KI für Alle 2: Verstehen, Bewerten, Reflektieren

Themenblock Generative Modelle: 08_05Modellauswahl_Diffusion

Diffusionsmodelle

Erarbeitet von
Dr. Ann-Kathrin Selker

Lernziele

- Du kannst Diffusionsmodelle an einem Beispiel erklären

Projekt ai4all „KI für Alle"
HeiCAD –Heine Center for Artificial Intelligence and Data Science

Inhalt

Einstieg

Heutzutage ist es möglich, innerhalb kürzester Zeit mit der Hilfe von Textprompts fotorealistische, hochauflösende Bilder zu erzeugen. Doch wie funktioniert das eigentlich?

In diesem Video nehmen wir einmal sogenannte Diffusionsmodelle unter die Lupe, deren Prinzip 2015 vorgestellt wurden.

Quelle [1]

Angenommen, wir haben die folgenden Bilder.

Bilder aus dem Datensatz CelebA-HQ (**Quelle [2]**)

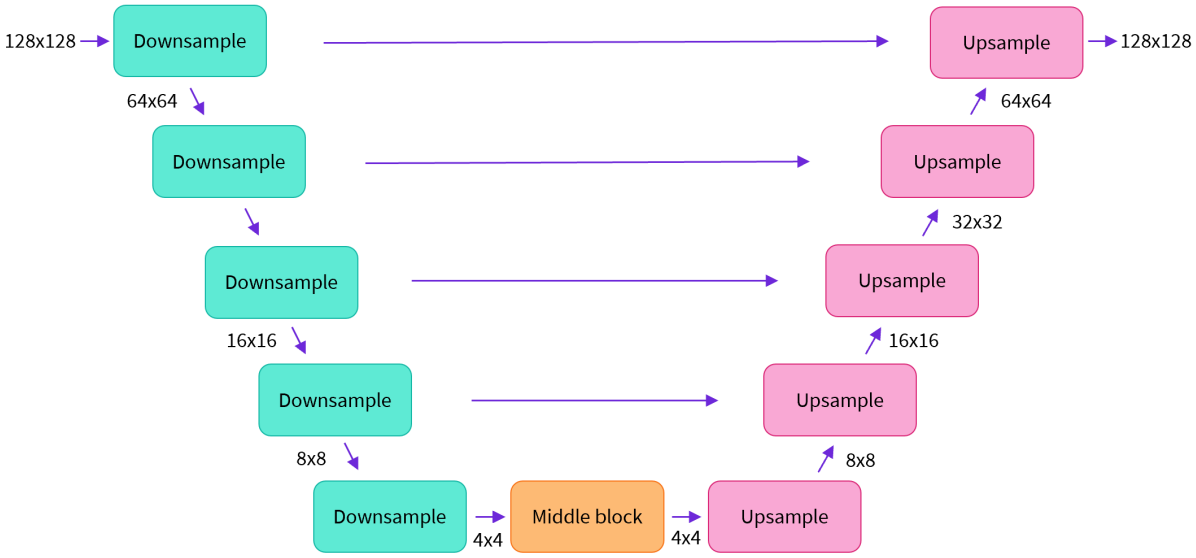
Es handelt sich hier um Bilder von berühmten Personen. Gucken wir uns doch mal an, wie wir anhand dieser Bilder neue Bilder mithilfe von Diffusionsmodellen generieren können.

Architektur

Ein Diffusionsmodell ist ein UNet. UNets werden auch in der Bildsegmentierung verwendet. Dabei handelt es sich um ein neuronales Netz, genauer gesagt ein Convolutional Neural Network (CNN), das Informationen (auch Kontext genannt) an höhere Schichten übergeben kann.

Quelle [3]

Der Name leitet sich von seiner Form ab: Wie hier im Bild zu sehen ist, ordnen sich die Schichten U-förmig an.



U-Net (**Quelle [4]**)

Um so einem UNet beizubringen, Bilder zu generieren, gehen wir einen kleinen Umweg: Wir trainieren das UNet darauf, in einem gegebenen Bild Rauschen zu identifizieren. Zur Erinnerung: Unter Rauschen (engl. noise) bei Bildern verstehen wir Abweichungen bei den Pixelwerten vom Originalbild. Im Folgenden verwenden wir Gaußsches Rauschen, benannt nach der Gaußschen Normalverteilung des deutschen Mathematikers Johann Carl Friedrich Gauß. Das Rauschen wird dabei so erzeugt, dass kleine Abweichungen vom Originalwert viel wahrscheinlicher sind als große Abweichungen. Das folgt dem Prinzip der Normalverteilung, bei dem Ergebnisse um den Erwartungswert herum wahrscheinlicher sind als Ergebnisse, die weiter entfernt liegen.

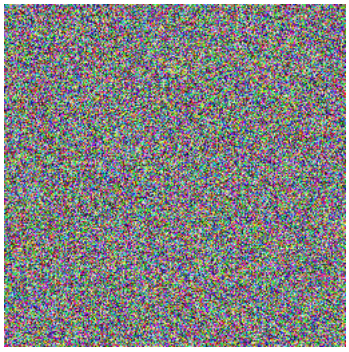
Wenn wir den Bildern aus unserem Datensatz Rauschen hinzufügen, erhalten wir z. B. diese Bilder.

Ziel des UNets ist es, gegeben ein verrauschtes Bild, das vorhandene Rauschen zu identifizieren. Dafür gibt es ein Rauschen aus, das es für die richtige Antwort hält. Dieses ausgegebene Rauschen wird daraufhin mit dem tatsächlichen Rauschen verglichen. Die Höhe des Verlustes berechnet sich dann nach den Abweichungen zwischen beiden. Da das Ziel beim Training ja ist, die Verluste zu minimieren, lernt das Machine Learning Modell dadurch also, das Rauschen besser vorherzusagen.

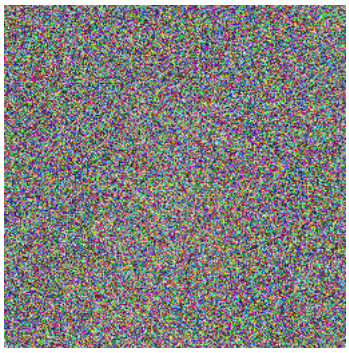
Zusammengefasst funktioniert das Training folgendermaßen: In einem sogenannten Vorwärtsprozess fügen wir unseren Trainingsbildern schrittweise immer mehr Rauschen hinzu, bis die Bilder am Ende nur noch aus reinem Rauschen bestehen. Im dazugehörigen Rückwärtsprozess versucht unser Modell dann schrittweise, das Rauschen des jeweiligen Schritts richtig zu identifizieren und zu entfernen, bis am Ende idealerweise das ursprüngliche Trainingsbild wiederhergestellt wurde.

Bildgenerierung

Ein Diffusionsmodell identifiziert also Rauschen in Bildern. Das hilft auf den ersten Blick noch nicht dabei, neue Bilder zu generieren. Aber angenommen, wir nehmen ein zufällig erstelltes Rauschen wie dieses hier und lassen das auf Berühmtheiten trainierte Modell damit den Rückwärtsprozess durchführen.

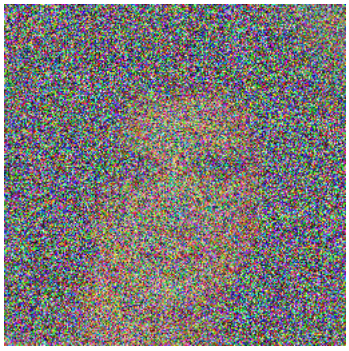


Das Modell versucht jetzt schrittweise das Rauschen zu identifizieren, was dieses Bild von einem Bild einer berühmten Person unterscheidet – das ist immerhin das, was wir dem Modell beim Training im Rückwärtsprozess beigebracht haben. Dieses jeweils identifizierte Rauschen wird in jedem Schritt entfernt und das Resultat als Ausgangsbild für den nächsten Schritt genommen. Je nachdem, welche Variante du genau verwendest, kann es schon mal einige Schritte dauern, bis du eine Person erkennen kannst. In dem von mir benutzten Modell erkennst du selbst nach 700 Schritten noch nichts.

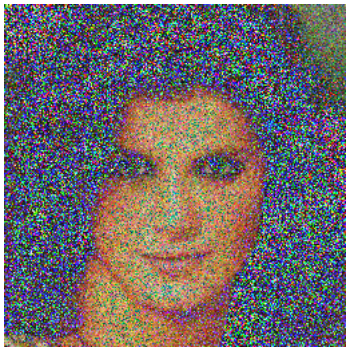


Erstellt mit Code aus **Quelle [4]**

Nach 800 Schritten ist langsam ein Gesicht zu erkennen. Es könnte eine Frau sein.



Nach 900 Schritten kann man eine junge, dunkelhaarige Frau erkennen,



und nach 1000 Schritten ist ein fertiges Bild entstanden und unser Eindruck hat sich bestätigt.

Nach diesem Prinzip kann das Modell also neue Bilder aus reinem Rauschen generieren, die es vorher noch nicht gegeben hat.

Kontext hinzufügen

Beim Diffusionsmodell wird dem UNet ein zeitlicher Kontext (sprich die Nummer des aktuellen Schritts) übergeben, weil je nach Schritt das hinzugefügte Rauschen anders berechnet wird. Es ist aber auch möglich, zusätzlich noch einen weiteren Kontext zu übergeben.

Du kennst bereits im Zusammenhang mit Textrepräsentation das Prinzip der Einbettung. Noch einmal zur Erinnerung: Wir haben gezeigt, dass man Wörter als Vektor darstellen kann.
Vektoren geben einen Punkt im Raum an. Ähnliche Wörter haben auch einen ähnlichen Vektor, sprich sie befinden sich auch im Raum nah beieinander. Die Einbettungen berechnen sich sogar so, dass man am Ende mit ihnen sinnvoll rechnen kann: Mit der sogenannten Vektorarithmetik berechnet sich zum Beispiel mithilfe der jeweiligen Vektoren aus Berlin – Deutschland + Frankreich der Vektor für Paris.

Mit diesem Prinzip ist es möglich, als Kontext eine komplette Bildbeschreibung in Vektorform an das UNet zu übergeben. Dadurch lernt das Modell zusätzlich noch den Zusammenhang zwischen der Beschreibung und dem vorhandenen Rauschen. Und durch die Vektorarithmetik kann das Modell auch mit Beschreibungen (sprich Vektoren) umgehen, die es während des Trainings noch nicht gesehen hat.

Stable Diffusion

Etwas bekannter als einfache Diffusionsmodelle ist dir vielleicht die Anwendung Stable Diffusion, mit der du ebenfalls mit Textbeschreibungen Bilder generieren kannst. Stable Diffusion manipuliert nicht die tatsächlichen Pixelwerte eines Bildes, sondern benutzt stattdessen Bildeinbettungen. Dies funktioniert nach demselben Prinzip wie Texteinbettungen: Bildern werden Vektoren zugewiesen, die ihren Platz im Raum angeben (dem sogenannten Latenzraum, engl. Latent Space). Ähnliche Vektoren bedeuten ähnliche Bilder. Der Diffusionsprozess wird dann nicht mehr auf die tatsächlichen Bilder, sondern

stattdessen auf diese Bildvektoren angewendet. Dieser Vorgang spart nicht nur sehr viel Zeit ein, sondern ermöglicht auch die Erzeugung von Bildern mit viel höherer Auflösung als zuvor.

Doch wieso genau nennen sich diese Machine Learning Modelle jetzt eigentlich Diffusionsmodelle? Unter Diffusion versteht man in der Biologie bzw. Physik ganz grob gesagt die (selbstständige) Vermischung zweier Stoffe, wie dieser Tropfen rote Tinte, der sich selbstständig nach und nach mit dem Wasser mischt und sich die rote Farbe daher im Wasser ausbreitet.

Video: Tropfen Wasser in Tinte (**Quelle [5]**)

Genau dieses Prinzip findet sich auch bei den Diffusionsmodellen wieder: Es gibt die real existierenden Bilder (Tropfen Tinte) und alle möglichen Bilder, sprich alle Kombinationen aus Pixeln, die möglich sind (Wasser). Durch Hinzufügen von Rauschen breiten sich die existierenden Bilder immer weiter im Raum der möglichen Bilder aus.

Abschluss

Dieses Video behandelt das Thema Diffusionsmodelle natürlich nur sehr oberflächlich. Für eine tiefere und genauere Erklärung von Diffusionsmodellen benötigst du neben fortgeschrittenen Machine Learning Kenntnissen auch eine fundierte Kenntnis über Wahrscheinlichkeitstheorie.

Das Prinzip der Diffusionsmodelle funktioniert auch bei anderen Datenarten als Bildern. Ein großer Punkt ist dabei Audio und Video: Durch Diffusionsmodelle können zum Beispiel aus Textprompts kurze Clips wie dieser Teddy

Video: Teddy malt einen Teddy (**Quelle [6]**)

oder auch Sprache generiert werden.

Audioclip: „This audio is generated by a text-to-speech model for Emma Watson.” (**Quelle [7]**)

In diesem Video hast du die Arbeitsweise von Diffusionsmodellen kennengelernt. Du hast gesehen, wie Modelle mithilfe von Rauschen trainiert werden können und kannst den Prozess anhand von Beispielen veranschaulichen.

Quellen

Quelle [1] Sohl-Dickstein, J.; Weiss, E.; Maheswaranathan, N. & Ganguli, S. (2015). Deep Unsupervised Learning using Nonequilibrium Thermodynamics. *Proceedings of the 32nd International Conference on Machine Learning*, *37*, 2256–2265.

Quelle [2] Liu, Z.; Luo, P.; Wang, X. & Tang, X. (2015). Deep Learning Face Attributes in the Wild. *Proceedings of the IEEE International Conference on Computer Vision,* 3730-3738

Quelle [3] Ronneberger, O.; Fischer, P. & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, 234-241 https://doi.org/10.1007/978-3-319-24574-4_28

Quelle [4] Introducing Hugging Face's new library for diffusion models. Zugriff am 26.03.2024, https://colab.research.google.com/github/huggingface/notebooks/blob/main/diffusers/diffusers_intro.ipynb

Quelle [5] okanakdeniz. Zugriff am 26.03.2024, https://de.vecteezy.com/video/5171588-abstract-tinte-spread-hintergrund

Quelle [6] Meta AI. A teddy bear painting a portrait. Zugriff am 26.03.2024, https://makeavideo.studio/

Quelle [7] Kim, S.; Kim, H. & Yoon, S. (2022) Guided-TTS 2: A Diffusion Model for High-quality Adaptive Text-to-Speech with Untranscribed Data https://arxiv.org/abs/2205.15370 https://ksw0306.github.io/guided-tts2-demo/

Weiterführendes Material

https://colab.research.google.com/github/huggingface/notebooks/blob/main/diffusers/diffusers_intro.ipynb

https://huggingface.co/learn/diffusion-course

https://adityas03.medium.com/stable-diffusion-for-dummies-7a785e0edd9d

Disclaimer