



(10) **DE 10 2024 205 216 A1** 2024.12.12

(12) **Offenlegungsschrift**

(21) Aktenzeichen: **10 2024 205 216.7**

(22) Anmeldetag: **06.06.2024**

(43) Offenlegungstag: **12.12.2024**

(51) Int Cl.: **G06N 3/0475 (2023.01)**

(30) Unionspriorität:
18/332,355 **09.06.2023** **US**

(71) Anmelder:
**Robert Bosch Gesellschaft mit beschränkter
Haftung, 70469 Stuttgart, DE**

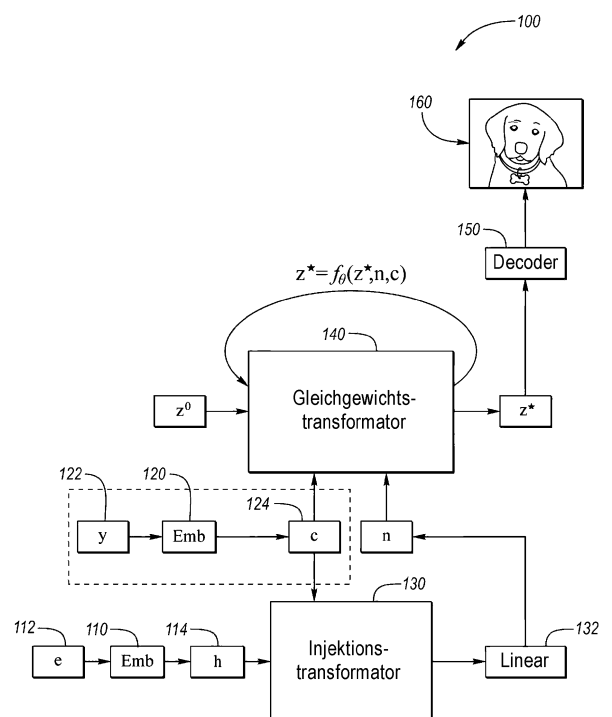
(72) Erfinder:
**Batalov, Ivan, Pittsburgh, PA, US; Azari, Bahare,
San Jose, CA, US; Geng, Zhengyang, Pittsburgh,
PA, US; Pokle, Ashwini, Pittsburgh, PA, US;
Condessa, Filipe Cabrita, Pittsburgh, PA, US;
Kolter, Jeremy Zieg, Pittsburgh, PA, US**

Prüfungsantrag gemäß § 44 PatG ist gestellt.

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen.

(54) Bezeichnung: **Einschritt-Diffusionsdestillation über tiefe Gleichgewichtsmodelle**

(57) Zusammenfassung: Es werden generative Gleichgewichtstransformatoren offenbart. Die offenbarten Ausführungsformen stellen eine einfache und effektive Technik bereit, die einen Mehrschritt-Diffusionsprozess unter ausschließlicher Verwendung von Rausch-/Bildpaaren in ein generatives Einschritt-Modell destillieren kann.



Beschreibung

Technisches Gebiet

[0001] Die vorliegende Offenbarung bezieht sich auf die Destillation von Diffusionsmodellen. In den offenbarten Ausführungsformen kann ein tiefes Gleichgewichtsmodell in einer destillierten Architektur genutzt werden. Die offenbarten Ausführungsformen können die Destillation von Diffusionsmodellen direkt aus dem anfänglichen Rauschen zu einem resultierenden Bild ermöglichen.

Hintergrund

[0002] Diffusionsmodelle haben in einem breiten Spektrum generativer Aufgaben, einschließlich der Erzeugung hochwertiger Bilder sowie der Audio- und Videosynthese, eine außergewöhnliche Leistungsfähigkeit gezeigt. Wissensdestillation bezieht sich auf den Prozess der Wissensübertragung von einem großen Modell oder einem Satz von Modellen auf ein einzelnes kleineres (und schnelleres) Modell, das unter realen Bedingungen praktisch eingesetzt werden kann. Die Wissensdestillation kann als eine Form der Modellkomprimierung betrachtet werden.

Kurzdarstellung

[0003] Ein wesentlicher Nachteil von Diffusionsmodellen ist ihr langsamer generativer Prozess, der die praktische Anwendbarkeit von Diffusionsmodellen in Echtzeit- oder ressourcenbeschränkten Szenarien einschränkt. Vorhandene Destillationsverfahren für Diffusionsmodelle zielen darauf ab, den Mehrschritt-Sampling-Prozess in einen effizienteren Prozess mit wenigen oder nur einem Schritt zu verdichten. Diese Verfahren haben jedoch den Nachteil, dass mehrere Trainingsdurchläufe erforderlich sind, um den langwierigen Sampling-Prozess zu destillieren und große Speicher- und Rechenressourcen benötigen, da eine doppelte Kopie des Modells gepflegt werden muss. Die offenbarten Ausführungsformen können einen Mehrschritt-Diffusionsprozess unter ausschließlicher Verwendung von Rausch-/Bildpaaren in ein generatives Einschritt-Modell destillieren.

[0004] In einigen Ausführungsformen umfassen die offenbarten Verfahren das Umwandeln von Rauschen in einen Rauscheinbettungsvektor; das Tokenisieren des Rauscheinbettungsvektors über einen Injektionstransformator; das Eingeben des tokenisierten Rauschens in einen Gleichgewichtstransformator; das Lösen eines Fixpunkts über den Gleichgewichtstransformator; und das Decodieren des Fixpunkts zum Erzeugen einer Bildprobe.

Kurze Beschreibung der Zeichnungen

Fig. 1 veranschaulicht ein Ausführungsbeispiel zum Erzeugen von Bildproben gemäß der vorliegenden Offenbarung.

Fig. 2 veranschaulicht ein Ausführungsbeispiel zum Erzeugen von Bildproben gemäß der vorliegenden Offenbarung.

Fig. 3 veranschaulicht ein Ausführungsbeispiel eines allgemeinen Computersystems gemäß der vorliegenden Offenbarung.

Ausführliche Beschreibung

[0005] Hierin werden Ausführungsformen der vorliegenden Offenbarung beschrieben. Es versteht sich jedoch, dass die offenbarten Ausführungsformen lediglich Beispiele sind und andere Ausführungsformen verschiedene und alternative Formen annehmen können. Die Figuren sind nicht unbedingt maßstabsgetreu; einige Merkmale könnten übertrieben oder minimiert sein, um Details bestimmter Komponenten zu zeigen. Daher sind die hierin offenbarten spezifischen strukturellen und funktionalen Details nicht als einschränkend auszulegen, sondern lediglich als repräsentative Grundlage, um einem Fachmann den vielfältigen Einsatz der vorliegenden Erfindung zu lehren. Wie der Fachmann verstehen wird, können verschiedene Merkmale, die mit Bezug auf eine der Figuren veranschaulicht und beschrieben sind, mit Merkmalen kombiniert werden, die in einer oder mehreren anderen Figuren veranschaulicht sind, um Ausführungsformen zu erzeugen, die nicht ausdrücklich veranschaulicht oder beschrieben sind. Die veranschaulichten Kombinationen von Merkmalen stellen repräsentative Ausführungsformen für typische Anwendungen bereit. Für bestimmte Anwen-

dungen oder Implementierungen könnten jedoch verschiedene Kombinationen und Modifikationen der Merkmale im Einklang mit den Lehren dieser Offenbarung erwünscht sein.

[0006] Diffusionsmodelle haben auf einer breiten Palette generativer Aufgaben nachweislich eine bemerkenswerte Leistungsfähigkeit, wie bei der Erzeugung und Bearbeitung qualitativ hochwertiger Bilder, der Audiosynthese, von Videos, 3D-Formen, Texten und Molekülen. Diese Modelle werden mit einem Rauschunterdrückungsziel trainiert, das aus Score-Matching, variationsbedingter Inferenz oder optimalem Transport abgeleitet wird, wodurch saubere Datenproben durch progressives Entrauschen des anfänglichen Gaußschen Rauschens während des Inferenzprozesses erzeugt werden können. Im Gegensatz zum adversarialen Training führt das Rauschunterdrückungsziel zu einer stabileren Trainingsprozedur, was wiederum eine effektive Skalierung von Diffusionsmodellen ermöglicht. Trotz der vielversprechenden Ergebnisse besteht ein großer Nachteil von Diffusionsmodellen in ihrem langsamen generativen Prozess, der oft Hunderte bis Tausende von Modellbewertungen erforderlich macht. Diese Rechenkomplexität begrenzt die Anwendbarkeit von Diffusionsmodellen in Echtzeit- oder ressourcenbeschränkten Szenarien.

[0007] Um den langsamen generativen Prozess von Diffusionsmodellen zu beschleunigen, haben Forscher Destillationsverfahren vorgeschlagen, die darauf abzielen, den Mehrschritt-Sampling-Prozess in einen effizienteren Prozess mit wenigen oder nur einem Schritt zu destillieren. Allerdings bringen diese Techniken oftmals ihre eigenen Herausforderungen mit sich. Für eine erfolgreiche Wissensübertragung vom größeren auf das kleinere Modell müssen die Destillationsziele sorgfältig ausgestaltet werden. Ferner sind zur Destillation eines langen Sampling-Prozesses auf einen Prozess mit wenigen Schritten oftmals mehrere Trainingsdurchläufe erforderlich. Die meisten gängigen Techniken zur Online-Destillation erfordern die Pflege doppelter Kopien des Modells, was zu erhöhten Speicher- und Rechenanforderungen führt. Daher besteht eindeutig ein Bedarf an stromlinienförmigeren und effizienteren Ansätzen, die die Rechenanforderungen zur Destillation von Diffusionsmodellen angehen, ohne die generativen Fähigkeiten zu opfern.

[0008] Eine vorgeschlagene Technik zur Wissensdestillation destilliert einen Denoising Diffusion Implicit Model („DDIM“)-Mehrschritt-Sampler in einen Einschritt-Sampler, indem das Schülermodell anhand von einer Million synthetischer Bildproben trainiert wird. Andere Techniken, wie die progressive Destillation, zielen darauf ab, ein T-Schritt-Lehrer-DDIM in ein neues T/2-Schritt-Schüler-DDIM zu destillieren und diesen Prozess zu wiederholen, bis Einschritt-Modelle erreicht werden. Die transitive Schließzeit-Destillation verallgemeinert die progressive Destillation, um $N > 2$ Schritte auf einmal zusammen zu destillieren und so die Gesamtzahl der Trainingsphasen zu reduzieren. Konsistenzmodelle erreichen eine Online-Destillation in einem einzigen Durchlauf, indem sie die Vorteile eines sorgfältig ausgestalteten Lehrer- und Destillationsverlustziels ausnutzen. Der Diffusion Fourier-Neuraloperator („DFNO“) ordnet die anfängliche Gaußsche Verteilung der Lösungstrajektorie des umgekehrten Diffusionsprozesses durch Einfügen zeitlicher Fourier-Integraloperatoren in das vortrainierte U-Net-Backbone zu. Ein anderer Ansatz schlägt vor, klassifikatorfreie, geleitete Diffusionsmodelle in wenige Schritte umfassende generative Modelle zu destillieren, indem zunächst ein kombiniertes bedingtes und unbedingtes Modell destilliert wird und das resultierende Modell dann schrittweise destilliert wird, um eine schnellere Erzeugung zu ermöglichen.

[0009] Während die Destillation ein effektiver Ansatz zur Beschleunigung der Sampling-Rate vorhandener Diffusionsmodelle sein kann, gibt es alternative Arbeitsansätze zur Reduzierung der Länge von Sampling-Ketten durch Berücksichtigung alternativer Formulierungen von Diffusionsmodellen, Korrektur von Bias- und Kürzungsfehlern im Rauschunterdrückungsprozess und durch trainingsfreie schnelle Sampler bei Inferenz. Mehrere verschiedene Arbeiten wie DDIM, Improved DDPM, FastDPM, SGM-CLD und EDM modifizieren oder optimieren den Vorwärtsdiffusionsprozess, sodass der Rauschunterdrückungsprozess effizienter gemacht werden kann. DDIM basiert zum Beispiel auf einer nicht-Markovschen Definition der Diffusionsprozesse, die dasselbe Trainingsziel hat, deren umgekehrter Prozess aber viel schneller ein Sampling durchlaufen kann. DPM-Löser und GENIE sind Löser für gewöhnliche Differentialgleichungen höherer Ordnung („ODE“), die in wenigen Schritten Proben erzeugen. Es gibt auch Ansätze, die Diffusionsmodelle mit anderen Familien generativer Modelle kombinieren, um ein schnelleres Sampling zu ermöglichen.

[0010] Ein Ziel der vorliegenden Offenbarung besteht darin, den Destillationsprozess von Diffusionsmodellen zu rationalisieren und gleichzeitig die Wahrnehmungsqualität der vom ursprünglichen Modell erzeugten Bilder beizubehalten. Offenbarte Ausführungsformen können Verbesserungen gegenüber früheren Ansätzen aufzeigen. Die offenbarten Ausführungsformen können eine einfache und effektive Technik umfassen, die einen Mehrschritt-Diffusionsprozess unter ausschließlicher Verwendung von Rausch-/Bildpaaren in ein generatives Einschritt-Modell destillieren kann. Die offenbarten Ausführungsformen können eine Architektur einschließen, die ein neuartiges Deep Equilibrium-Modell („DEQ“) umfasst, das hier als Generative Equilibrium Transformer

(„GET“) bezeichnet werden kann. In einigen offenbarten Ausführungsformen kann GET als Netzwerk mit unendlicher Tiefe interpretiert werden, das gewichtsgebundene Transformatorschichten verwendet, die einen Fixpunkt im Vorwärtsdurchlauf lösen. Die offenbarten Ausführungsformen ermöglichen eine adaptive Anwendung dieser Schichten im Vorwärtsdurchlauf und erreichen so ein Gleichgewicht zwischen Inferenzgeschwindigkeit und Probenqualität. Darüber hinaus integrieren die offenbarten Ausführungsformen einen nahezu parameterfreien Klassenkonditionierungsmechanismus in die Architektur, wodurch sein Nutzen auf die klassenbedingte Bilderzeugung erweitert wird.

[0011] Die hierin offenbarten Ansätze zur Destillation über Rausch-/Bildpaare, die durch ein Diffusionsmodell erzeugt werden, können sowohl auf Vision Transformer („ViT“) als auch auf GET-basierte Architekturen angewendet werden. In den offenbarten Ausführungsformen erreicht die GET-basierte Architektur jedoch wesentlich bessere Ergebnisse mit kleineren Architekturen. Die GET-basierte Architektur kann im Kontext sowohl der bedingten als auch der unbedingten Bilderzeugung eine wahrnehmbare Bildqualität produzieren, die gegenüber anderen komplexen Destillationstechniken, wie der progressiven Destillation, gleichwertig oder besser ist. In einigen Ausführungsformen zeigt GET im Vergleich zu Architekturen wie ViT eine deutlich bessere Parameter- und Dateneffizienz auf, da GET die Frechet-Inception-Distanz („FID“)-Werte eines fünfmal größeren ViT erreicht, was das transformative Potenzial von GET bei der Verbesserung der Effizienz generativer Modelle unterstreicht.

[0012] Offenbarte Ausführungsformen können zum Generieren synthetischer Bilddatensätze zum Trainieren maschineller Lernsysteme verwendet werden, die für jede Computer Vision-Aufgabe verwendet werden können, einschließlich: 1) computergesteuerte Maschinen, Roboter, Fahrzeuge, Haushaltsgeräte, Elektrowerkzeuge, Fertigungsmaschinen, persönliche Assistenten und Zugangskontrollsysteme; und 2) Systeme zur Beförderung von Informationen, wie Überwachungssysteme oder medizinische (Bildgebungs-) Systeme. Die offenbarten Ausführungsformen können auch verwendet werden, um vorhandene Trainingsdatensätze durch die Erzeugung zusätzlicher Bilder oder durch Modifizieren der Bilder in einem vorhandenen Datensatz zu erweitern.

[0013] Der langsame generative Prozess von Diffusionsmodellen, der Hunderte von Modellbewertungen erfordert, begrenzt ihre Praktikabilität in Echtzeit oder Szenarien mit eingeschränkten Ressourcen. Mit dem Ziel der Beschleunigung des generativen Prozesses sind Destillationsverfahren erschienen, die auf das Destillieren des Mehrschritt-Sampling-Prozesses zu einem effizienteren Schritt mit wenigen oder nur einem Schritt abzielen. Diese Verfahren sind gewöhnlich komplex, da sie mehrere Trainingsphasen erfordern und das resultierende destillierte Modell möglicherweise nicht die Wahrnehmungsqualität der vom ursprünglichen Modell erzeugten Bilder aufweist und bei vielen etablierten Ansätzen die Notwendigkeit besteht, eine doppelte Kopie des Modells aufzubewahren, was wiederum zu einem erhöhten Speicherbedarf führt.

[0014] Tiefe Gleichgewichtsmodelle (Deep Equilibrium - „DEQ“) (z. B. wie beschrieben in Deep Equilibrium Models, Bai et al.) berechnen interne Darstellungen, indem sie in ihrem Vorwärtsdurchlauf einen Fixpunkt lösen. Spezifisch sei ein tiefes Vorwärtsdurchlaufmodell mit L Schichten in Betracht zu ziehen:

$$z^{[i+1]} = f_{\theta}^{[i]} \left(z^{[i]}; x \right) \text{ für } i = 0, \dots, L-1; \quad (1)$$

wobei $x \in \mathbb{R}^{n_x}$ die Eingabeinjektion ist, $z^{[i]} \in \mathbb{R}^{n_z}$ der verborgene Zustand der i -ten Schicht ist und $f_{\theta}^{[i]} : \mathbb{R}^{n_x \times n_z} \rightarrow \mathbb{R}^{n_z}$ die Merkmalstransformation der i -ten Schicht ist, parametrisiert durch θ . Wenn das obige Modell gewichtsgebunden ist, d. h. $f_{\theta}^{[i]} = f_{\theta}, \forall i$, dann nähert sich an der Grenze unendlicher Tiefe die Ausgabe $z^{[i]}$ dieses Netzwerks einem Fixpunkt z^* :

$$\lim_{i \rightarrow \infty} f_{\theta} \left(z^{[i]}; x \right) = f_{\theta} (z^*; x) = z^*. \quad (2)$$

[0015] DEQs lösen diesen Fixpunkt z^* unter Verwendung von Black-Box-Wurzelfindungsalgorithmen wie der Broyden-Methode oder der Anderson-Beschleunigung im Vorwärtsdurchlauf direkt. Da Vorwärtsdurchlauf eine explizite Backpropagation durch exakte Operationen nicht verlässlich ist, nutzen DEQs implizite Differenzierung, um analytisch durch den Fixpunkt zu differenzieren. Wenn $g_{\theta}(z^*; x) = f_{\theta}(z^*; x) - z^*$, dann ist die Jacobi-Matrix von z^* in Bezug auf die Modellgewichtungen θ gegeben durch

$$\frac{\partial \mathbf{z}^*}{\partial \theta} = - \left(\frac{\partial \mathbf{g}_\theta(\mathbf{z}^*, \mathbf{x})}{\partial \mathbf{z}^*} \right)^{-1} \frac{\partial \mathbf{f}_\theta(\mathbf{z}^*; \mathbf{x})}{\partial \theta}. \quad (3)$$

[0016] Die Berechnung der Inversen der Jacobi-Matrix kann bei der Arbeit mit hochdimensionalen Merkmalskarten schnell unlösbar werden. Der inverse Jacobi-Term kann durch eine Identitätsmatrix (d. h. eine Jacobi-freie oder eine ungefähre inverse Jacobi-Matrix) ersetzt werden, ohne dass die endgültige Leistungsfähigkeit dadurch beeinträchtigt wird.

[0017] Diffusionsmodelle oder Score-basierte generative Modelle stören Bilder schrittweise mit einer zunehmenden Menge an Gaußischem Rauschen und kehren diesen Prozess dann durch sequenzielles Entrauschen um, um Bilder zu erzeugen. Hierbei ist insbesondere ein Datensatz von i.i.d-Samples in Betracht zu ziehen p_{data} , dann ist der Diffusionsprozess $\{x(t)\}_{t=0}^T$ für $t \in [0, T]$ durch eine stochastische Differentialgleichung („SDE“) nach Itô gegeben:

$$dx = f(x, t)dt + g(t)dw \quad (4)$$

wobei w der Standard-Wiener-Prozess ist, $f(\cdot, t): \mathbb{R}^d \rightarrow \mathbb{R}^d$ der Driftkoeffizient ist, $g(\cdot): \mathbb{R} \rightarrow \mathbb{R}$ der Diffusionskoeffizient ist und $x(0) \sim p_{\text{data}}$ und $x(T) \sim N(0, I)$.

[0018] Alle Diffusionsprozesse weisen einen entsprechenden deterministischen Prozess auf, der als Wahrscheinlichkeitsfluss-ODE („PF-ODE“) bekannt ist und dessen Trajektorien die gleichen Randwahrscheinlichkeitsdichten wie SDE teilen. ODE kann wie folgt geschrieben werden:

$$dx = -\dot{\sigma}(t)\sigma(t)\nabla_x \log p(x, \sigma(t))dt$$

wobei $\sigma(t)$ der Rauschplan des Diffusionsprozesses ist und $\nabla_x \log p(x, \sigma(t))$ die Bewertungsfunktion darstellt. Es wurde gezeigt, dass die optimale Wahl von $\sigma(t)$ in Gleichung (5) $\sigma(t) = t$ ist. Somit kann PF-ODE vereinfacht werden zu

$$\frac{dx}{dt} = -t\nabla_x \log p(x, \sigma(t)) = \frac{x - D_\theta(x; t)}{t},$$

wobei $D_\theta(\cdot, t)$ eine mit einem neuronalen Netzwerk parametrisierte Entrauschungsfunktion ist, die den erwarteten L_2 Entrauschungsfehler für aus p_{data} gezogene Samples minimiert. Durch numerische Verfahren wie das Euler-Verfahren, das Runge-Kutta-Verfahren und Heuns Löser zweiter Ordnung können aus dieser ODE effizient Samples erzeugt werden.

[0019] Die offenbarten Ausführungsformen umfassen ein destilliertes Einschritt-Diffusionsmodell, das hierin als GET bezeichnet wird. GET kann einen DEQ Vision-Transformator umfassen, der Diffusionsmodelle in generative Modelle destillieren kann, die in der Lage sind, Bilder unter Verwendung nur einer einzelnen Modellauswertung schnell abzutasten. Das GET-Modell kann einen Satz von Gaußischem Rauschen e und optional (in **Fig. 1** durch ein gepunktetes Kästchen angegeben) Klassenbezeichnungen y Bildern $\tilde{x}$ zuordnen.

[0020] Mit Bezug auf **Fig. 1** kann eine Ausführungsform einer GETbasierten Architektur 100 ein Einbettungsnetzwerk 110 zum Einbetten von Rauschen 112, ein Einbettungsnetzwerk 120 zum Einbetten von Klassenbezeichnungen 122, einen Injektionstransformator („InjectionT“) 130 und einen Gleichgewichtstransformator („EquilibriumT“) 140 einschließen. Das Einbettungsnetzwerk 110 kann das Rauschen e 104 in den Einbettungsvektor h 114 einbetten und das Einbettungsnetzwerk 120 kann Klassenbezeichnungen y 122 in den Einbettungsvektor c einbetten, wie gemäß Gleichung (7) gezeigt.

$$h, c = \text{Emb}(e), \text{Emb}(y); \text{ falls } y \neq 0 \quad (7)$$

[0021] Das Rauscheinbettungsnetzwerk 110 kann Eingangsrauschen $e \in \mathbb{R}^{H \times W \times C}$ in eine Sequenz zweidimensionaler („2D“) Patches $p \in \mathbb{R}^{N \times (P^2 \cdot C)}$ umwandeln, wobei C die Anzahl der Kanäle ist, P die Größe des Patches ist, H und W die Höhe und Breite des Originalbildes bezeichnen und $N = \frac{HW}{P^2}$ die resultierende Anzahl der Patches ist. Durch Hinzufügen einer standardmäßigen sinusförmigen Positionscodierung wird der Rau-

scheinbettungsvektor h produziert. Da der gesamte generative Prozess direkt destilliert wird, ist die in Standarddiffusionsmodellen übliche Zeiteinbettung nicht erforderlich.

[0022] Der Injektionstransformator 130 gemäß Gleichung (8) kann tokenisiertes Rauschen (z. B. Einbettungsvektor h) in eine Zwischendarstellung n umwandeln. In **Fig. 1** wird die Ausgabe in eine optionale lineare Komponente 132 eingegeben. Die lineare Komponente 132 kann eine lineare Projektion des Ausgangs des Injektionstransformators 130 durchführen, um die Zwischendarstellung n zu berechnen. Alternativ kann eine lineare Komponente weggelassen werden oder in den Injektionstransformator 132 aufgenommen werden, der dann die Zwischendarstellung n ausgibt. Diese Zwischendarstellung n kann der Eingang für den Gleichgewichtstransformator 140 sein. Der Gleichgewichtstransformator 140 kann gemäß Gleichung (9) eine Gleichgewichtsschicht sein. Der Gleichgewichtstransformator 140 kann den Fixpunkt z^* durch Aufnahme der Rauschinjektion n und eines optionalen Klasseneinbettungsvektors lösen.

$$n = \text{InjectionT}(h, c) \quad (8)$$

$$z^* = \text{EquilibriumT}(z^*, n, c) \quad (9)$$

[0023] Der Injektionstransformator 130 und der Gleichgewichtstransformator 140 können aus einer Sequenz von Transformatorblöcken (d. h. GET-Blöcken) bestehen, die ähnliche Blockausgestaltungen aufweisen, wie in **Fig. 2** veranschaulicht. Sowohl der Injektionstransformator 130 als auch der Gleichgewichtstransformator 140 können aus einer Sequenz von GET-Blöcken bestehen. Der Injektionstransformator 130 kann nur einmal aufgerufen werden, um die Rauschinjektion n zu produzieren. Der Gleichgewichtstransformator 140 definiert eine Funktion f_θ einer impliziten Schicht $z^* = f_\theta(z^*, n, c)$, die mehrmals aufgerufen werden kann und bis zur Konvergenz einen gewichtsgebundenen Berechnungsgraphen erstellt.

[0024] GET-Blöcke können eine ähnliche Blockausgestaltung für die Rauschinjektion (d. h. InjectionT) und die Gleichgewichtsschicht (d. h. EquilibriumT) nutzen, die sich nur an der Injektionsschnittstelle unterscheiden kann. Insbesondere kann der GET-Block auf einem standardmäßigen Pre-LN-Transformatorblock aufgebaut werden, wie unten gezeigt:

$$z = z + \text{Attention}(\text{LN}(z), u)$$

$$z = z + \text{FFN}(\text{LN}(z), u).$$

[0025] Hier stellt $z \in \mathbb{R}^N \times \mathbb{D}$ das latente Token dar, $u \in \mathbb{R}^N \times 3\mathbb{D}$ ist die Eingabeinjektion, LN, FFN und Attention stehen jeweils für Schichtnormalisierung, ein zweischichtiges Feedforward-Netzwerk mit einer verborgenen Dimension der Größe $\mathbb{D} \times \mathbb{E}$ und eine Aufmerksamkeitsschicht mit einer Injektionsschnittstelle.

[0026] Für GET-Blöcke, die in dem Injektionstransformator 130 verwendet werden, ist u gleich dem Klasseneinbettungstoken $c \in \mathbb{R}^1 \times 3\mathbb{D}$ zur bedingten Bilderzeugung. Das heißt $u = c$ für bedingte Modelle und anderenfalls $u = 0$. Für GET-Blöcke in dem Gleichgewichtstransformator ist u die gesendete Summe der Rauschinjektion $n \in \mathbb{R}^N \times \mathbb{D}$ und des Klasseneinbettungstoken $c \in \mathbb{R}^1 \times 3\mathbb{D}$. Das heißt $u = n + c$ für bedingte Modelle und anderenfalls $u = n$.

[0027] Eine Standard-Transformator-Aufmerksamkeitsschicht kann modifiziert werden, um eine additive Injektionsschnittstelle vor der Abfrage $q \in \mathbb{R}^N \times \mathbb{D}$, Eingabe $k \in \mathbb{R}^N \times \mathbb{D}$ und Wert $v \in \mathbb{R}^N \times \mathbb{D}$ zu integrieren,

$$q, k, v = zW_i + u$$

$$z = \text{MHA}(z, k, v)$$

$$z = zW_o$$

wobei $W_i \in \mathbb{R}^{\mathbb{D} \times 3\mathbb{D}}$ und $W_o \in \mathbb{R}^{\mathbb{D} \times \mathbb{D}}$. Die Injektionsschnittstelle ermöglicht Interaktionen zwischen den latenten Token und der Eingabeinjektion im Multi-Head-Dot-Product-Attention-Vorgang („MHA“),

$$qk^T = (zW_q + u_q)(zW_k + u_k)^T = zW_qW_k^Tz^T + zW_qu_k^T + u_qW_k^Tz^T + u_q^Tu_k,$$

wobei $W_q, W_k \in \mathbb{R}^{D \times D}$ Slices von W_i und u_q sind, $W_k \in \mathbb{R}^{D \times D}$ Slices von u sind. Dieses Schema fügt im Vergleich zum Standard-MHA-Vorgang keinen höheren Rechenaufwand hinzu, erzielt jedoch einen ähnlichen Effekt wie Cross-Attention und bietet eine gute Stabilität während des Trainings.

[0028] Die Ausgabe des Gleichgewichtstransformators 140, z^* , kann decodiert und neu angeordnet werden, um eine Bildprobe $\tilde{x}$ 160 gemäß Gleichung (10) zu erzeugen. Der Decoder 150 kann eine Schichtnormalisierungskomponente und eine lineare Schicht umfassen, um Patches $\bar{p} \in \mathbb{R}^N \times D$ zu erzeugen. Die resultierenden Patches $\bar{p}$ könne wieder auf die Auflösung des Eingangsrauschens ezurückgesetzt werden, um eine Bildprobe $\tilde{x} = R^H \times W \times C$ zu produzieren.

$$\tilde{x} = \text{Decoder} (z^*). \quad (10)$$

[0029] Fig. 2 veranschaulicht ein Ausführungsbeispiel 200 zum Erzeugen von Bildproben gemäß der vorliegenden Offenbarung. In Fig. 2 bettet ein Einbettungsnetzwerk 202 Rauschen e in einen Einbettungsvektor h (nicht gezeigt) ein und der Einbettungsvektor h wird in einen Injektionstransformator eingegeben. Bei dem Injektionstransformator kann es sich zum Beispiel um den in Fig. 1 veranschaulichten Injektionstransformator 130 handeln. Der Injektionstransformator kann einen oder mehrere GET-Blöcke 214 einschließen. Der Einbettungsvektor h kann nach einer Schichtnormalisierungskomponente 206 des GET-Blocks in einen GET-Block eingegeben werden. Das Einbettungsnetzwerk 202 kann zum Beispiel dasselbe Einbettungsnetzwerk 110 sein, das in Fig. 1 gezeigt ist. In Fig. 2 bettet das optionale (durch ein gepunktetes Kästchen angegeben) Einbettungsnetzwerk 204 Klassenbezeichnungen y in einen Einbettungsvektor c ein, und der Einbettungsvektor c wird nach der Normalisierungsschicht 206 in den Injektionstransformator eingegeben.

[0030] Der Injektionstransformator in dieser Ausführungsform 200 kann einen oder mehrere GET-Blöcke 214 einschließen. Jeder GET-Block 214 schließt die Schichtnormalisierungskomponenten 206, 210 ein. Die Schichtnormalisierung ist eine Technik, die beim Deep Learning verwendet wird, um die Aktivierungen (Ausgabe) einer neuronalen Netzwerkschicht zu normalisieren. Sie normalisiert jedes Trainingsbeispiel unabhängig über seine Merkmale hinweg und reduziert so die Abhängigkeit von der Batchgröße. Die Schichtnormalisierungskomponente 206 empfängt als Eingabe den Einbettungsvektor h (nicht gezeigt) vom Einbettungsnetzwerk 202.

[0031] Jeder GET-Block 214 schließt eine Multi-Head-Attention-Komponente („MHA“) 208 ein. Multi-Head Attention ist eine Komponente, die in transformatorbasierten Modellen verwendet wird. Attention-Mechanismen ermöglichen es Modellen, sich beim Treffen von Vorhersagen oder beim Erzeugen von Ausgaben auf unterschiedliche Teile der Eingabesequenz zu konzentrieren. Multi-Head Attention erweitert diese Idee durch die Verwendung mehrerer Sätze von Aufmerksamkeitsgewichten, sogenannter „Attention Heads“, um unterschiedliche Arten von Beziehungen und Informationen in der Eingabesequenz zu erfassen. Die MHA-Komponente 208 kann als Eingabe die Ausgabe von der Schichtnormalisierungskomponente 206 empfangen. Die MHA-Komponente 208 kann optional den Einbettungsvektor c als Eingabe empfangen.

[0032] Jeder GET-Block 214 schließt eine zweite Schichtnormalisierungskomponente 210 ein. Die Schichtnormalisierungskomponente 210 kann als Eingabe die Ausgabe der MHA-Komponente 208 empfangen.

[0033] Jeder GET-Block 214 schließt eine Multi-Layer Perceptron-Komponente („MLP“) 212 ein. Eine MLP kann aus linearen Schichten kombiniert mit nichtlinearen Aktivierungsfunktionen bestehen. Die linearen Schichten führen Transformationen durch, die Gewichte und Verzerrungen beinhalten, während die nichtlinearen Aktivierungsfunktionen Nichtlinearitäten in das Netzwerk einführen. Die MLP-Komponente 212 kann als Eingabe die Ausgabe von der zweiten Schichtnormalisierungskomponente 210 empfangen.

[0034] In Fig. 2 wird die Ausgabe n des Injektionstransformators in einen Gleichgewichtstransformator eingegeben. Der Gleichgewichtstransformator kann zum Beispiel der in Fig. 1 veranschaulichte Gleichgewichtstransformator 140 sein und kann einen oder mehrere GET-Blöcke 220 einschließen. Der Gleichgewichtstransformator in Fig. 2 kann optional als zusätzliche Eingabe für n den Einbettungsvektor c empfangen. Wie in Fig. 2 veranschaulicht, weisen der eine oder die mehreren GET-Blöcke 220 eine ähnliche Struktur wie der GET-Block 214 im Injektionstransformator auf. Das heißt, die GET-Blöcke 220 weisen eine erste Schichtnormalisierungskomponente 222 ähnlich der Schichtnormalisierungskomponente 206, eine MHA-Komponente 224 ähnlich der MHA-Komponente 208, eine zweite Schichtnormalisierungskomponente 226 ähnlich der Schichtnormalisierungskomponente 210 und eine MLP-Komponente 228 ähnlich der MLP-Komponente 212 auf.

[0035] Der Gleichgewichtstransformator in **Fig. 2** löst den Fixpunkt z^* . Der Fixpunkt z^* kann decodiert und neu angeordnet werden, um eine Bildprobe zu erzeugen. Zum Beispiel kann der Fixpunkt z^* einem Decoder, wie dem Decoder 150 von **Fig. 1**, bereitgestellt werden, um eine Bildprobe $\tilde{x}$ zu erzeugen.

[0036] **Fig. 3** veranschaulicht ein Ausführungsbeispiel eines allgemeinen Computersystems 300 gemäß der vorliegenden Offenbarung. Das Computersystem 300 kann einen Satz von Anweisungen einschließen, die ausgeführt werden können, um das Computersystem 300 zu veranlassen, eines oder mehrere der hierin offenbarten Verfahren oder computergestützten Funktionen durchzuführen. Zum Beispiel kann das Computersystem 300 ausführbare Anweisungen zum Durchführen von Funktionen der in **Fig. 1** und **Fig. 2** veranschaulichten Komponenten einschließen. Das Computersystem 300 kann über ein Netzwerk mit anderen Computersystemen oder Peripheriegeräten verbunden sein. Darüber hinaus kann das Computersystem 300 andere Rechenvorrichtungen einschließen oder darin eingeschlossen sein.

[0037] Wie in **Fig. 3** veranschaulicht, kann das Computersystem 300 einen oder mehrere Prozessoren 302 einschließen. Der eine oder die mehreren Prozessoren 302 können zum Beispiel eine oder mehrere zentrale Verarbeitungseinheiten (CPUs), eine oder mehrere Grafikprozessoreinheiten (GPUs) oder beides einschließen. Das Computersystem 300 kann einen Hauptspeicher 304 und einen statischen Speicher 306 einschließen, die über einen Bus 308 miteinander kommunizieren können. Wie gezeigt, kann das Computersystem 300 ferner eine Videoanzeigeeinheit 310 einschließen, wie eine Flüssigkristallanzeige (LCD), eine Projektionsfernsehanzeige, eine Flachbildschirmanzeige, eine Plasmaanzeige oder eine Festkörperanzeige. Zusätzlich kann das Computersystem 300 eine Eingabevorrichtung 312 einschließen, wie eine Fernbedienungsvorrichtung mit einem drahtlosen Tastenfeld, einer Tastatur, einem an eine Spracherkennungsmaschine gekoppelten Mikrofon, einer Kamera, wie einer Videokamera oder einer Standbildkamera, oder einer Cursor-Steuervorrichtung 314, wie einer Maus-Vorrichtung. Das Computersystem 300 kann auch eine Festplattenlaufwerkeinheit 316, eine Signalerzeugungsvorrichtung 318, wie einen Lautsprecher, und eine Netzwerkschnittstellenvorrichtung 320 einschließen. Die Netzwerkschnittstelle 320 kann es dem Computersystem 300 ermöglichen, über ein Netzwerk 328 mit anderen Systemen zu kommunizieren. Zum Beispiel kann die Netzwerkschnittstelle 320 dem maschinellen Lernsystem 120 ermöglichen, mit einem Datenbankserver (nicht gezeigt) oder einer Steuerung in einem Fertigungssystem (nicht gezeigt) zu kommunizieren.

[0038] In einigen Ausführungsformen, wie in **Fig. 3** dargestellt, kann die Festplattenlaufwerkeinheit 316 ein oder mehrere computerlesbare Medien 322 einschließen, in die ein oder mehrere Anweisungssätze 324, z. B. Software, eingebettet sein können. Zum Beispiel können die Anweisungen 324 eines oder mehrere der Verfahren oder Funktionen verkörpern, wie die hierin offenbarten Verfahren oder Funktionen. In einer bestimmten Ausführungsform können die Anweisungen 324 während der Ausführung durch das Computersystem 300 vollständig oder mindestens teilweise im Hauptspeicher 304, im statischen Speicher 306 und/oder im Prozessor 302 liegen. Der Hauptspeicher 304 und der Prozessor 302 können auch computerlesbare Medien einschließen.

[0039] In einigen Ausführungsformen können dedizierte Hardwareimplementierungen, wie anwendungsspezifische integrierte Schaltungen, programmierbare Logikarrays und andere Hardwarevorrichtungen, konstruiert werden, um eines oder mehrere der hierin beschriebenen Verfahren oder Funktionen zu implementieren. Anwendungen, die die Einrichtungen und Systeme verschiedener Ausführungsformen einschließen können, können grob eine Vielfalt elektronischer und Computersysteme einschließen. Eine oder mehrere der hierin beschriebenen Ausführungsformen können Funktionen unter Verwendung von zwei oder mehr spezifischen miteinander verbundenen Hardwaremodulen oder -vorrichtungen mit zugehörigen Steuer- und Datensignalen implementieren, die zwischen und durch die Module oder als Teile einer anwendungsspezifischen integrierten Schaltung kommuniziert werden können. Dementsprechend umschließt das vorliegende System Software-, Firmware- und Hardwareimplementierungen oder Kombinationen davon.

[0040] Obschon das computerlesbare Medium als einzelnes Medium gezeigt ist, beinhaltet der Begriff „computerlesbares Medium“ ein einzelnes Medium oder mehrere Medien, wie eine zentralisierte oder verteilte Datenbank und/oder zugeordnete Caches und Server, die einen oder mehrere Anweisungssätze speichern. Der Begriff „computerlesbares Medium“ soll auch jedes Medium einschließen, das in der Lage ist, einen Satz von Anweisungen zur Ausführung durch einen Prozessor zu speichern oder zu codieren oder das ein Computersystem dazu veranlasst, eines oder mehrere der hierin beschriebenen Verfahren oder Funktionen durchzuführen.

[0041] In einigen Ausführungsformen sind einige oder alle computerlesbaren Medien nicht-transitorische Medien. In einem bestimmten, nicht einschränkenden, Ausführungsbeispiel kann das computerlesbare

Medium einen Festkörperspeicher wie eine Speicherkarte oder ein anderes Gehäuse einschließen, das einen oder mehrere nichtflüchtige Nur-Lese-Speicher einhäust. Ferner kann es sich bei dem computerlesbaren Medium um einen Direktzugriffsspeicher oder einen anderen flüchtigen wiederbeschreibbaren Speicher handeln. Zusätzlich kann das computerlesbare Medium ein magnetooptisches oder optisches Medium einschließen, wie eine Diskette oder Bänder oder eine andere Speichervorrichtung zum Erfassen von Trägerwellensignalen, wie ein über ein Übertragungsmedium kommuniziertes Signal.

[0042] Hierin werden ein Generative Equilibrium Transformer („GET“), ein Deep Equilibrium Vision Transformer, der sich gut für generative Modelle mit nur einem Schritt eignet, offenbart. Die einzigartige Architektur von GET ermöglicht ein Gleichgewicht zwischen Inferenzgeschwindigkeit und Qualität der erzeugten Bilder. Außerdem wird hierin ein stromlinienförmiger Diffusionsdestillationsprozess und eine Offenbarung offenbart, dass das direkte Training anhand von Rausch-/Bildpaaren aus Diffusionsmodellen (mit der GET-Architektur) sowohl für klassenbedingte als auch für nicht klassenbedingte Fälle eine effektive Strategie zum Destillieren einer Mehrschritt-Sampling-Kette in ein generatives Ein-Schritt-Modell ist. Eine oder mehrere offenbarte Ausführungsformen zeigen, dass implizite Modelle für generative Aufgaben klassische Netzwerke im Hinblick auf Leistungsfähigkeit, Modellgröße, Modellberechnung, Trainingspeicher und Geschwindigkeit deutlich überreffen können.

[0043] Obwohl vorstehend Ausführungsbeispiele beschrieben wurden, ist nicht beabsichtigt, dass diese Ausführungsformen alle möglichen Formen beschreiben, die von den Ansprüchen umschlossen sind. Die in der Patentschrift verwendeten Wörter dienen der Beschreibung und nicht der Einschränkung und es versteht sich, dass verschiedene Änderungen vorgenommen werden können, ohne vom Geist und Umfang der Offenbarung abzuweichen. Wie zuvor beschrieben, können die Merkmale verschiedener Ausführungsformen kombiniert werden, um weitere Ausführungsformen der Erfindung zu bilden, die möglicherweise nicht explizit beschrieben oder veranschaulicht sind. Obschon verschiedene Ausführungsformen als Vorteile bereitstellend oder gegenüber anderen Ausführungsformen oder Implementierungen aus dem Stand der Technik in Bezug auf eine oder mehrere gewünschte Eigenschaften bevorzugt beschrieben werden könnten, ist dem Fachmann klar, dass bei einem oder mehreren Merkmalen oder Eigenschaften Kompromisse eingegangen werden können, um die gewünschten Gesamtsystemattribute zu erreichen, die von der spezifischen Anwendung und Implementierung abhängen. Diese Attribute können Festigkeit, Haltbarkeit, Marktfähigkeit, Aussehen, Verpackung, Größe, Wartungsfreundlichkeit, Gewicht, Herstellbarkeit, einfache Montage usw. einschließen, ohne darauf beschränkt zu sein. Somit sind Ausführungsformen, die in Bezug auf eine oder mehrere Eigenschaften als weniger wünschenswert als andere Ausführungsformen oder Implementierungen nach dem Stand der Technik beschrieben werden, nicht außerhalb des Umfangs der Offenbarung und können für bestimmte Anwendungen wünschenswert sein.

Patentansprüche

1. Verfahren, umfassend:
Umwandeln von Rauschen in einen Rauscheinbettungsvektor;
Tokenisieren des Rauscheinbettungsvektors über einen Injektionstransformator;
Eingeben des tokenisierten Rauschens in einen Gleichgewichtstransformator;
Lösen eines Fixpunkts über den Gleichgewichtstransformator; und
Decodieren des Fixpunkts zum Erzeugen einer Bildprobe.
2. Verfahren nach Anspruch 1, wobei der Injektionstransformator eine Abfolge von Transformatorblöcken einschließt.
3. Verfahren nach Anspruch 2, wobei jeder Transformatorblock einschließt:
eine Normalisierungskomponente der ersten Schicht, die den Rauscheinbettungsvektor empfängt;
eine Normalisierungskomponente der zweiten Schicht;
eine Multi-Head-Attention-Komponente zwischen der Normalisierungskomponente der ersten Schicht und der Normalisierungskomponente der zweiten Schicht; und
eine Multi-Layer-Perceptron-Komponente.
4. Verfahren nach Anspruch 1, wobei der Gleichgewichtstransformator eine Abfolge von Transformatorblöcken einschließt.
5. Verfahren nach Anspruch 4, wobei jeder Transformatorblock einschließt:
eine Normalisierungskomponente der ersten Schicht, die das tokenisierte Rauschen empfängt;

eine Normalisierungskomponente der zweiten Schicht;
 eine Multi-Head-Attention-Komponente zwischen der Normalisierungskomponente der ersten Schicht und der Normalisierungskomponente der zweiten Schicht; und
 eine Multi-Layer-Perceptron-Komponente.

6. Verfahren nach Anspruch 1, wobei die Decodierung durch einen Decoder durchgeführt wird, der eine Schichtnormalisierungskomponente und eine lineare Schicht umfasst.

7. Verfahren nach Anspruch 1, ferner umfassend:
 Umwandeln einer Klassenbezeichnung in einen Klasseneinbettungsvektor;
 Eingeben des Klasseneinbettungsvektors in den Injektionstransformator vor dem Tokenisieren des Rauscheinbettungsvektors; und
 Eingeben des Klasseneinbettungsvektors in den Gleichgewichtstransformator vor dem Lösen des Fixpunkts.

8. Nicht-transitorischer Speicher, der computerausführbare Anweisungen einschließt, die, wenn sie von einem System ausgeführt werden, das System veranlassen, Vorgänge durchzuführen, die einschließen:
 Umwandeln von Rauschen in einen Rauscheinbettungsvektor;
 Tokenisieren des Rauscheinbettungsvektors über einen Injektionstransformator;
 Eingeben des tokenisierten Rauschens in einen Gleichgewichtstransformator;
 Lösen eines Fixpunkts über den Gleichgewichtstransformator; und
 Decodieren des Fixpunkts zum Erzeugen einer Bildprobe.

9. Speicher nach Anspruch 8, wobei der Injektionstransformator eine Abfolge von Transformatorblöcken einschließt.

10. Speicher nach Anspruch 9, wobei jeder Transformatorblock einschließt:
 eine Normalisierungskomponente der ersten Schicht, die den Rauscheinbettungsvektor empfängt;
 eine Normalisierungskomponente der zweiten Schicht;
 eine Multi-Head-Attention-Komponente zwischen der Normalisierungskomponente der ersten Schicht und der Normalisierungskomponente der zweiten Schicht; und
 eine Multi-Layer-Perceptron-Komponente.

11. Speicher nach Anspruch 8, wobei der Gleichgewichtstransformator eine Abfolge von Transformatorblöcken einschließt.

12. Speicher nach Anspruch 11, wobei jeder Transformatorblock einschließt:
 eine Normalisierungskomponente der ersten Schicht, die das tokenisierte Rauschen empfängt;
 eine Normalisierungskomponente der zweiten Schicht;
 eine Multi-Head-Attention-Komponente zwischen der Normalisierungskomponente der ersten Schicht und der Normalisierungskomponente der zweiten Schicht; und
 eine Multi-Layer-Perceptron-Komponente.

13. Speicher nach Anspruch 8, wobei die Decodierung durch einen Decoder durchgeführt wird, der eine Schichtnormalisierungskomponente und eine lineare Schicht umfasst.

14. Speicher nach Anspruch 8, wobei die Vorgänge ferner einschließen:
 Umwandeln einer Klassenbezeichnung in einen Klasseneinbettungsvektor;
 Eingeben des Klasseneinbettungsvektors in den Injektionstransformator vor dem Tokenisieren des Rauscheinbettungsvektors; und
 Eingeben des Klasseneinbettungsvektors in den Gleichgewichtstransformator vor dem Lösen des Fixpunkts.

15. System, umfassend:
 einen oder mehrere Prozessoren; und
 einen nicht-transitorischen Speicher, der computerausführbare Anweisungen einschließt, die, wenn sie von dem einen oder den mehreren Prozessoren ausgeführt werden, das System veranlassen, Vorgänge durchzuführen, die einschließen:
 Umwandeln von Rauschen in einen Rauscheinbettungsvektor;
 Tokenisieren des Rauscheinbettungsvektors über einen Injektionstransformator;

Eingeben des tokenisierten Rauschens in einen Gleichgewichtstransformator;
Lösen eines Fixpunkts über den Gleichgewichtstransformator; und
Decodieren des Fixpunkts zum Erzeugen einer Bildprobe.

16. System nach Anspruch 15, wobei der Injektionstransformator eine Abfolge von Transformatorblöcken einschließt.

17. System nach Anspruch 16, wobei jeder Transformatorblock einschließt:
eine Normalisierungskomponente der ersten Schicht, die den Rauscheinbettungsvektor empfängt;
eine Normalisierungskomponente der zweiten Schicht;
eine Multi-Head-Attention-Komponente zwischen der Normalisierungskomponente der ersten Schicht und der Normalisierungskomponente der zweiten Schicht; und
eine Multi-Layer-Perceptron-Komponente.

18. System nach Anspruch 15, wobei der Gleichgewichtstransformator eine Abfolge von Transformatorblöcken einschließt.

19. System nach Anspruch 18, wobei jeder Transformatorblock einschließt:
eine Normalisierungskomponente der ersten Schicht, die das tokenisierte Rauschen empfängt;
eine Normalisierungskomponente der zweiten Schicht;
eine Multi-Head-Attention-Komponente zwischen der Normalisierungskomponente der ersten Schicht und der Normalisierungskomponente der zweiten Schicht; und
eine Multi-Layer-Perceptron-Komponente.

20. System nach Anspruch 15, wobei die Decodierung durch einen Decoder durchgeführt wird, der eine Schichtnormalisierungskomponente und eine lineare Schicht umfasst.

Es folgen 3 Seiten Zeichnungen

Anhängende Zeichnungen

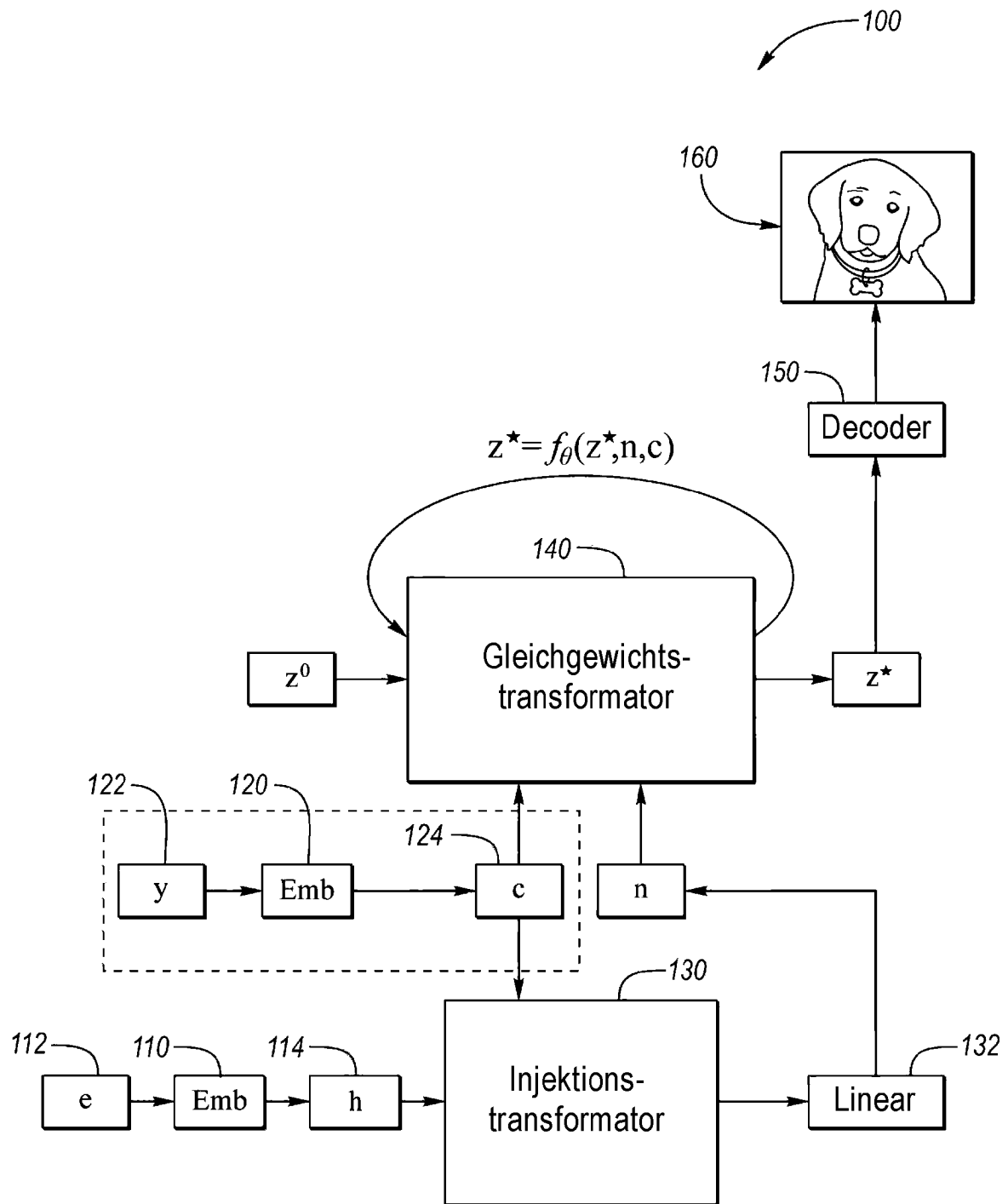


FIG. 1

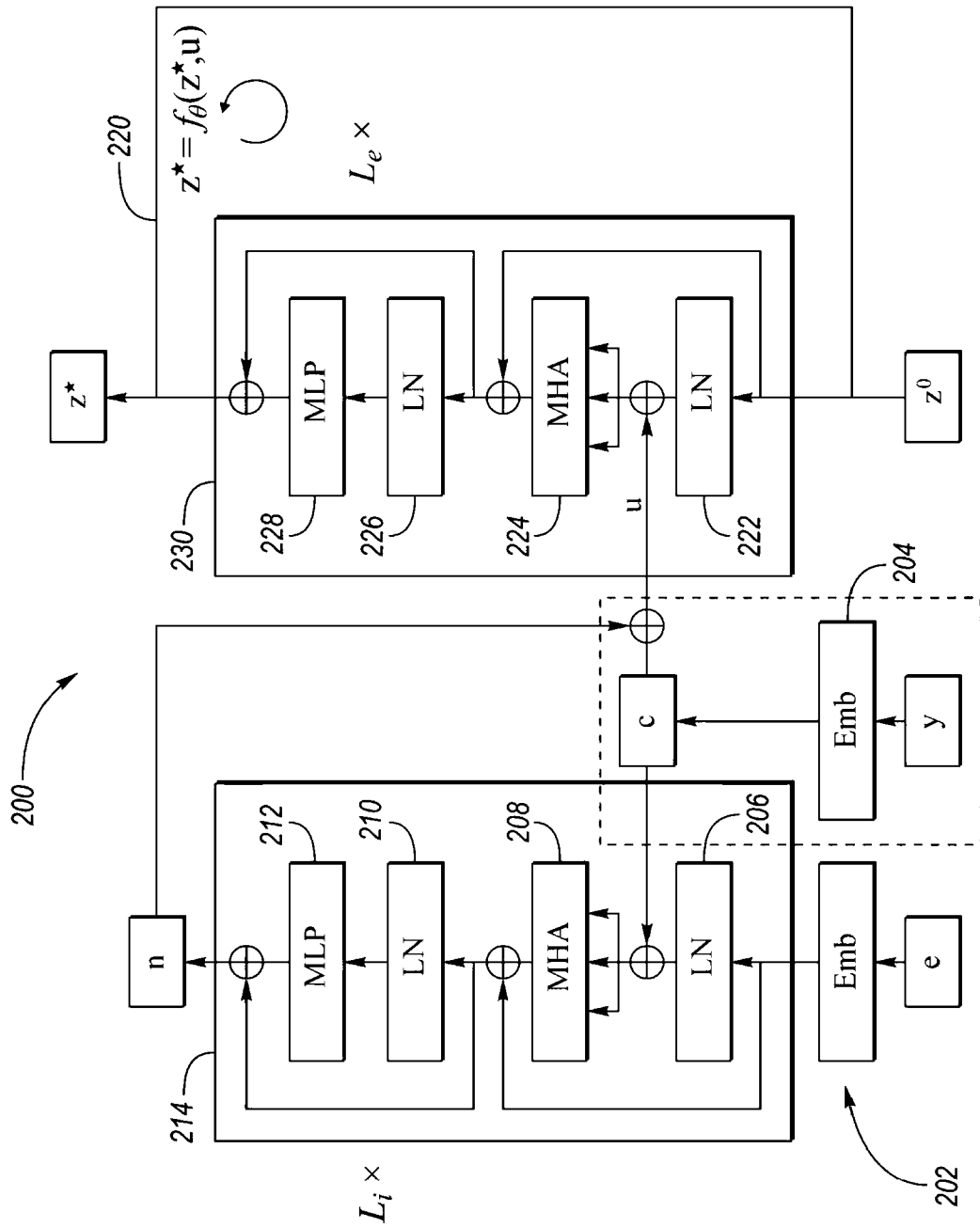
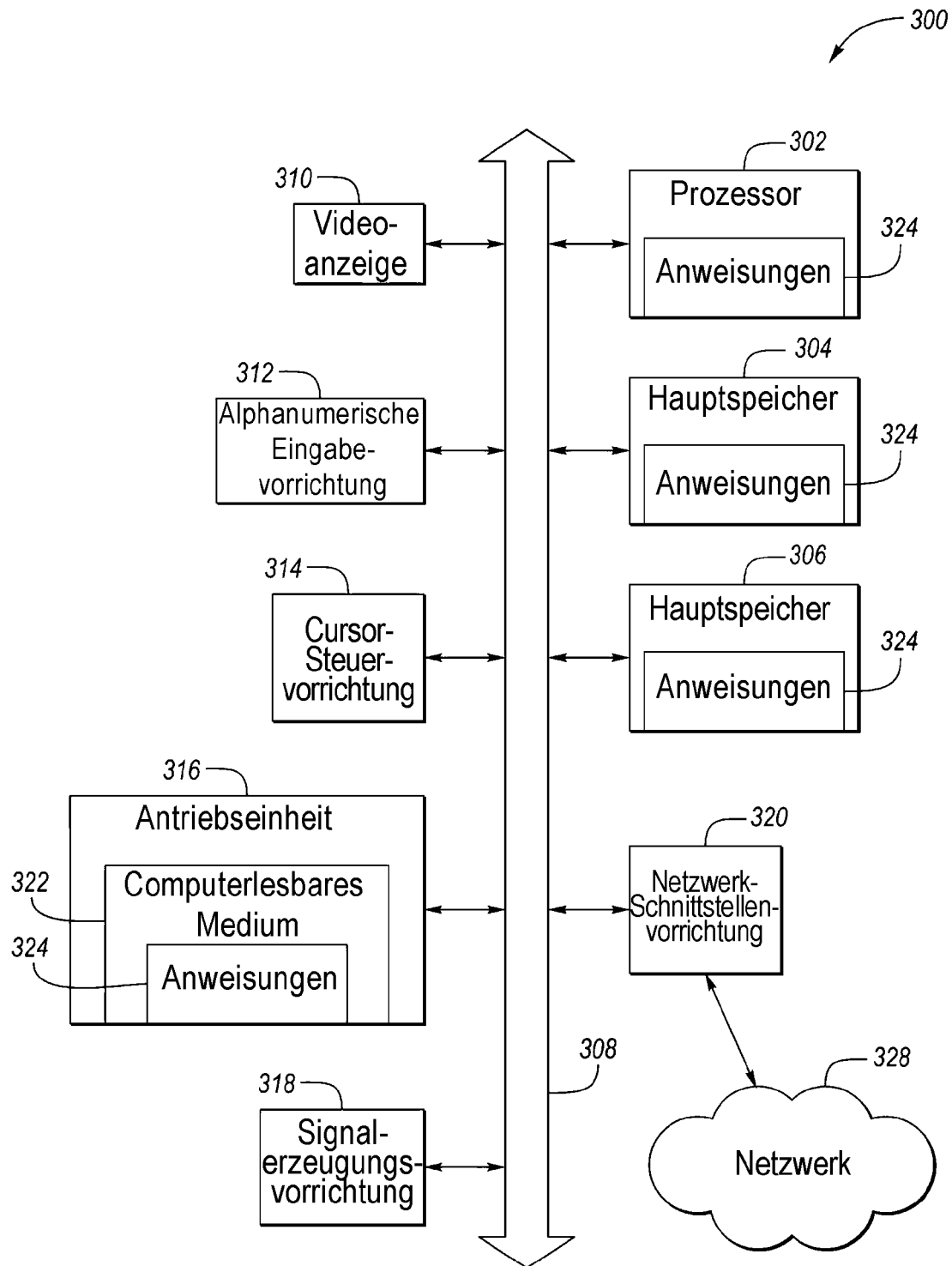


FIG. 2

**FIG. 3**