrics is utilized. Quantitative metrics provide objective and numerical measures to assess different aspects of model performance. In image-based tasks like text-based image synthesis or generation, metrics such as inception score (IS), Frechet inception distance (FID), PSNR, and SSIM quantify the quality, fidelity, and similarity of generated images. Text-based tasks, such as language translation, rely on metrics like BLEU, accuracy, and F1 score to measure the accuracy, fluency, and alignment of translations. Speech-related tasks, like text-to-speech synthesis, utilize metrics such as MOS, PESQ, and RTF to evaluate the quality, intelligibility, and real-time performance of synthesized

speech. For code generation and understanding tasks, metrics like BLEU, CodeBLEW, pass@k, and code evaluation benchmarks quantitatively assess the correctness, similarity, and performance of generated code. Additionally, qualitative metrics play a crucial role by capturing subjective assessments and human judgments. These metrics involve human evaluation, expert reviews, and user feedback to evaluate factors such as visual appeal, coherence, naturalness, readability, and user satisfaction. By considering both quantitative and qualitative metrics, a comprehensive evaluation approach is achieved, ensuring that the models perform effectively and meet both objective standards and subjective expectations across a wide range of tasks.

4.5. Challenges and Implementation Issues

There are several challenges in the implementation of AIGC. Addressing these challenges requires continuous research, collaboration, and interdisciplinary efforts. Researchers, developers, policymakers, and users need to work together to develop improved techniques, frameworks, and guidelines that advance the capabilities and responsible deployment of AIGC.

- Training data requirements: Generative AI models require large and diverse datasets to learn the underlying patterns and generate meaningful outputs. However, acquiring and curating such datasets can be challenging. It may involve manually collecting or generating a vast amount of data that accurately represent the target domain. The quality of the data is crucial, as the model's performance heavily relies on the richness and diversity of the training data [32].
- Computational resources: Training and deploying generative AI models can be computationally intensive. Large-scale models with millions or billions of parameters and complex tasks may require significant computational power, specialized hardware like GPUs or TPUs, and ample storage resources. The high computational requirements can limit the accessibility and affordability of generative AI for individuals or organizations with limited resources. Developing more efficient model architectures and optimization techniques, as well as leveraging cloud computing resources, can help address this challenge [30].
- Mode collapse: Mode collapse occurs when a generative model fails to capture the full diversity of the training data and instead generates repetitive or limited variations. For example, an image generation model may consistently produce images of a specific object, ignoring other possible objects in the training data [32]. Overcoming mode collapse is a significant challenge in generative AI research. Techniques such as improving model architectures, optimizing loss functions, or using ensemble methods are explored to encourage the model to generate a broader range of outputs.
- Interpretability and transparency: Many generative AI models, particularly deep neural networks, are often considered black boxes, meaning their decision-making processes are not easily interpretable by humans. This lack of interpretability and transparency can hinder trust, especially in critical domains where explanations and justifications are required [69]. Researchers are actively exploring techniques to enhance the interpretability of generative models, such as visualization methods, attention mechanisms, or generating explanations alongside the outputs, to provide insights into the model's inner workings.
- Evaluation and feedback: Evaluating the quality and creativity of generative outputs is a complex task. Traditional evaluation metrics may not fully capture the desired characteristics of generated content, such as novelty, coherence, or semantic relevance. Developing reliable evaluation metrics specific to generative AI is an ongoing research area. Additionally, obtaining meaningful feedback from users or experts is crucial to improve the models iteratively [69]. Collecting feedback at scale and effectively incorporating it into the training process is challenging but necessary for model refinement.

- Generalization and adaptation: Generative models may struggle to generalize well to unseen or domain-shifted data. They may be sensitive to changes in input distribution or fail to capture the underlying patterns in new contexts. Adapting generative models to new domains or ensuring their reliable performance across different datasets and scenarios is an ongoing challenge. Techniques such as transfer learning, domain adaptation, or fine-tuning on specific target data are explored to improve generalization and adaptation capabilities.
- Ethical considerations: Generative AI technologies raise ethical concerns, particularly when they can be misused for malicious purposes. For instance, deepfake technology can create highly realistic but fabricated content, leading to potential misinformation or harm. Ensuring responsible and ethical use of generative models requires establishing guidelines, regulations, and safeguards. This includes implementing techniques for detecting and mitigating the misuse of generative AI, promoting transparency and accountability, and addressing potential biases in the generated outputs [23,173].

5. Conclusions and Future Work

In this study, we have explored the requirements, models, input–output formats, and evaluation metrics relevant to generative AI systems. By addressing the research questions, we have provided valuable insights and a comprehensive understanding of these aspects. We identified three distinct categories of requirements—hardware, software, and user experience—for implementing generative AI systems. This knowledge is crucial for researchers, developers, and practitioners in effectively planning and designing such systems. Additionally, we presented a taxonomy of generative AI models based on their architecture, including popular models such as VAEs, GANs, diffusion models, transformers, language models, normalizing flow models, and hybrid models. This taxonomy serves as a guide for selecting appropriate models based on specific application requirements, fostering advancements in the field. Furthermore, we classified the input and output formats used in generative AI systems, providing a comprehensive and organized overview. This classification, along with the associated tasks and models, offers researchers and practitioners a valuable resource for developing customized generative AI systems for various applications. Moreover, we proposed a classification system based on output types and discussed commonly used evaluation metrics. This contribution enables the establishment of robust evaluation frameworks for generative AI models, enhancing their credibility and facilitating comparative analyses.

Moving forward, there are several avenues for future research in the field of generative AI. Firstly, it would be valuable to investigate further the interplay between the identified requirements and the performance of generative AI systems. Understanding how different hardware, software, and user experience factors impact system outcomes can lead to more optimized and efficient implementations. Additionally, as the field continues to evolve, it is important to continually update and expand the taxonomy of generative AI models. New model architectures and variations emerge regularly, and incorporating them into the taxonomy would provide a comprehensive and up-to-date overview. Furthermore, exploring additional input and output formats for generative AI tasks can contribute to a more comprehensive understanding of system capabilities and limitations. Investigating novel formats and their suitability for specific tasks can open up new possibilities for generative AI applications. Lastly, expanding the evaluation metrics and developing standardized benchmarks can advance the field by providing more rigorous and consistent means of assessing the quality and performance of generative AI models. This would facilitate fair comparisons between different approaches and promote further advancements in the field.

Funding: This research received no external funding.

Data Availability Statement: See Appendixes A and B.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. Number of papers published per year.

Year Published	Number of Papers
2014	2
2015	1
2016	7
2017	19
2018	22
2019	13
2020	12
2021	12
2022	26
2023	8
Total	122

Table A2. Numerical overview of different paper types.

Paper Type	Number of Papers
Conferences	60
Journals	14
Archives	48
Total	122

Table A3. Number of papers published at various venues.

Publication Venue	Number of Papers	Reference Code
arXiv	48	[33,34,43,49,50,55,58,63,69,78,80,82,84, 89,94,102,104–106,111–117,119–121,123, 124,126,128,133,139,142,143,145,147,150, 153,157,159,160,162,164,165,167]
IEEE/CVF Conference on Computer Vision and Pattern Recognition	15	[4,59,61,64,65,81,90,92,99,103,122,127, 131,136,151]
International Conference on Machine Learning	10	[45,57,70,86,88,110,130,132,149,152]
Advances in Neural Information Processing Systems	8	[42,67,77,79,83,101,109,148]
IEEE/CVF International Conference on Computer Vision	7	[56,62,87,93,98,138,163]
European Conference on Computer Vision	4	[118,140,141,144]
International Conference on Multimedia	2	[53,134]
Conference on Empirical Methods in Natural Language Processing	1	[60]
International Conference on Learning Representations	1	[95]
International Conference on Neural Information Processing Systems	1	[108]
International Conference on Image Processing	1	[135]
Winter Conference on Applications of Computer Vision	1	[54]
ACM Transactions on Graphics	1	[66]

Table A3. *Cont.*

Publication Venue	Number of Papers	Reference Code
International Conference on Acoustics, Speech and Signal Processing	1	[47]
IEEE Access	1	[36]
Mathematical Problems in Engineering	1	[156]
Neural Computing and Applications	1	[51]
Knowledge-Based Systems	1	[154]
Annual Meeting of the Association for Computational Linguistics	1	[129]
Sensors	1	[85]
International Conference on Knowledge Discovery & Data Mining	1	[76]
Journal of Cheminformatics	1	[125]
Medical Image Computing and Computer Assisted Intervention	1	[137]
Information	1	[158]
International Joint Conference on Neural Networks	1	[48]
Applied Cryptography and Network Security	1	[155]
Neurocomputing	1	[161]
American Association for the Advancement of Science	1	[35]
Advances in Multimedia Information Processing	1	[96]
Neuroinformatics	1	[100]
International Conference on Machine Vision	1	[146]
International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision	1	[97]
International Conference on Very Large Data Bases	1	[52]
The Journal of Machine Learning Research	1	[39]
IEEE Transactions on Image Processing	1	[91]

Appendix B

Table A4. Abbreviations of Evaluation Metrics Used in Generative AI Techniques.

Metric	Abbreviation
IS	Inception Score
FID	Fréchet Inception Distance
MS-SSIM	Multi-Scale Structural Similarity Index Measure
PSNR	Peak Signal-to-Noise Ratio
SSIM	Structural Similarity Index
LPIPS	Learned Perceptual Image Patch Similarity
CS-FID	Class-Conditional Fréchet Inception Distance
MAE	Mean Absolute Error
Sdir	Directional CLIP Similarity
MOS	Mean Opinion Score
NRMSE	Normalized Root Mean Square Error

Table A4. *Cont.*

Metric	Abbreviation
BLEU	Bilingual Evaluation Understudy
METOR	Metric for Evaluation of Translation with Explicit Ordering
CIDEr	Consensus-based Image Description Evaluation
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
RMSRE	Root Mean Squared Relative Error
BLEURT	Bilingual Evaluation Understudy for Natural Language Understanding in Translation
EM	Exact Match
MTPB	Multi-Turn Programming Benchmark
MDR	Matching Distance Ratio
MPJPE	Mean per Joint Positioning Error
KID	Kernel Inception Distance
RTF	Real-Time Factor
STOI	Short-Time Objective Intelligibility
PESQ	Perceptual Evaluation of Speech Quality
N.U.V	Novel, Unique, and Valid Molecules
FVD	Fréchet Video Distance
FCD	Fréchet ChemNet Distance
MRR	Mean Reciprocal Rank
QED	Quantitative Estimate of Drug-Likeness
DCR	Distance to the Closest Record
MRE	Mean Relative Error

References

1. Cao, Y.; Li, S.; Liu, Y.; Yan, Z.; Dai, Y.; Yu, P.S.; Sun, L. A comprehensive survey of ai-generated content (aigc): A history of generative ai from gan to chatgpt. *arXiv* **2023**, arXiv:2303.04226.
2. Zhang, C.; Zhang, C.; Zheng, S.; Qiao, Y.; Li, C.; Zhang, M.; Dam, S.; Myaet Thwal, C.; Tun, Y.L.; Huy, L.; et al. A Complete Survey on Generative AI (AIGC): Is ChatGPT from GPT-4 to GPT-5 All You Need? *arXiv* **2023**, arXiv:2303.11717. [CrossRef].
3. Generative AI Market Size to Hit around USD 118.06 Bn by 2032. 2023. Available online: https://www.globenewswire.com/en/news-release/2023/05/15/2668369/0/en/Generative-AI-Market-Size-to-Hit-Around-USD-118-06-Bn-By-2032.html/ (accessed on 29 June 2023).
4. Karras, T.; Laine, S.; Aila, T. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4401–4410.
5. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
6. Wang, K.; Gou, C.; Duan, Y.; Lin, Y.; Zheng, X.; Wang, F.Y. Generative adversarial networks: Introduction and outlook. *IEEE/CAA J. Autom. Sin.* **2017**, *4*, 588–598. [CrossRef]
7. Pan, Z.; Yu, W.; Yi, X.; Khan, A.; Yuan, F.; Zheng, Y. Recent progress on generative adversarial networks (GANs): A survey. *IEEE Access* **2019**, *7*, 36322–36333. [CrossRef]
8. Cao, Y.J.; Jia, L.L.; Chen, Y.X.; Lin, N.; Yang, C.; Zhang, B.; Liu, Z.; Li, X.X.; Dai, H.H. Recent Advances of Generative Adversarial Networks in Computer Vision. *IEEE Access* **2019**, *7*, 14985–15006. [CrossRef]
9. Cheng, J.; Yang, Y.; Tang, X.; Xiong, N.; Zhang, Y.; Lei, F. Generative Adversarial Networks: A Literature Review. *KSII Trans. Internet Inf. Syst.* **2020**, *14*, 4625–4647.
10. Dutta, I.K.; Ghosh, B.; Carlson, A.; Totaro, M.; Bayoumi, M. Generative adversarial networks in security: A survey. In Proceedings of the 2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York, NY, USA, 28–31 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 0399–0405.

11. Harshvardhan, G.; Gourisaria, M.K.; Pandey, M.; Rautaray, S.S. A comprehensive survey and analysis of generative models in machine learning. *Comput. Sci. Rev.* **2020**, *38*, 100285.
12. Miao, Y.; Koenig, R.; Knecht, K. The Development of Optimization Methods in Generative Urban Design: A Review. In Proceedings of the 11th Annual Symposium on Simulation for Architecture and Urban Design (SimAUD 2020), Vienna, Austria, 25–27 May 2020.
13. Jin, L.; Tan, F.; Jiang, S.; Köker, R. Generative Adversarial Network Technologies and Applications in Computer Vision. *Intell. Neurosci.* **2020**, *2020*, 1459107. [CrossRef]
14. Aggarwal, A.; Mittal, M.; Battineni, G. Generative adversarial network: An overview of theory and applications. *Int. J. Inf. Manag. Data Insights* **2021**, *1*, 100004. [CrossRef]
15. Eckerli, F.; Osterrieder, J. Generative adversarial networks in finance: An overview. *arXiv* **2021**, arXiv:2106.06364.
16. Jabbar, A.; Li, X.; Omar, B. A survey on generative adversarial networks: Variants, applications, and training. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 157. [CrossRef]
17. Jose, L.; Liu, S.; Russo, C.; Nadort, A.; Ieva, A.D. Generative Adversarial Networks in Digital Pathology and Histopathological Image Processing: A Review. Available online: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8609288/ (accessed on 25 July 2023).
18. de Rosa, G.H.; Papa, J.P. A survey on text generation using generative adversarial networks. *Pattern Recognit.* **2021**, *119*, 108098. [CrossRef]
19. Tong, X.; Liu, X.; Tan, X.; Li, X.; Jiang, J.; Xiong, Z.; Xu, T.; Jiang, H.; Qiao, N.; Zheng, M. Generative models for De Novo drug design. *J. Med. Chem.* **2021**, *64*, 14011–14027. [CrossRef] [PubMed]
20. Aldausari, N.; Sowmya, A.; Marcus, N.; Mohammadi, G. Video generative adversarial networks: A review. *ACM Comput. Surv. (CSUR)* **2022**, *55*, 30. [CrossRef]
21. Zeng, X.; Wang, F.; Luo, Y.; Kang, S.-g.; Tang, J.; Lightstone, F.C.; Fang, E.F.; Cornell, W.; Nussinov, R.; Feixiong, C. Deep generative molecular design reshapes drug discovery. *Cell Rep. Med.* **2022**, *3*, 100794. [CrossRef]
22. Li, C.; Zhang, C.; Waghwase, A.; Lee, L.H.; Rameau, F.; Yang, Y.; Bae, S.H.; Hong, C.S. Generative AI meets 3D: A Survey on Text-to-3D in AIGC Era. *arXiv* **2023**, arXiv:2305.06131.
23. Dwivedi, Y.K.; Kshetri, N.; Hughes, L.; Slade, E.L.; Jeyaraj, A.; Kar, A.K.; Baabdullah, A.M.; Koohang, A.; Raghavan, V.; Ahuja, M.; et al. "So what if ChatGPT wrote it?" Multidisciplinary perspectives on opportunities, challenges and implications of generative conversational AI for research, practice and policy. *Int. J. Inf. Manag.* **2023**, *71*, 102642. [CrossRef]
24. Danel, T.; Łęski, J.; Podlewska, S.; Podolak, I.T. Docking-based generative approaches in the search for new drug candidates. *Drug Discov. Today* **2023**, *28*, 103439. [CrossRef]
25. Gozalo-Brizuela, R.; Garrido-Merchán, E.C. A survey of Generative AI Applications. *arXiv* **2023**, arXiv:2306.02781.
26. Gozalo-Brizuela, R.; Garrido-Merchan, E.C. ChatGPT is not all you need. A State of the Art Review of large Generative AI models. *arXiv* **2023**, arXiv:2301.04655.
27. Liu, Y.; Yang, Z.; Yu, Z.; Liu, Z.; Liu, D.; Lin, H.; Li, M.; Ma, S.; Avdeev, M.; Shi, S. Generative artificial intelligence and its applications in materials science: Current situation and future perspectives. *J. Mater.* **2023**, *9*, 798–816. [CrossRef]
28. Roumeliotis, K.I.; Tselikas, N.D. ChatGPT and Open-AI Models: A Preliminary Review. *Future Internet* **2023**, *15*, 192. [CrossRef]
29. Zhang, M.; Qamar, M.; Kang, T.; Jung, Y.; Zhang, C.; Bae, S.H.; Zhang, C. A survey on graph diffusion models: Generative ai in science for molecule, protein and material. *arXiv* **2023**, arXiv:2304.01565.
30. Zhang, C.; Zhang, C.; Li, C.; Qiao, Y.; Zheng, S.; Dam, S.K.; Zhang, M.; Kim, J.U.; Kim, S.T.; Choi, J.; et al. One small step for generative ai, one giant leap for agi: A complete survey on chatgpt in aigc era. *arXiv* **2023**, arXiv:2304.06488.
31. Zhang, C.; Zhang, C.; Zheng, S.; Zhang, M.; Qamar, M.; Bae, S.H.; Kweon, I.S. A Survey on Audio Diffusion Models: Text To Speech Synthesis and Enhancement in Generative AI. *arXiv* **2023**, arXiv:2303.13336.
32. Zhang, C.; Zhang, C.; Zhang, M.; Kweon, I.S. Text-to-image Diffusion Models in Generative AI: A Survey. *arXiv* **2023**, arXiv:2303.07909.
33. Thoppilan, R.; De Freitas, D.; Hall, J.; Shazeer, N.; Kulshreshtha, A.; Cheng, H.T.; Jin, A.; Bos, T.; Baker, L.; Du, Y.; et al. Lamda: Language models for dialog applications. *arXiv* **2022**, arXiv:2201.08239.
34. Schick, T.; Dwivedi-Yu, J.; Jiang, Z.; Petroni, F.; Lewis, P.; Izacard, G.; You, Q.; Nalmpantis, C.; Grave, E.; Riedel, S. PEER: A Collaborative Language Model. *arXiv* **2022**, arXiv:2208.11663.
35. Li, Y.; Choi, D.; Chung, J.; Kushman, N.; Schrittwieser, J.; Leblond, R.; Eccles, T.; Keeling, J.; Gimeno, F.; Dal Lago, A.; et al. Competition-level code generation with alphacode. *Science* **2022**, *378*, 1092–1097. [CrossRef]
36. Fang, W.; Ding, Y.; Zhang, F.; Sheng, J. Gesture Recognition Based on CNN and DCGAN for Calculation and Text Output. *IEEE Access* **2019**, *7*, 28230–28237. [CrossRef]
37. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Proceedings, Part V 13; Springer: Berlin/Heidelberg, Germany, 2014; pp. 740–755.
38. Young, P.; Lai, A.; Hodosh, M.; Hockenmaier, J. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Trans. Assoc. Comput. Linguist.* **2014**, *2*, 67–78. [CrossRef]
39. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 5485–5551.

40. Fonseca, E.; Favory, X.; Pons, J.; Font, F.; Serra, X. Fsd50k: An open dataset of human-labeled sound events. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *30*, 829–852. [CrossRef]
41. Kim, C.D.; Kim, B.; Lee, H.; Kim, G. Audiocaps: Generating captions for audios in the wild. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*; Association for Computational Linguistics: Minneapolis, MN, USA, 2019; pp. 119–132.
42. Ren, Y.; Ruan, Y.; Tan, X.; Qin, T.; Zhao, S.; Zhao, Z.; Liu, T.Y. Fastspeech: Fast, robust and controllable text to speech. *arXiv* **2019**, arXiv:1905.09263.
43. Brock, A.; Donahue, J.; Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. *arXiv* **2018**, arXiv:1809.11096.
44. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
45. Berabi, B.; He, J.; Raychev, V.; Vechev, M. Tfix: Learning to fix coding errors with a text-to-text transformer. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 18–24 July 2021; pp. 780–791.
46. Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; Chen, X. Improved techniques for training gans. *arXiv* **2016**, arXiv:1606.03498.
47. Elizalde, B.; Deshmukh, S.; Al Ismail, M.; Wang, H. Clap learning audio concepts from natural language supervision. In Proceedings of the ICASSP 2023–2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Rhodes Island, Greece, 4–10 June 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 1–5.
48. Manco, I.; Benetos, E.; Quinton, E.; Fazekas, G. MusCaps: Generating Captions for Music Audio. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 18–22 July 2021; pp. 1–8. [CrossRef]
49. Chandramouli, P.; Gandikota, K.V. LDEdit: Towards Generalized Text Guided Image Manipulation via Latent Diffusion Models. *arXiv* **2022**, arXiv:2210.02249.
50. Nijkamp, E.; Pang, B.; Hayashi, H.; Tu, L.; Wang, H.; Zhou, Y.; Savarese, S.; Xiong, C. CodeGen: An Open Large Language Model for Code with Multi-Turn Program Synthesis. *arXiv* **2022**, arXiv:2203.13474.
51. Jain, D.K.; Zareapoor, M.; Jain, R.; Kathuria, A.; Bachhety, S. GAN-Poser: An improvised bidirectional GAN model for human motion prediction. *Neural Comput. Appl.* **2020**, *32*, 14579–14591. [CrossRef]
52. Park, N.; Mohammadi, M.; Gorde, K.; Jajodia, S.; Park, H.; Kim, Y. Data Synthesis Based on Generative Adversarial Networks. *Proc. VLDB Endow.* **2018**, *11*, 1071–1083. [CrossRef]
53. Huang, R.; Zhao, Z.; Liu, H.; Liu, J.; Cui, C.; Ren, Y. Prodiff: Progressive fast diffusion model for high-quality text-to-speech. In Proceedings of the 30th ACM International Conference on Multimedia, Lisboa, Portugal, 10–14 October 2022; pp. 2595–2605.
54. Chang, B.; Zhang, Q.; Pan, S.; Meng, L. Generating handwritten chinese characters using cyclegan. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 199–207.
55. Jiang, R.; Wang, C.; Zhang, J.; Chai, M.; He, M.; Chen, D.; Liao, J. AvatarCraft: Transforming Text into Neural Human Avatars with Parameterized Shape and Pose Control. *arXiv* **2023**, arXiv:2303.17606.
56. Kupyn, O.; Martyniuk, T.; Wu, J.; Wang, Z. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 8878–8887.
57. Jin, W.; Barzilay, R.; Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 2323–2332.
58. Edwards, C.; Lai, T.; Ros, K.; Honke, G.; Ji, H. Translation between molecules and natural language. *arXiv* **2022**, arXiv:2204.11817.
59. Vinyals, O.; Toshev, A.; Bengio, S.; Erhan, D. Show and tell: A neural image caption generator. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3156–3164.
60. Edwards, C.; Zhai, C.; Ji, H. Text2mol: Cross-modal molecule retrieval with natural language queries. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, online, 18–24 July 2021; pp. 595–607.
61. Kupyn, O.; Budzan, V.; Mykhailych, M.; Mishkin, D.; Matas, J. Deblurgan: Blind motion deblurring using conditional adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8183–8192.
62. Liang, X.; Hu, Z.; Zhang, H.; Gan, C.; Xing, E.P. Recurrent topic-transition gan for visual paragraph generation. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3362–3371.
63. Jeong, M.; Kim, H.; Cheon, S.J.; Choi, B.J.; Kim, N.S. Diff-tts: A denoising diffusion model for text-to-speech. *arXiv* **2021**, arXiv:2104.01409.
64. Chen, B.C.; Kae, A. Toward realistic image compositing with adversarial learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 8415–8424.
65. Chen, J.; Guo, H.; Yi, K.; Li, B.; Elhoseiny, M. Visualgpt: Data-efficient adaptation of pretrained language models for image captioning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 18030–18040.
66. Hong, F.; Zhang, M.; Pan, L.; Cai, Z.; Yang, L.; Liu, Z. AvatarCLIP: Zero-Shot Text-Driven Generation and Animation of 3D Avatars. *ACM Trans. Graph. (TOG)* **2022**, *41*, 1–19. [CrossRef]

67. Nam, S.; Kim, Y.; Kim, S.J. Text-Adaptive Generative Adversarial Networks: Manipulating Images with Natural Language. Available online: https://dl.acm.org/doi/pdf/10.5555/3326943.3326948 (accessed on 25 July 2023).
68. Zhang, H.; Li, Y.; Ma, F.; Gao, J.; Su, L. Texttruth: An unsupervised approach to discover trustworthy information from multi-sourced text data. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 2729–2737.
69. Evans, O.; Cotton-Barratt, O.; Finnveden, L.; Bales, A.; Balwit, A.; Wills, P.; Righetti, L.; Saunders, W. Truthful AI: Developing and governing AI that does not lie. *arXiv* **2021**, arXiv:2110.06674.
70. Liang, P.P.; Wu, C.; Morency, L.P.; Salakhutdinov, R. Towards understanding and mitigating social biases in language models. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 18–24 July 2021; pp. 6565–6576.
71. Marchandot, B.; Matsushita, K.; Carmona, A.; Trimaille, A.; Morel, O. ChatGPT: The next frontier in academic writing for cardiologists or a pandora's box of ethical dilemmas. *Eur. Heart J. Open* **2023**, *3*, oead007. [CrossRef]
72. Wu, Y.; Yu, N.; Li, Z.; Backes, M.; Zhang, Y. Membership Inference Attacks Against Text-to-image Generation Models. *arXiv* **2022**, arXiv:2210.00968.
73. Carlini, N.; Tramer, F.; Wallace, E.; Jagielski, M.; Herbert-Voss, A.; Lee, K.; Roberts, A.; Brown, T.B.; Song, D.; Erlingsson, U.; et al. Extracting Training Data from Large Language Models. In Proceedings of the USENIX Security Symposium, Virtual, 11–13 August 2021; Volume 6.
74. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. Available online: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf (accessed on 25 July 2023).
75. Jiang, Z.; Xu, F.F.; Araki, J.; Neubig, G. How can we know what language models know? *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 423–438. [CrossRef]
76. Zang, C.; Wang, F. MoFlow: An invertible flow model for generating molecular graphs. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Virtual Event, 6–10 July 2020; pp. 617–626.
77. Ding, M.; Yang, Z.; Hong, W.; Zheng, W.; Zhou, C.; Yin, D.; Lin, J.; Zou, X.; Shao, Z.; Yang, H.; et al. Cogview: Mastering text-to-image generation via transformers. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 19822–19835.
78. Mansimov, E.; Parisotto, E.; Ba, J.L.; Salakhutdinov, R. Generating images from captions with attention. *arXiv* **2015**, arXiv:1511.02793.
79. Liu, Q.; Allamanis, M.; Brockschmidt, M.; Gaunt, A. Constrained Graph Variational Autoencoders for Molecule Design. *Adv. Neural Inf. Process. Syst.* Available online: https://proceedings.neurips.cc/paper_files/paper/2018/file/b8a03c5c15fcfa8dae0b03351eb1742f-Paper.pdf (accessed on 25 July 2023).
80. Xie, T.; Fu, X.; Ganea, O.E.; Barzilay, R.; Jaakkola, T. Crystal diffusion variational autoencoder for periodic material generation. *arXiv* **2021**, arXiv:2110.06197.
81. Isola, P.; Zhu, J.Y.; Zhou, T.; Efros, A.A. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, Hawaii, USA, 21–26 July 2017; pp. 1125–1134.
82. Michelsanti, D.; Tan, Z.H. Conditional generative adversarial networks for speech enhancement and noise-robust speaker verification. *arXiv* **2017**, arXiv:1709.01703.
83. Xu, L.; Skoularidou, M.; Cuesta-Infante, A.; Veeramachaneni, K. Modeling Tabular Data Using Conditional Gan. *Adv. Neural Inf. Process. Syst.* Available online: https://proceedings.neurips.cc/paper_files/paper/2019/file/254ed7d2de3b23ab10936522dd547b78-Paper.pdf (accessed on 25 July 2023).
84. Dash, A.; Gamboa, J.C.B.; Ahmed, S.; Liwicki, M.; Afzal, M.Z. Tac-gan-text conditioned auxiliary classifier generative adversarial network. *arXiv* **2017**, arXiv:1703.06412.
85. Nam, S.; Jeon, S.; Kim, H.; Moon, J. Recurrent gans password cracker for iot password security enhancement. *Sensors* **2020**, *20*, 3106. [CrossRef] [PubMed]
86. Reed, S.; Akata, Z.; Yan, X.; Logeswaran, L.; Schiele, B.; Lee, H. Generative adversarial text to image synthesis. In Proceedings of the International Conference on Machine Learning, PMLR, New York, NY, USA, 20–22 June 2016; pp. 1060–1069.
87. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2223–2232.
88. Kim, T.; Cha, M.; Kim, H.; Lee, J.K.; Kim, J. Learning to discover cross-domain relations with generative adversarial networks. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6 August–11 August 2017; pp. 1857–1865.
89. Taigman, Y.; Polyak, A.; Wolf, L. Unsupervised cross-domain image generation. *arXiv* **2016**, arXiv:1611.02200.
90. Ledig, C.; Theis, L.; Huszár, F.; Caballero, J.; Cunningham, A.; Acosta, A.; Aitken, A.; Tejani, A.; Totz, J.; Wang, Z.; et al. Photo-realistic single image super-resolution using a generative adversarial network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4681–4690.
91. Lucas, A.; Lopez-Tapia, S.; Molina, R.; Katsaggelos, A.K. Generative adversarial networks and perceptual losses for video super-resolution. *IEEE Trans. Image Process.* **2019**, *28*, 3312–3327. [CrossRef]

92. Xu, T.; Zhang, P.; Huang, Q.; Zhang, H.; Gan, Z.; Huang, X.; He, X. AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake, UT, USA, 18–23 June 2018; IEEE Computer Society: Los Alamitos, CA, USA, 2018; pp. 1316–1324. [CrossRef]
93. Zhang, H.; Xu, T.; Li, H.; Zhang, S.; Wang, X.; Huang, X.; Metaxas, D.N. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5907–5915.
94. Xu, L.; Veeramachaneni, K. Synthesizing tabular data using generative adversarial networks. *arXiv* **2018**, arXiv:1811.11264.
95. Samangouei, P.; Kabkab, M.; Chellappa, R. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
96. Shi, H.; Dong, J.; Wang, W.; Qian, Y.; Zhang, X. SSGAN: Secure steganography based on generative adversarial networks. In Proceedings of the Advances in Multimedia Information Processing–PCM 2017: 18th Pacific-Rim Conference on Multimedia, Harbin, China, 28–29 September 2017; Revised Selected Papers, Part I 18; Springer: Berlin/Heidelberg, Germany, 2018; pp. 534–544.
97. Hartmann, S.; Weinmann, M.; Wessel, R.; Klein, R. Streetgan: Towards Road Network Synthesis with Generative Adversarial Networks. Available online: https://otik.uk.zcu.cz/bitstream/11025/29554/1/Hartmann.pdf (accessed on 25 July 2023).
98. Patashnik, O.; Wu, Z.; Shechtman, E.; Cohen-Or, D.; Lischinski, D. Styleclip: Text-driven manipulation of stylegan imagery. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 2085–2094.
99. Dolhansky, B.; Ferrer, C.C. Eye in-painting with exemplar generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7902–7911.
100. Xue, Y.; Xu, T.; Zhang, H.; Long, L.R.; Huang, X. Segan: Adversarial network with multi-scale l 1 loss for medical image segmentation. *Neuroinformatics* **2018**, *16*, 383–392. [CrossRef]
101. Ho, J.; Jain, A.; Abbeel, P. *Denoising Diffusion Probabilistic Models*; Curran Associates Inc.: Red Hook, NY, USA, 2020; NIPS'20.
102. Zhang, L.; Qiu, Q.; Lin, H.; Zhang, Q.; Shi, C.; Yang, W.; Shi, Y.; Yang, S.; Xu, L.; Yu, J. DreamFace: Progressive Generation of Animatable 3D Faces under Text Guidance. *arXiv* **2023**, arXiv:2304.03117.
103. Lin, C.H.; Gao, J.; Tang, L.; Takikawa, T.; Zeng, X.; Huang, X.; Kreis, K.; Fidler, S.; Liu, M.Y.; Lin, T.Y. Magic3d: High-resolution text-to-3d content creation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, Canada, 18–22 June 2023; pp. 300–309.
104. Wu, J.Z.; Ge, Y.; Wang, X.; Lei, W.; Gu, Y.; Hsu, W.; Shan, Y.; Qie, X.; Shou, M.Z. Tune-A-Video: One-Shot Tuning of Image Diffusion Models for Text-to-Video Generation. *arXiv* **2022**, arXiv:2212.11565.
105. Morehead, A.; Cheng, J. Geometry-complete diffusion for 3d molecule generation. *arXiv* **2023**, arXiv:2302.04313.
106. Molad, E.; Horwitz, E.; Valevski, D.; Acha, A.R.; Matias, Y.; Pritch, Y.; Leviathan, Y.; Hoshen, Y. Dreamix: Video diffusion models are general video editors. *arXiv* **2023**, arXiv:2302.01329.
107. Zhao, W.X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. A survey of large language models. *arXiv* **2023**, arXiv:2303.18223.
108. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems–Volume 2*; MIT Press: Cambridge, MA, USA, 2014; NIPS'14, pp. 3104–3112.
109. Alayrac, J.B.; Donahue, J.; Luc, P.; Miech, A.; Barr, I.; Hasson, Y.; Lenc, K.; Mensch, A.; Millican, K.; Reynolds, M.; et al. Flamingo: A visual language model for few-shot learning. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 23716–23736.
110. Kanade, A.; Maniatis, P.; Balakrishnan, G.; Shi, K. Learning and evaluating contextual embedding of source code. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 13–18 July 2020; pp. 5110–5121.
111. Bhavya, B.; Xiong, J.; Zhai, C. Analogy Generation by Prompting Large Language Models: A Case Study of InstructGPT. *arXiv* **2022**, arXiv:2210.04186.
112. Kale, M.; Rastogi, A. Text-to-text pre-training for data-to-text tasks. *arXiv* **2020**, arXiv:2005.10433.
113. Chen, M.; Tan, X.; Li, B.; Liu, Y.; Qin, T.; Zhao, S.; Liu, T.Y. Adaspeech: Adaptive text to speech for custom voice. *arXiv* **2021**, arXiv:2103.00993.
114. Feng, Z.; Guo, D.; Tang, D.; Duan, N.; Feng, X.; Gong, M.; Shou, L.; Qin, B.; Liu, T.; Jiang, D.; et al. Codebert: A pre-trained model for programming and natural languages. *arXiv* **2020**, arXiv:2002.08155.
115. Wang, Y.; Wang, W.; Joty, S.; Hoi, S.C. Codet5: Identifier-aware unified pre-trained encoder–decoder models for code understanding and generation. *arXiv* **2021**, arXiv:2109.00859.
116. Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H.P.d.O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. Evaluating large language models trained on code. *arXiv* **2021**, arXiv:2107.03374.
117. Melnyk, I.; Dognin, P.; Das, P. Knowledge Graph Generation From Text. *arXiv* **2022**, arXiv:2211.10511.
118. Tevet, G.; Gordon, B.; Hertz, A.; Bermano, A.H.; Cohen-Or, D. Motionclip: Exposing human motion generation to clip space. In Proceedings of the Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, 23–27 October 2022; Proceedings, Part XXII; Springer: Berlin/Heidelberg, Germany, 2022; pp. 358–374.
119. Villegas, R.; Babaeizadeh, M.; Kindermans, P.J.; Moraldo, H.; Zhang, H.; Saffar, M.T.; Castro, S.; Kunze, J.; Erhan, D. Phenaki: Variable length video generation from open domain textual description. *arXiv* **2022**, arXiv:2210.02399.

120. Borsos, Z.; Marinier, R.; Vincent, D.; Kharitonov, E.; Pietquin, O.; Sharifi, M.; Teboul, O.; Grangier, D.; Tagliasacchi, M.; Zeghidour, N. Audiolm: A language modeling approach to audio generation. *arXiv* **2022**, arXiv:2209.03143.
121. Narang, S.; Raffel, C.; Lee, K.; Roberts, A.; Fiedel, N.; Malkan, K. Wt5?! training text-to-text models to explain their predictions. *arXiv* **2020**, arXiv:2004.14546.
122. Zhang, Z.; Song, Y.; Qi, H. Age progression/regression by conditional adversarial autoencoder. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5810–5818.
123. Oord, A.v.d.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; Kavukcuoglu, K. Wavenet: A generative model for raw audio. *arXiv* **2016**, arXiv:1609.03499.
124. Brock, A.; Lim, T.; Ritchie, J.M.; Weston, N. Neural photo editing with introspective adversarial networks. *arXiv* **2016**, arXiv:1609.07093.
125. Maziarka, Ł.; Pocha, A.; Kaczmarczyk, J.; Rataj, K.; Danel, T.; Warchoł, M. Mol-CycleGAN: A generative model for molecular optimization. *J. Cheminform.* **2020**, *12*, 1–18. [CrossRef]
126. Demir, U.; Unal, G. Patch-based image inpainting with generative adversarial networks. *arXiv* **2018**, arXiv:1803.07422.
127. Yu, J.; Lin, Z.; Yang, J.; Shen, X.; Lu, X.; Huang, T.S. Generative image inpainting with contextual attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 5505–5514.
128. Liu, S.; Su, D.; Yu, D. Diffgan-tts: High-fidelity and efficient text-to-speech with denoising diffusion gans. *arXiv* **2022**, arXiv:2201.11972.
129. Distiawan, B.; Qi, J.; Zhang, R.; Wang, W. GTR-LSTM: A triple encoder for sentence generation from RDF data. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, 15–20 July 2018; pp. 1627–1637.
130. Radford, A.; Kim, J.W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. Learning transferable visual models from natural language supervision. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 18–24 July 2021; pp. 8748–8763.
131. Kim, G.; Kwon, T.; Ye, J.C. Diffusionclip: Text-guided diffusion models for robust image manipulation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–20 June 2022; pp. 2426–2435.
132. Gehring, J.; Auli, M.; Grangier, D.; Yarats, D.; Dauphin, Y.N. Convolutional sequence to sequence learning. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6 August–11 August 2017; pp. 1243–1252.
133. Poole, B.; Jain, A.; Barron, J.T.; Mildenhall, B. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv* **2022**, arXiv:2209.14988.
134. Wu, H.; Zheng, S.; Zhang, J.; Huang, K. Gp-gan: Towards realistic high-resolution image blending. In Proceedings of the 27th ACM International Conference on Multimedia, Nice, France, 21–25 October 2019; pp. 2487–2495.
135. Antipov, G.; Baccouche, M.; Dugelay, J.L. Face aging with conditional generative adversarial networks. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 2089–2093.
136. Tang, X.; Wang, Z.; Luo, W.; Gao, S. Face Aging with Identity-Preserved Conditional Generative Adversarial Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7939–7947. [CrossRef]
137. Chen, Y.; Shi, F.; Christodoulou, A.G.; Xie, Y.; Zhou, Z.; Li, D. Efficient and accurate MRI super-resolution using a generative adversarial network and 3D multi-level densely connected network. In Proceedings of the Medical Image Computing and Computer Assisted Intervention–MICCAI 2018: 21st International Conference, Granada, Spain, 16–20 September 2018; Proceedings, Part I; Springer: Berlin/Heidelberg, Germany, 2018; pp. 91–99.
138. Huang, R.; Zhang, S.; Li, T.; He, R. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2439–2448.
139. Berthelot, D.; Schumm, T.; Metz, L. Began: Boundary equilibrium generative adversarial networks. *arXiv* **2017**, arXiv:1703.10717.
140. Wang, X.; Yu, K.; Wu, S.; Gu, J.; Liu, Y.; Dong, C.; Qiao, Y.; Change Loy, C. Esrgan: Enhanced super-resolution generative adversarial networks. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, Munich, Germany, 8–14 September 2018.
141. Li, C.; Wand, M. Precomputed real-time texture synthesis with markovian generative adversarial networks. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part III 14; Springer: Berlin/Heidelberg, Germany, 2016; pp. 702–716.
142. Jetchev, N.; Bergmann, U.; Vollgraf, R. Texture synthesis with spatial generative adversarial networks. *arXiv* **2016**, arXiv:1611.08207.
143. Bergmann, U.; Jetchev, N.; Vollgraf, R. Learning texture manifolds with the periodic spatial GAN. *arXiv* **2017**, arXiv:1705.06566.
144. Hamada, K.; Tachibana, K.; Li, T.; Honda, H.; Uchida, Y. Full-Body High-Resolution Anime Generation with Progressive Structure-Conditional Generative Adversarial Networks. In Proceedings of the Computer Vision—ECCV 2018 Workshops, Munich, Germany, 8–14 September 2018; Leal-Taixé, L., Roth, S., Eds.; Springer International Publishing: Cham, Swizerland, 2019; pp. 67–74.

145. Dhariwal, P.; Jun, H.; Payne, C.; Kim, J.W.; Radford, A.; Sutskever, I. Jukebox: A generative model for music. *arXiv* **2020**, arXiv:2005.00341.
146. Volkhonskiy, D.; Nazarov, I.; Burnaev, E. Steganographic generative adversarial networks. In Proceedings of the Twelfth International Conference on Machine Vision (ICMV 2019). SPIE, Amsterdam, Netherlands, 16–18 November 2020; Volume 11433, pp. 991–1005.
147. Nichol, A.; Dhariwal, P.; Ramesh, A.; Shyam, P.; Mishkin, P.; McGrew, B.; Sutskever, I.; Chen, M. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv* **2021**, arXiv:2112.10741.
148. Saharia, C.; Chan, W.; Saxena, S.; Li, L.; Whang, J.; Denton, E.L.; Ghasemipour, K.; Gontijo Lopes, R.; Karagol Ayan, B.; Salimans, T.; et al. Photorealistic text-to-image diffusion models with deep language understanding. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 36479–36494.
149. Odena, A.; Olah, C.; Shlens, J. Conditional image synthesis with auxiliary classifier gans. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6 August–11 August 2017; pp. 2642–2651.
150. Chang, H.; Zhang, H.; Barber, J.; Maschinot, A.; Lezama, J.; Jiang, L.; Yang, M.H.; Murphy, K.; Freeman, W.T.; Rubinstein, M.; et al. Muse: Text-To-Image Generation via Masked Generative Transformers. *arXiv* **2023**, arXiv:2301.00704.
151. Tao, M.; Bao, B.K.; Tang, H.; Xu, C. GALIP: Generative Adversarial CLIPs for Text-to-Image Synthesis. *arXiv* **2023**, arXiv:2301.12959
152. Popov, V.; Vovk, I.; Gogoryan, V.; Sadekova, T.; Kudinov, M. Grad-tts: A diffusion probabilistic model for text-to-speech. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual Event, 18–24 July 2021; pp. 8599–8608.
153. Engel, J.; Agrawal, K.K.; Chen, S.; Gulrajani, I.; Donahue, C.; Roberts, A. Gansynth: Adversarial neural audio synthesis. *arXiv* **2019**, arXiv:1902.08710.
154. Hayashi, H.; Abe, K.; Uchida, S. GlyphGAN: Style-consistent font generation based on generative adversarial networks. *Knowl.-Based Syst.* **2019**, *186*, 104927. [CrossRef]
155. Hitaj, B.; Gasti, P.; Ateniese, G.; Perez-Cruz, F. Passgan: A deep learning approach for password guessing. In Proceedings of the Applied Cryptography and Network Security: 17th International Conference, ACNS 2019, Bogota, Colombia, 5–7 June 2019; Proceedings 17; Springer: Berlin/Heidelberg, Germany, 2019; pp. 217–237.
156. Zhou, X.; Pan, Z.; Hu, G.; Tang, S.; Zhao, C. Stock market prediction on high-frequency data using generative adversarial nets. *Math. Probl. Eng.* **2018**, *2018*, 4907423 [CrossRef]
157. Muthukumar, P.; Zhong, J. A stochastic time series model for predicting financial trends using nlp. *arXiv* **2021**, arXiv:2102.01290.
158. Wu, W.; Huang, F.; Kao, Y.; Chen, Z.; Wu, Q. Prediction method of multiple related time series based on generative adversarial networks. *Information* **2021**, *12*, 55. [CrossRef]
159. Singer, U.; Polyak, A.; Hayes, T.; Yin, X.; An, J.; Zhang, S.; Hu, Q.; Yang, H.; Ashual, O.; Gafni, O.; et al. Make-a-video: Text-to-video generation without text-video data. *arXiv* **2022**, arXiv:2209.14792.
160. Ho, J.; Chan, W.; Saharia, C.; Whang, J.; Gao, R.; Gritsenko, A.; Kingma, D.P.; Poole, B.; Norouzi, M.; Fleet, D.J.; et al. Imagen video: High definition video generation with diffusion models. *arXiv* **2022**, arXiv:2210.02303.
161. Yu, Y.; Huang, Z.; Li, F.; Zhang, H.; Le, X. Point Encoder GAN: A deep learning model for 3D point cloud inpainting. *Neurocomputing* **2020**, *384*, 192–199. [CrossRef]
162. Zhang, K.A.; Cuesta-Infante, A.; Xu, L.; Veeramachaneni, K. SteganoGAN: High capacity image steganography with GANs. *arXiv* **2019**, arXiv:1901.03892.
163. Dong, H.; Yu, S.; Wu, C.; Guo, Y. Semantic image synthesis via adversarial learning. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5706–5714.
164. Couairon, G.; Verbeek, J.; Schwenk, H.; Cord, M. Diffedit: Diffusion-based semantic image editing with mask guidance. *arXiv* **2022**, arXiv:2210.11427.
165. Cao, Y.; Cao, Y.P.; Han, K.; Shan, Y.; Wong, K.Y.K. Dreamavatar: Text-and-shape guided 3d human avatar generation via diffusion models. *arXiv* **2023**, arXiv:2304.00916.
166. Ramesh, A.; Pavlov, M.; Goh, G.; Gray, S.; Voss, C.; Radford, A.; Chen, M.; Sutskever, I. Zero-Shot Text-to-Image Generation. In Proceedings of the 38th International Conference on Machine Learning, PMLR, online, 18–24 July 2021; Meila, M., Zhang, T., Eds.; Proceedings of Machine Learning Research; MIT Press: Cambridge, MA, USA, 2021; Volume 139, pp. 8821–8831.
167. Li, Z.; Lu, S.; Guo, D.; Duan, N.; Jannu, S.; Jenks, G.; Majumder, D.; Green, J.; Svyatkovskiy, A.; Fu, S.; et al. CodeReviewer: Pre-Training for Automating Code Review Activities. *arXiv* **2022**, arXiv:2203.09095.
168. Liu, C.; Lu, S.; Chen, W.; Jiang, D.; Svyatkovskiy, A.; Fu, S.; Sundaresan, N.; Duan, N. Code Execution with Pre-trained Language Models. *arXiv* **2023**, arXiv:2305.05383.
169. Guo, D.; Ren, S.; Lu, S.; Feng, Z.; Tang, D.; Liu, S.; Zhou, L.; Duan, N.; Yin, J.; Jiang, D.; et al. GraphCodeBERT: Pre-training Code Representations with Data Flow. *arXiv* **2020**, arXiv:2009.08366
170. Guo, D.; Lu, S.; Duan, N.; Wang, Y.; Zhou, M.; Yin, J. UniXcoder: Unified Cross-Modal Pre-Training for Code Representation. Available online: https://aclanthology.org/2022.acl-long.499/ (accessed on 25 July 2023).
171. Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 7–9 July 2015; pp. 2048–2057.

172. Pan, Y.; Qiu, Z.; Yao, T.; Li, H.; Mei, T. To create what you tell: Generating videos from captions. In Proceedings of the 25th ACM International Conference on Multimedia, Mountain View, CA USA, 23–27 October 2017; pp. 1789–1798.
173. Samuelson, P. Legal Challenges to Generative AI, Part I. *Commun. ACM* **2023**, *66*, 20–23. [CrossRef]