TECHNICAL APPLICATIONS AND DATA MANAGEMENT. SS 2023.

VORLESUNG 10 UND 11

06.06.2023

MÜNCHEN

STUDIENGANG SUSTAINABILITY MANAGEMENT.

AGENDA

1. Generative AI
2. Beispiele Generative AI
3. Detaillierung Chat GPT

GEPLANTE ROADMAP DER VORLESUNG.

ROADMAP	WAS HABEN WIR VOR?
Vorlesung 1	Übersicht und Einführung
Vorlesung 2	Einführung Data Science und Data Science Workflow, Detaillierung Data Engineering
Vorlesung 3 und Vorlesung 4	Deskriptive und explorative Datenanalyse und Vertiefung anhand Case Study
	Vertiefung Datenanalyse anhand Case Study
Vorlesung 5	Aufgabenstellung Data Science, Übersicht und Einführung Machine Learning, unüberwachtes Lernen
Vorlesung 6 und Vorlesung 7	Überwachtes Lernen
	Vertiefung überwachtes Lernen anhand Case Study
Vorlesung 8 und Vorlesung 9	Neuronale Netze und Convolutional Neural Networks (CNN)
	Vertiefung CNN anhand Case Study, Aufgabenstellung AI
Vorlesung 10	Rekurrente Neuronale Netze
Vorlesung 11	Generative AI
Vorlesung 12	Ausblick
Vorlesung 13	„Fragestunde“

Folien der bisherigen Vorlesung verfügbar unter Link

Angebot für Sprechstunden zwischen Vorlesungen (nach Bedarf)

1. GENERATIVE AI

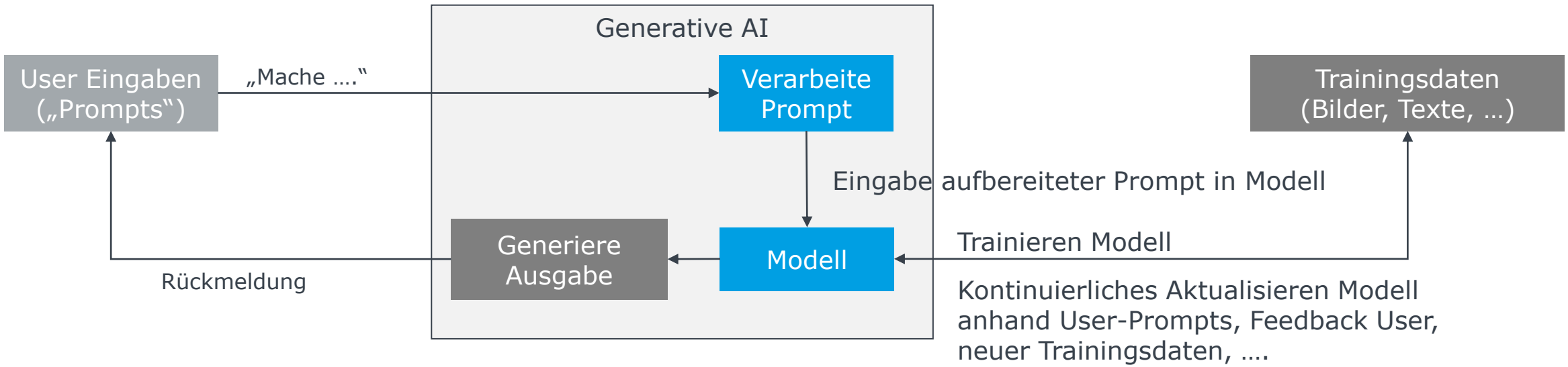
1.1 ÜBERSICHT GENERATIVE AI

WAS HABEN WIR BISHER GEMACHT?

- Data Science
- Unsupervised Learning auf tabellarischen Daten
- Supervised Learning auf tabellarische Daten und Bildern

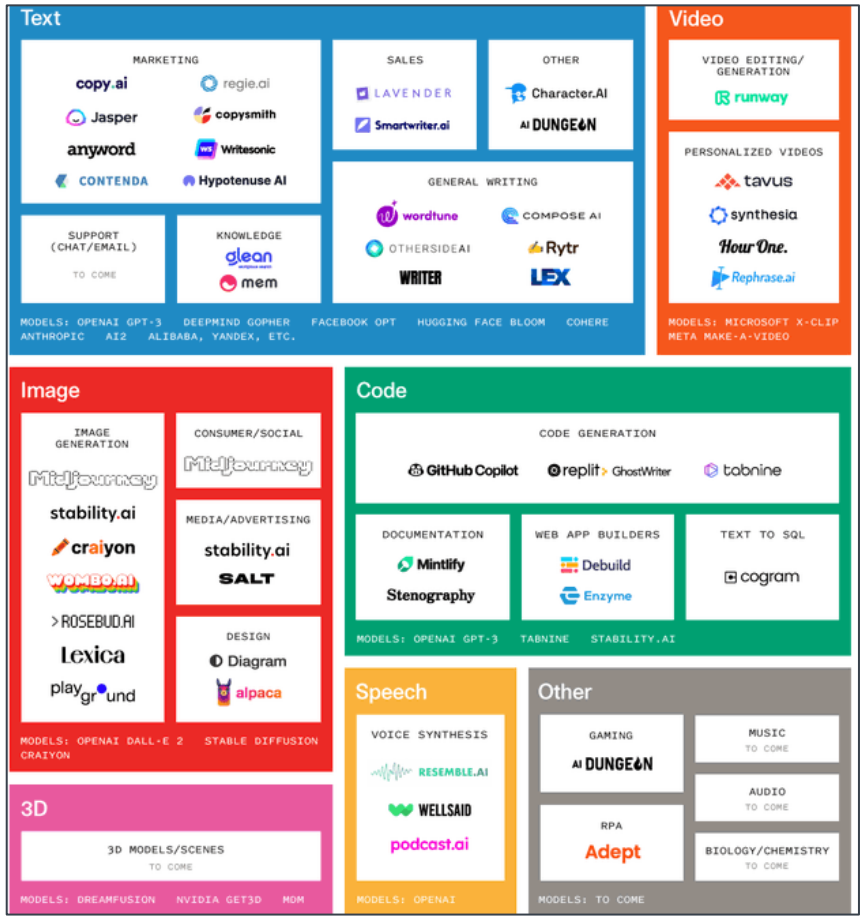
Bei den bisher betrachteten Verfahren erhalten wir durch den Einsatz von Machine Learning einen Informationsmehrwert.
Wir lernen neue Zusammenhänge, erstellen aber keine neuen Daten/ Content bzw. ändern die vorhandenen Daten nur leicht.

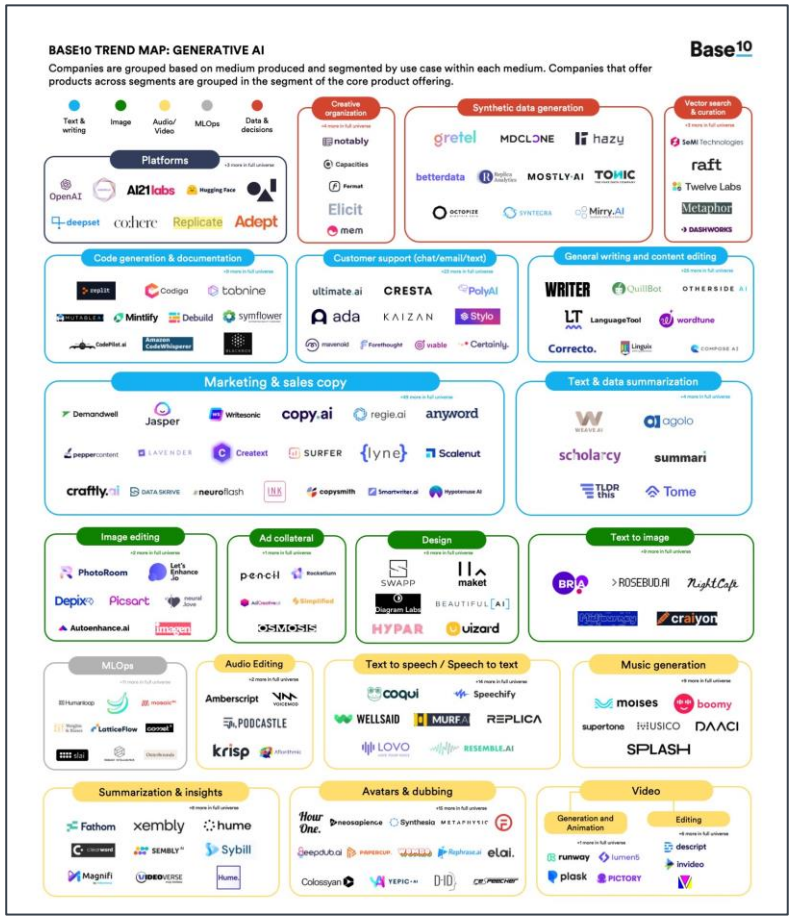
ÜBERSICHT GENERATIVE AI.

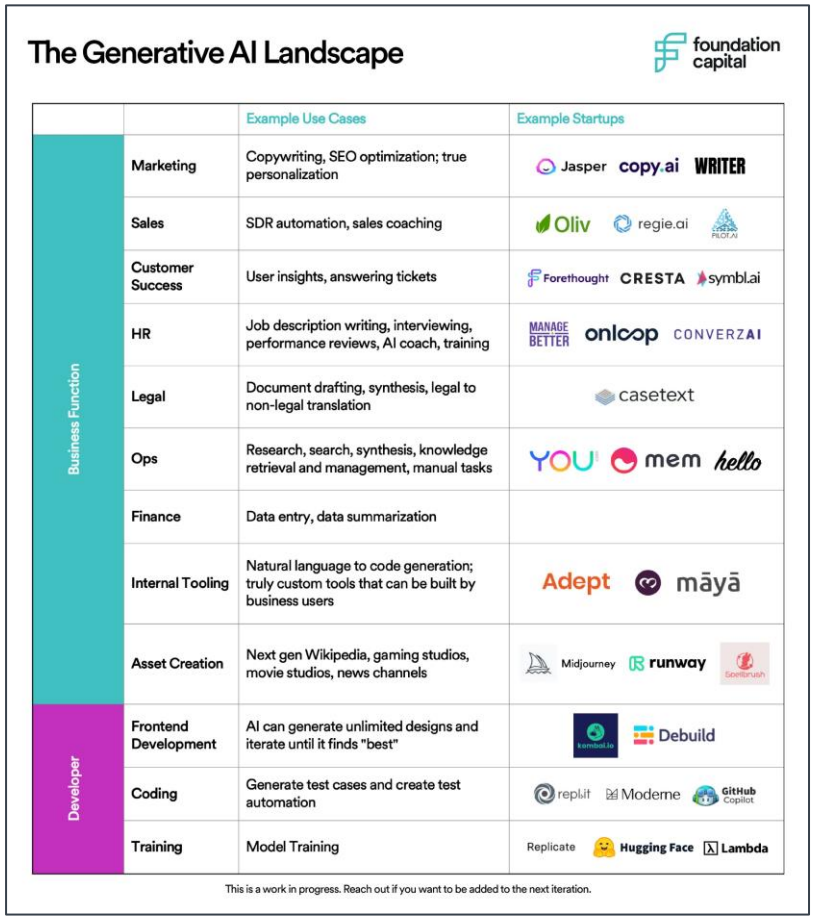


Ziel/ Anspruch Generative AI: Schaffen Mehrwert oder Entlasten Anwender durch Erstellen hochwertiger Ausgaben in kurzer Zeit

GENERATIVE AI IST EIN STARK WACHSENDES THEMA MIT VIELEN ANWENDUNGSFÄLLEN.







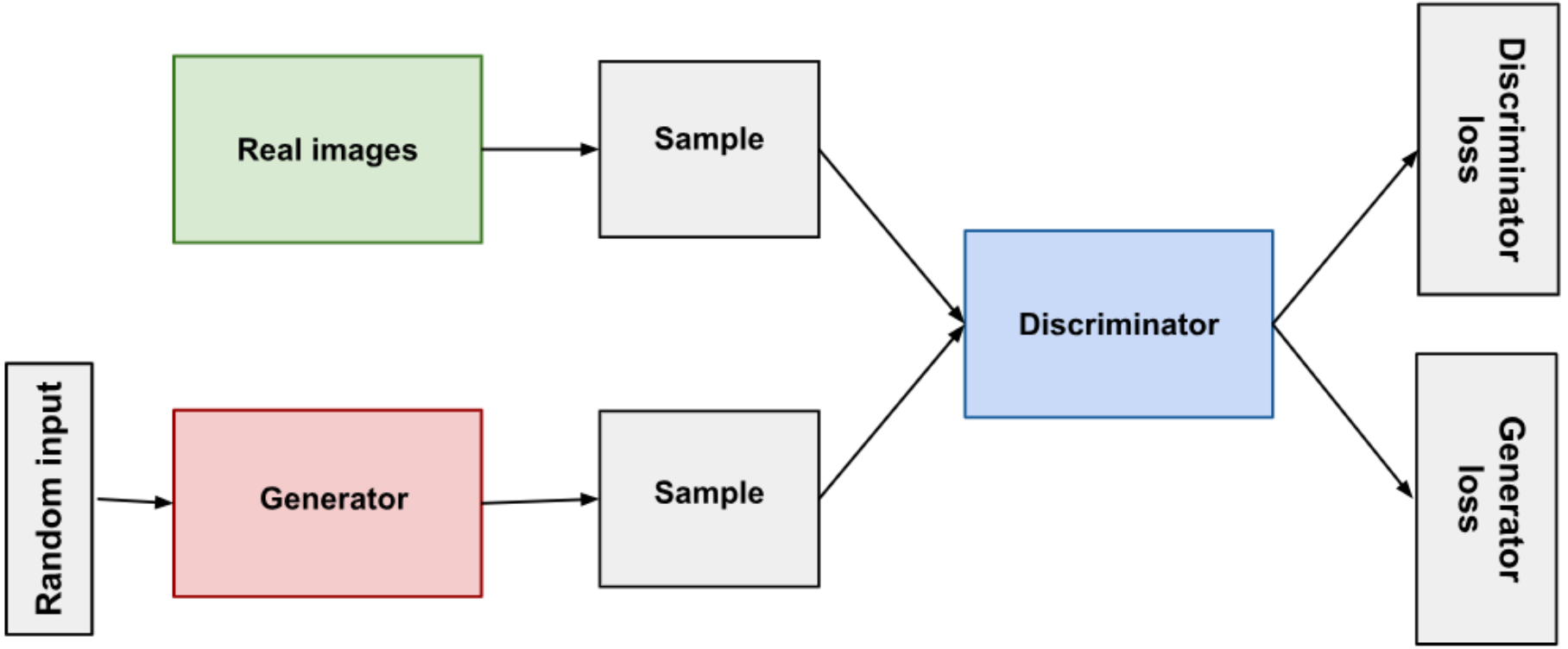
Quellen (von links nach recht): Link, Link, Link.

1.2 GENERATIVE AI FÜR BILDER

ÜBERSICHT BEKANNTE VERFAHREN (AUSZUG).

- 2014: Generative Adversarial Networks
- 2015: Neural Style Transfer
- 2021: Open AI: Dall-E
- 2022: Midjourney, Stable Diffusion

GENERAL ADVERSIAL NETWORKS: ÜBERSICHT.



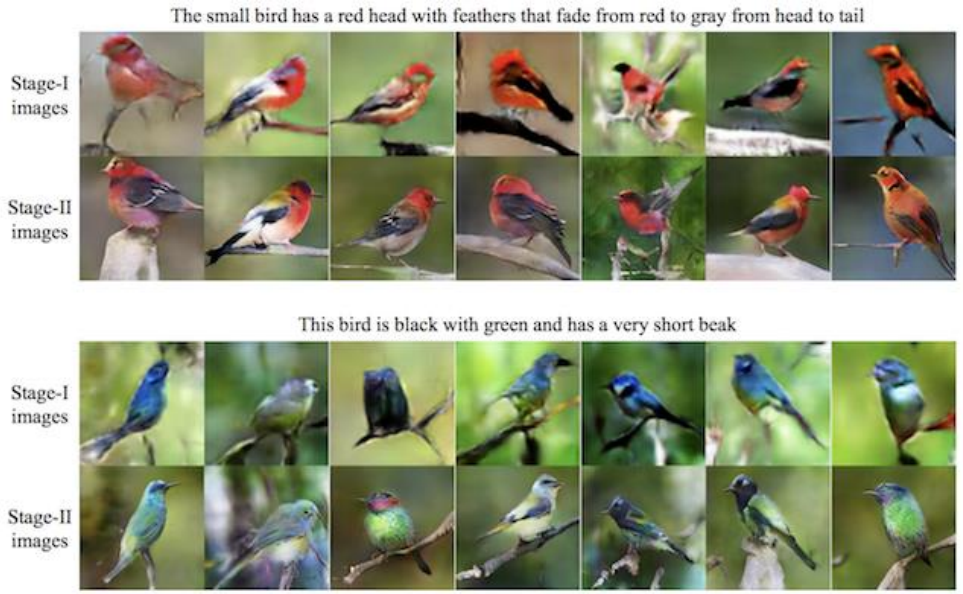
Paper: Goodfellow, Ian et al. (2014). Generative Adversarial Nets.Proceedings of the International Conference on Neural Information Processing Systems (NIPS 2014)

Vorgehensweise: Generator und Discriminator sind jeweils neuronale Netzwerke.
Generator erstellt Bilder, die Input für den Discriminator sind. Dabei lernt er, möglichst genaue, plausible Bilder zu erstellen.
Discriminator lernt, zwischen realen Bildern und den „falschen" vom Generator zu unterscheiden. Die Rückmeldung bei der Erkennung nutzen beide Netzwerke zur Verbesserung Erkennung und Generierung per Backpropagation.

Quelle: Link

GENERAL ADVERSIAL NETWORKS: ANWENDUNGSBEISPIELE.

Text-to-Image



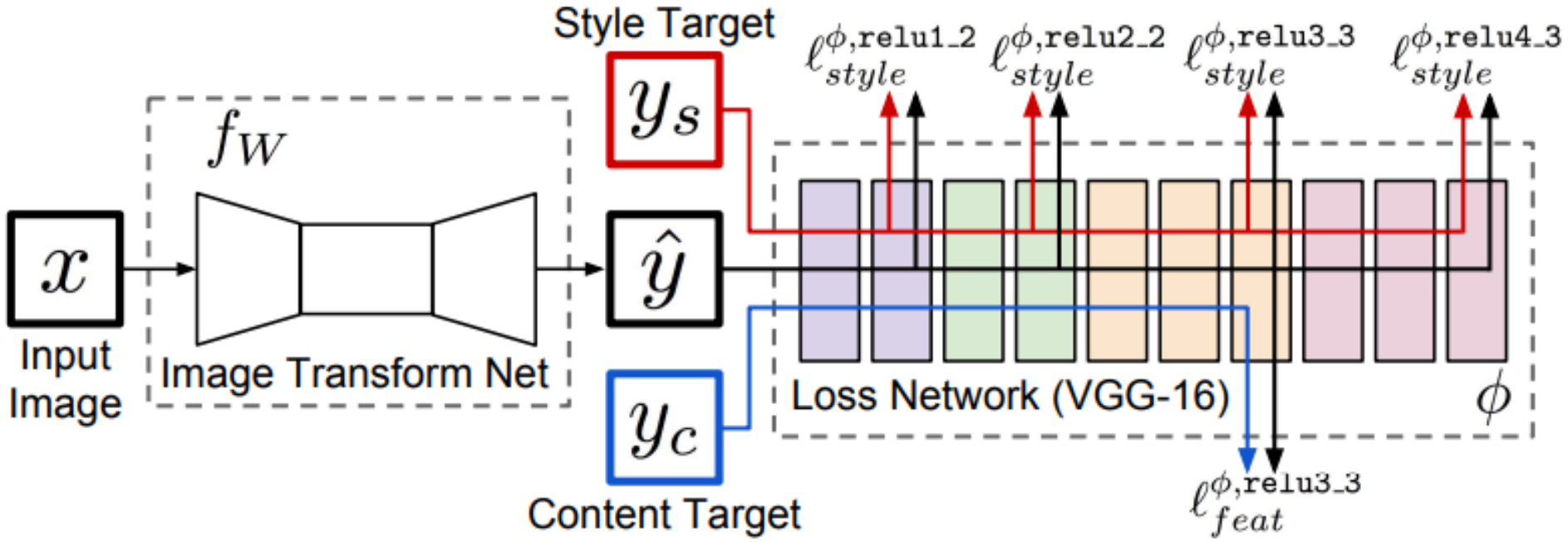
Han Zhang et al., „StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks", 2016. Link

Modeindustrie: Virtual try on

Bergmann: „Generative Models in e-Commerce", 2020. Vortrag an LMU, Link

Weitere Anwendungsbeispiele: Link, Link
- Erstellen/ Vorhersage nächster Frames für Videos
- Generieren von 3D-Objekten aus 2D Bildern

NEURAL STYLE TRANSFER: ÜBERSICHT.



Papers: Gatys, Leon A et al. "A Neural Algorithm of Artistic Style", arXiv:1508.06576, 2015, Johnson et al., „Perceptual Losses for Real-Time Style Transfer and Super-Resolution", 2016. Link

Code-Notebook: Hier, Hier

Vorgehensweise: „We train an image transformation network to transform input images into output images. We use a loss network pretrained for image classification to define perceptual loss functions that measure perceptual differences in content and style between images. The loss network remains fixed during the training process" (Modifikation Transfer Learning)

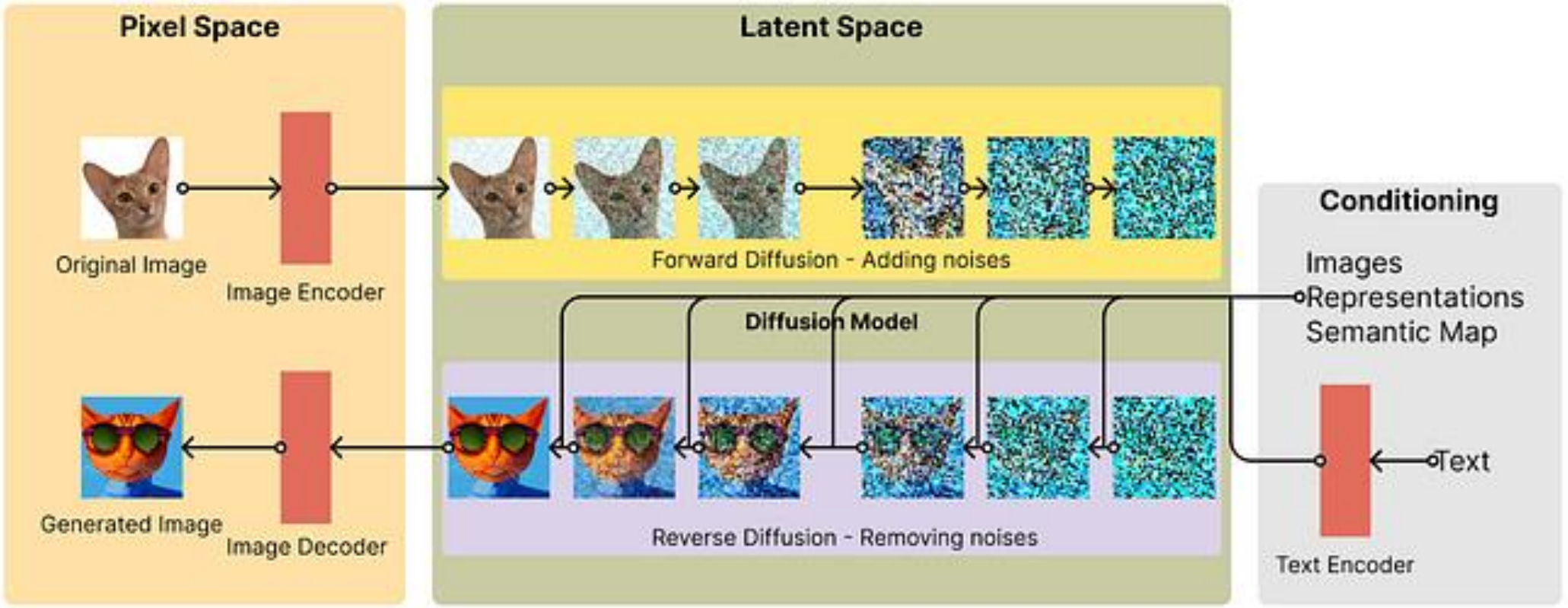
Quelle: Johnson et al., „Perceptual Losses for Real-Time Style Transfer and Super-Resolution", 2016. Link

NEURAL STYLE TRANSFER: ANWENDUNGSBEISPIEL.

Modifikation Transfer Learning: Nehme den Stil eines bekannten Bildes und zeichne ein anderes Bild in diesem Stil

Quellen:
1 Van Gogh, „Starry Night Over the Rhone"
2 unbekannt
3 Generated Picture via DeepArt.io

STABLE DIFFUSION: ÜBERSICHT.



Paper: Rombach et al, „High-Resolution Image Synthesis with Latent Diffusion Models", Link, 2021.
Source Code: Github
Online aufrufbar unter: StableDiffusionWeb, StableBoost.AI

STABLE DIFFUSION. BEISPIELHAFTE BILDER.

Prompt: "a photograph of an astronaut riding a horse"

Prompt: „an ultra detailed Darth vader standing portrait in front of a highly detailed landscape of a big and structured city inpired by Star Wars, portal to outer space, 4k digital art, octane render, trending on artstation, digital art, 8k, disney render, disney"

Quelle: Bild 1: Link, Bild 2: Link

AUSBLICK: ADOBE PHOTOSHOP GENERATIVE FILL.

Link

1.3 GENERATIVE AI FÜR TEXTE

ÜBERSICHT BEKANNTE VERFAHREN (AUSZUG).

- 1972 Rekurrente Neuronale Netzwerke (RNN)[1]: neuronale Netze mit Rückkopplung. Sehr mächtig, aber ressourcenintensiv.
- 1997 LSTM[2] basiert auf RNN und löste zuerst Vanishing Gradient[3]-Problem bei zu verarbeitenden längeren Sequenzen. Milliardenfach eingesetzt, bspw. für Spracherkennung (Alexa, Siri, …). Sehr ressourcenintensives Training.
- 2015 Attention[4] löst Problem längerer Sequenzen durch individuelles Gewichten einzelner In- und Outputs. Kein RNN!
- 2017 Transformer[5] basiert auf Attention, bietet flexibleres, schnelleres Lernen als LSTM aufgrund Parallelisieren. Kein RNN!
- 2019 BERT[6]: Transfer Learning, basierend auf Transformer.
- Generative Pre-trained Transformer (GPT):
  - 2018: GPT-1[7] vortrainiertes Modell basierend auf Transformer, das gegeben Input neue Wörter generiert.
  - 2019: GPT2[8]
  - 2020: GPT3[9]
  - 2022 InstructGPT[10]/ ChatGPT: Fine-Tuning via Supervised Learning per Chats mit Menschen sowie Reinforcement Learning
  - 2023: Chat GPT4 Einsatz multimodaler Daten

1 Amari, Shun-Ichi, "Learning patterns and pattern sequences by self-organizing nets of threshold elements", 1972.
2 Hochreiter, Schmidhuber: "Long Short-Term Memory ", 1997. Link
3 Vanishing Gradient: beim Trainieren von Netzen mit vielen Schichten und Einsatz Expontentialfunktion als Aktivierungsfunktion kann bei Backpropagation Fall entstehen, daß Gradienten sehr, sehr klein werden und damit die Gewichte und Bias des Modells, v.a. der ersten Schichten, nicht mehr geändert/ trainiert werden. Vgl. Hochreiter: „Untersuchungen zu dynamischen neuronalen Netzen", 1991.
4 Vaswani et al.: "Attention is all you need ", 2017. Link
5 Bahdanau et al.: "Neural Machine Translation by Jointly Learning to Align and Translate", 2015. Link. Basierend auf Schmidhuber, "Learning to control fast-weight memories.", 1992
6 Devlin et al.: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", 2019. Link
7 Radford et al, „Improving Language Understanding by Generative Pre-Training", 2018, Link
8 Radford et al., "Language Models are Unsupervised Multitask Learners", 2019, Link
9 Brown et al., "Language Models are Few-Shot Learners", 2020, Link
10 Ouyang, Long, et al. "Training language models to follow instructions with human feedback", 2022, Link

KURZE GESCHICHTE VON CHAT GPT

Modell	Veröffentlicht	Trainingsdaten	Anzahl Parameter	Max. Sequenzlänge
GPT-1	Juni 2018	BooksCorpus Datensatz (~11000 Bücher)	117 Millionen	1024
GPT-2	February 2019	BooksCorpus, CommonCrawl (WebArchiv: >3 Mrd. Websites mit ~417TB Daten), WebText (> 40GB von Reddit)	1.5 Milliarden	2048
GPT-3	June 2020	BookCorpus, Common Crawl, Wikipedia, Bücher, Artikel, ...	175 Milliarden	4096
GPT-4	March 2023	Unbekannt	>1 Billion	Unbekannt

Aktuell wird sehr viel geforscht, die Parameteranzahl zu reduzieren bei gleicher/ besserer Modellgenauigkeit – analog der Entwicklung der Transfer learning Modelle für Bilder (bspw. von VGG16 zu Inception). Das ist aber außerhalb des Fokus dieser Vorlesung…

Quellen: Link, Link

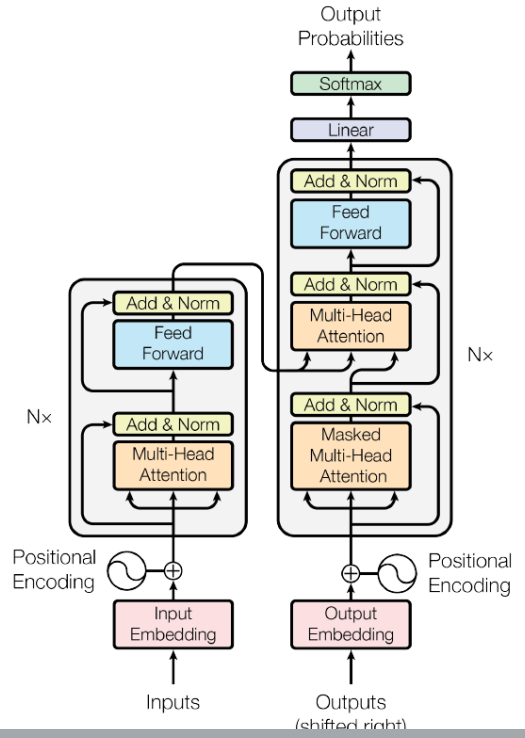
2. DETAILLIERUNG FALLBEISPIEL CHAT GPT

WIR SCHAUEN UNS CHAT GPT IN FOLGENDEN EINZELNEN SCHRITTEN AN.

1. Übersicht ChatGPT
2. Welche Daten werden verwendet?
3. Wie werden die Daten aufbereitet?
4. Wie erfolgt das Lernen?

(GROBE) ÜBERSICHT CHAT GPT

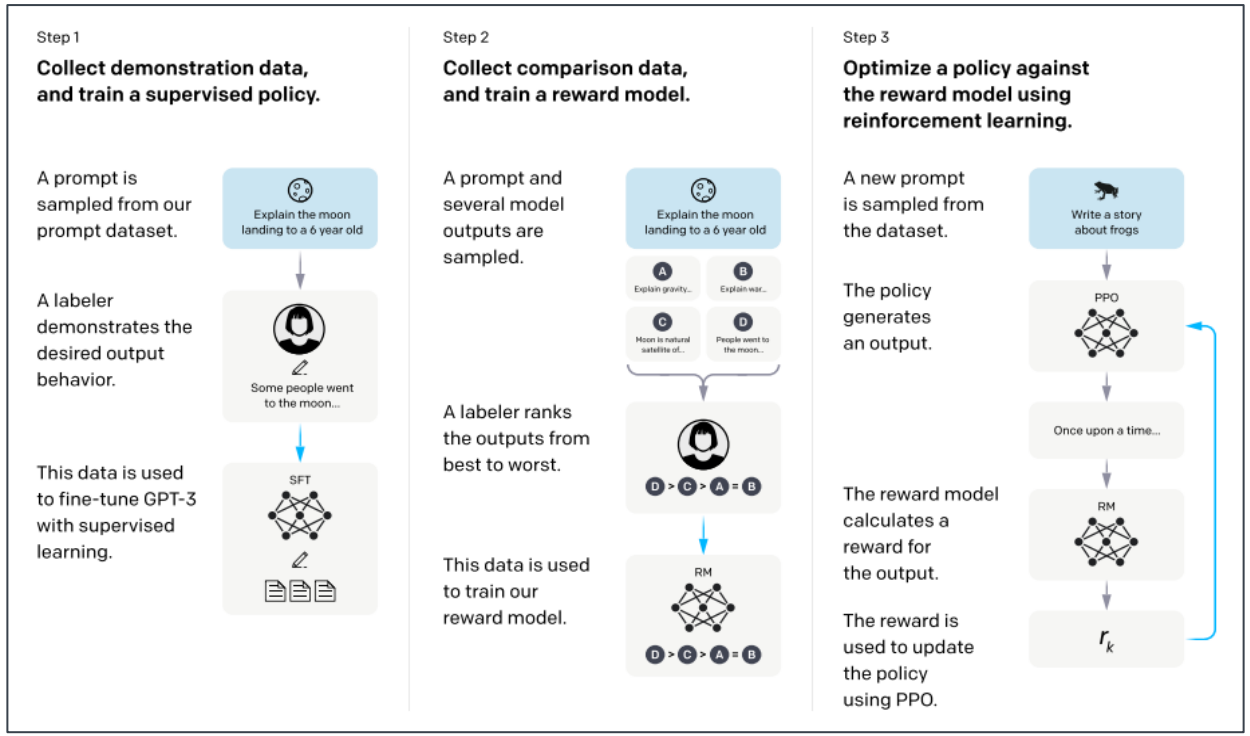
Schritt 1: Trainieren Transfomer Modell



Lernt „Large Language Modell", das für eine Inputsequenz die wahrscheinlichsten Folgewörter vorhersagt

+

Schritt 2: Fine-Tuning Modell



Modell lernt repräsentative Anwendungsfälle von/ mit Menschen und Belohnungsfunktion. Diese Funktion ermöglicht Modell, selbst zu erkennen, ob Chat „gut" läuft und so Optimierung Chatverlauf.

Quellen:
linkes Bild: Vaswani et al.: "Attention is all you need", (2017), Link.
Rechtes Bild: Ouyang, Long, et al. "Training language models to follow instructions with human feedback", 2022, Link

2.1 SEQUENTIELLE DATEN

WELCHE DATENTYPEN HABEN WIR BIS JETZT FÜR MACHINE LEARNING EINGESETZT?

Tabellarische Datensätze:

	PassengerClass	survived	name	sex	age	SiblingSpousesPresent	ParentsChildrenPresent	ticket	fare	cabin	embarked	boat	body	HomeDestination
1289	3	False	Wiklund, Mr. Karl Johan	male	21.0	1	0	3101266	6.4958	0	Southampton	0	0	unknown
1290	3	True	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	0	Southampton	0	0	unknown
1291	3	False	Willer, Mr. Aaron ("Abi Weller")	male	NaN	0	0	3410	8.7125	0	Southampton	0	0	unknown
1292	3	False	Willey, Mr. Edward	male	NaN	0	0	S.O./P.P. 751	7.5500	0	Southampton	0	0	unknown
1293	3	False	Williams, Mr. Howard Hugh "Harry"	male	NaN	0	0	A/5 2466	8.0500	0	Southampton	0	0	unknown
1294	3	False	Williams, Mr. Leslie	male	28.5	0	0	54636	16.1000	0	Southampton	0	14	unknown
1295	3	False	Windelov, Mr. Einar	male	21.0	0	0	SOTON/OQ 3101317	7.2500	0	Southampton	0	0	unknown
1296	3	False	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	0	Southampton	0	131	unknown
1297	3	False	Wiseman, Mr. Phillippe	male	NaN	0	0	A/4. 34244	7.2500	0	Southampton	0	0	unknown
1298	3	False	Wittevrongel, Mr. Camille	male	36.0	0	0	345771	9.5000	0	Southampton	0	0	unknown
1299	3	False	Yasbeck, Mr. Antoni	male	27.0	1	0	2659	14.4542	0	Cherbourg	C	0	unknown
1300	3	True	Yasbeck, Mrs. Antoni (Selini Alexander)	female	15.0	1	0	2659	14.4542	0	Cherbourg	0	0	unknown
1301	3	False	Youseff, Mr. Gerious	male	45.5	0	0	2628	7.2250	0	Cherbourg	0	312	unknown
1302	3	False	Yousif, Mr. Wazli	male	NaN	0	0	2647	7.2250	0	Cherbourg	0	0	unknown
1303	3	False	Youssef, Mr. Gerious	male	NaN	0	0	2627	14.4583	0	Cherbourg	0	0	unknown
1304	3	False	Zabour, Miss. Hileni	female	14.5	1	0	2665	14.4542	0	Cherbourg	0	328	unknown
1305	3	False	Zabour, Miss. Thamine	female	NaN	1	0	2665	14.4542	0	Cherbourg	0	0	unknown
1306	3	False	Zakarian, Mr. Mapriededer	male	26.5	0	0	2656	7.2250	0	Cherbourg	0	304	unknown
1307	3	False	Zakarian, Mr. Ortin	male	27.0	0	0	2670	7.2250	0	Cherbourg	0	0	unknown
1308	3	False	Zimmerman, Mr. Leo	male	29.0	0	0	315082	7.8750	0	Southampton	0	0	unknown

Einsatz von unsupervised und supervised Verfahren für das Lernen von Features aus den Daten

Bilder:

Lernen Features eines Bildes per CNN, dann Einsetzen in Supervised Learning Verfahren.

Hierbei handelt es sich um statische, sich über den zeitlichen Verlauf nicht ändernde, Daten

ÜBERSICHT SEQUENTIELLE, ZEITABHÄNGIGE DATEN.

- Zeitreihen: Aktienkurse, Messungen, Temperaturkurven, …
- Video: Folge von einzelnen Bildern (Frames per Second)
- Sprache: Folge von einzelnen gesprochenen Wörtern
- **Texte: Folge von einzelnen geschriebenen Wörtern**

Bei sequentiellen Daten haben vorherige Daten Einfluß auf die aktuellen und nachfolgenden Daten

2.2 NATURAL LANGUAGE PROCESSING UND LARGE LANGUAGE MODELL

SPRACHVERARBEITUNG/ NATURAL LANGUAGE PROCESSING HAT VIELE HERAUSFORDERUNGEN.

Sprachen unterscheiden sich deutlich:

- Verschiedene Formen eines Wortes:
  - Deklinationen, Kasus oder Konjugation (bspw. Englisch vs. Russisch vs. Finnisch).
  - Zeiten (viele Zeiten wie im Englischen oder Altgriechischen vs. keine Zeiten wie im Chinesischen).
- Satzstellung: Stellung Verb im Satz, Adjektiv vor oder nach Nomen, …
- Akzente (andere Betonungen) und Dialekte (andere Wörter).
- …

Texte unterscheiden sich deutlich:

- Schrift: lateinisches/kyrillisch/griechisches Buchstabenalphabet vs. chinesische/japanische/koreanische Zeichenschrift [Kanji]
- Wortabstand oder kein Wortabstand
- Schriftrichung: links, rechts (bspw. Hebräisch oder Arabisch) oder nach unten (bspw. chinesisch)

VERARBEITEN UND ERFASSEN VON TEXTEN PER LANGUAGE MODEL.

Input

I WOULD LIKE A VODKA WITH ? ? ?

Wörterbuch mit Wahrscheinlichkeitsverteilung

Word	Wahrscheinlichkeit
Soda	20%
Pure	10%
…	…
….	….
Car	1%

Gegeben ein Input von mehreren Wörtern, möchten wir wissen, was wahrscheinliche Folgewörter sind.
Da eine Sprache sehr viele Wörter hat, setzen wir Natural Language Processing ein, um die Vielzahl der Wörter zu beherrschen.

DETAILLIERUNG BEISPIELHAFTE NLP-TECHNIKEN:

- **Sentence breaking**: Aufspalten Text in einzelne Sätze, bspw. anhand Punkt oder Komma.
- **Lemmatization**: Vereinheitlichen ähnlicher Wörter/ Reduktion Varianz mit Wörterbuch. {Eats, eating, ate, eaten} → eat {ride, taking a ride} → ride
- **Stemming**: Vereinheitlichen eines Wortes/ Reduktion Varianten durch Abschneiden Endungen. {played, plays, playing, …} → play
- **Stop Word Removal**: Entfernen Füllwörter (Pronomen, Artikel, Präpositionen, Fragewörter,….) [i, my, her, hers, has, had, hadnt, no, nor, not, only, doesnt, …]
- **Text/Word Embedding**: Umwandeln einzelner Buchstaben/ ganzer Wörter in eindeutige Zahlen. (Rechner können ja nur mit Zahlen arbeiten….)

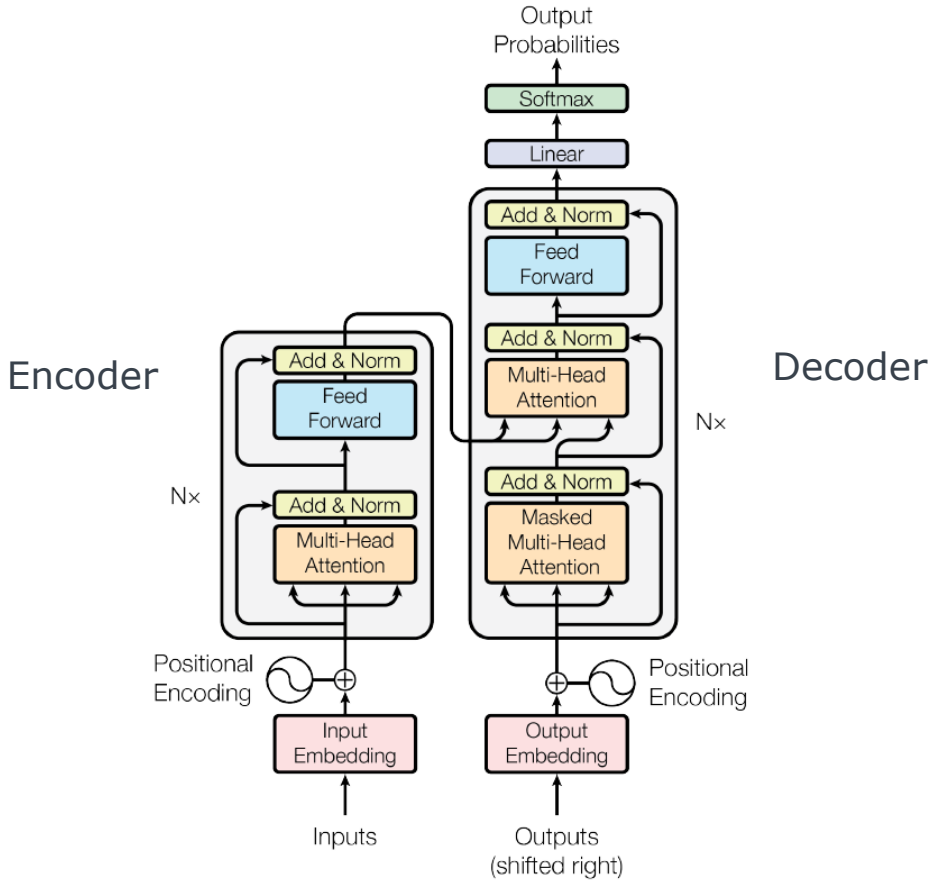
```
          1 → <start>
Bus       → 5
kommt     → 144
heute     → 114
          2 → <end>
```

NLP ermöglicht das effiziente Verarbeiten von Texten für Machine Learning

Quelle: https://www.nltk.org/

2.3 DETAILLIERUNG TRANSFORMER

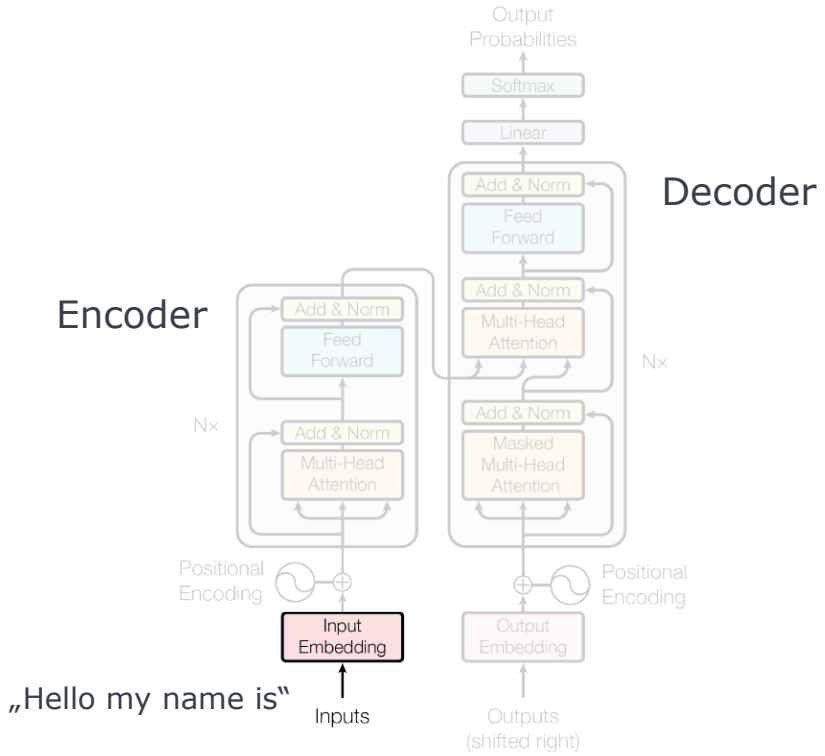
ÜBERSICHT TRANSFORMER.



Aufgabe Encoder: speichere Eingabe in sequentieller Form inkl. Attention (welche Input-Wörter hängen mit welchen zusammen?)
Aufgabe Decoder: nutze diese Form für Fokus auf passende Input-Wörter und erstelle Wahrscheinlichkeitsverteilung über gesamtes Vokabular für möglichst genaue Vorhersage des nächsten Ausgabeworts/ nächsten Ausgabewörter.

Quellen: Vaswani et al.: "Attention is all you need", (2017), Link.

TRANSFORMER IM DETAIL. ENCODER – INPUT EMBEDDING.



Input Embedding:

- Eingabetext wird aufgeteilt in einzelne Wörter („Tokens"). Kann auf mehreren Rechnern parallel erfolgen.
  „Hello my name is" → [Hello] [my] [name] [is]

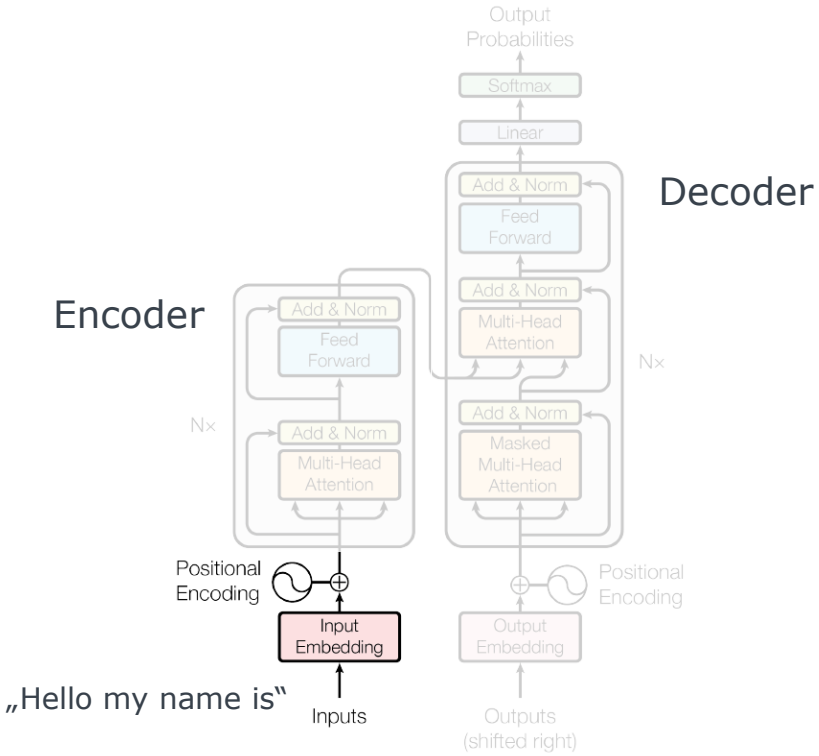
- Hinzufügen Tokens zu einem Vokabular mit eindeutiger Nummer.
  [Hello] [my] [name] [is] -> [3 721 68 1337]

- Jeder Token-Eintrag hat eine Verbindung zu dem (bisher) gelernten Model und dessen n-dimensionalen Vektor. Zusätzlich werden ähnliche Wörter gruppiert.
  [3 721 68 1337] -> [[0.123, -5.234, ...], [...], [...], [...], [...]]

Computer verstehen keine Wörter, sondern Zahlen. Deshalb bauen wir eine Art Wörterbuch mit eindeutiger Nummer je Wort. Um die Qualität des Wörterbuchs zu verbessern, gruppieren wir von der Bedeutung ähnliche Wörter.

Quellen: Link, Link, Link

TRANSFORMER IM DETAIL. ENCODER – POSITIONAL ENCODING



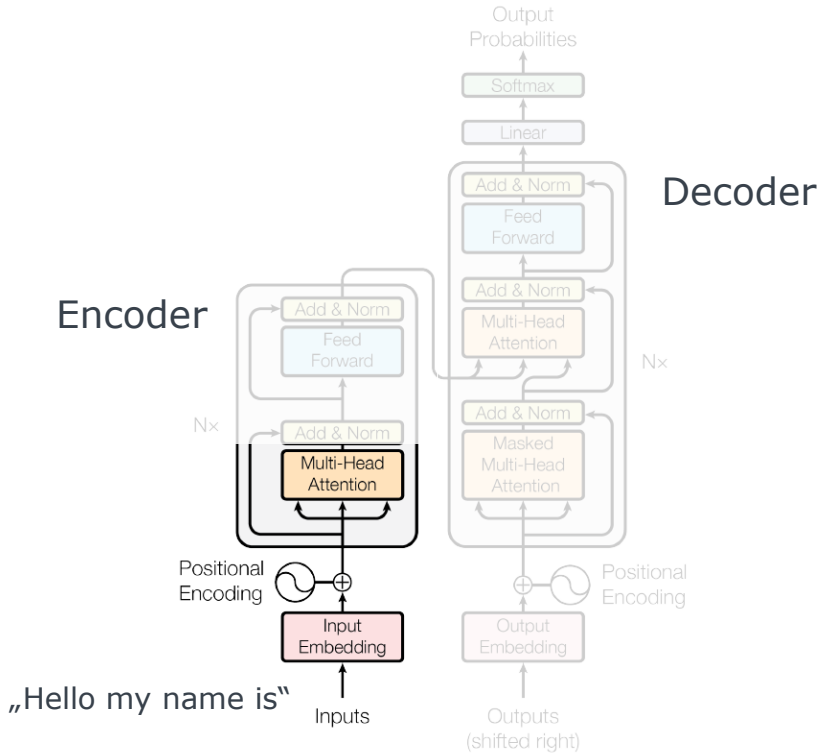
Positional Encoding:

- Speichere Position, die die einzelnen Wörter im Input-Satz hatten.
- Addiere die Position zu dem Embedding aus dem vorigen Schritt.

Durch Positional Encoding speichern wir die Reihenfolge der Wörter und können so bspw. „A liebt B“ von „B liebt A“ unterscheiden.

Quellen: Link, Link, Link

TRANSFORMER IM DETAIL.
ENCODER – MULTI-HEAD ATTENTION.



Self-headed attention:

- Modell soll die Verbindungen zwischen den einzelnen Input-Wörtern untereinander lernen (bspw. „wie" mit „geht" und „dir").

	Hello	my	name	is
Hello	0,75	0,15	0,05	0,05
my	0,1	0,6	0,2	0,1
name	0,05	0,2	0,65	0,1
is	0,2	0,1	0,1	0,6

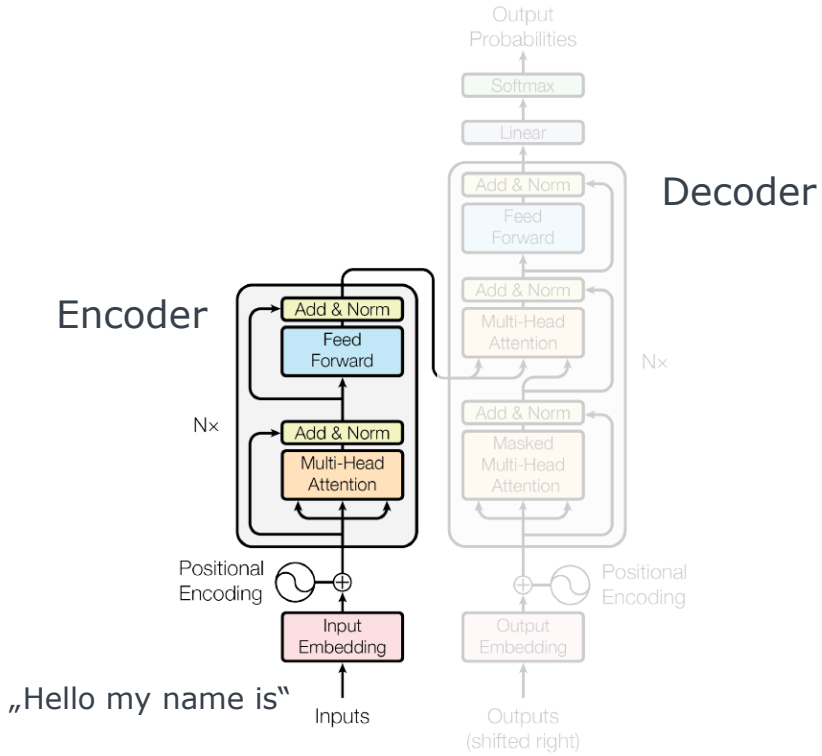
Multi-headed attention:

- vermeidet, daß self-headed attention den eigenen Input stärker gewichtet als die restlichen Wörter.
- Gewichten der Ergebnisse einzelner self-headed Attentions und Kombination zu finalem Vektor je Token. Dieser Vektor sagt, wie stark das Token auf die anderen Wörter schauen sollte.

Multi-headed Attenion erlaubt das Finden und Lernen von Korrelationen zwischen den einzelnen Bestandteilen eines Satzes.

Quellen: Link, Link, Link

TRANSFORMER IM DETAIL. ENCODER – RESIDUAL, NORMALISIERUNG UND FEED FORWARD



Residual:

- Der vorherige Ausgabevektor wird zum Positions-Encoding addiert. Dies ermöglicht ein starkes Verbessern der Trainierbarkeit.

Normalisierung:

- Stabilisieren Netzwerk und deutliche Reduktion Trainingszeiten.

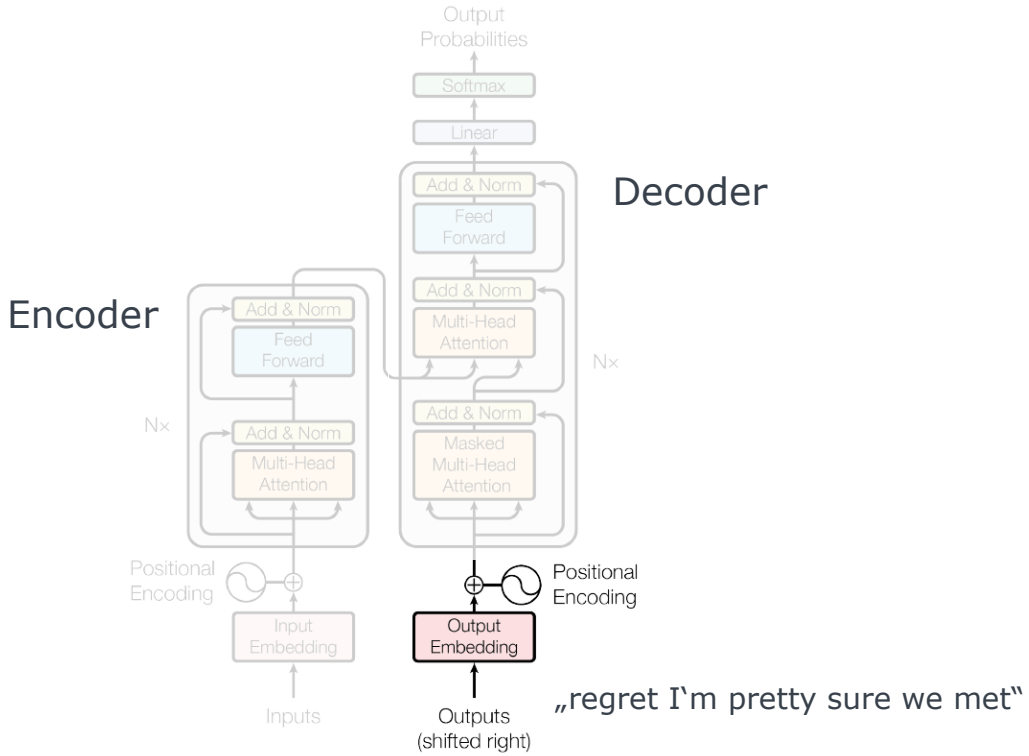
Feed forward:

- Einsatz eines einfachen, nicht-rekurrenten neuronalen Netzwerks für Verarbeitung Ergebnisse (analog bisherige Vorlesungen…).
- Speichern der Ergebnisse in einem Format für die spätere Berechnung Wahrscheinlichkeit (analog Bilderkennung!)

Einsatz dieser Techniken ermöglichte Verbesserung der Ergebnisse sowie Reduzierung Ressourceneinsatz.

Quellen: Link, Link, Link

TRANSFORMER IM DETAIL. DECODER – INPUT EMBEDDING UND POSITIONS ENCODING



Input Embedding:

- Gleiches Vorgehen für Output-Satz wie bei Input

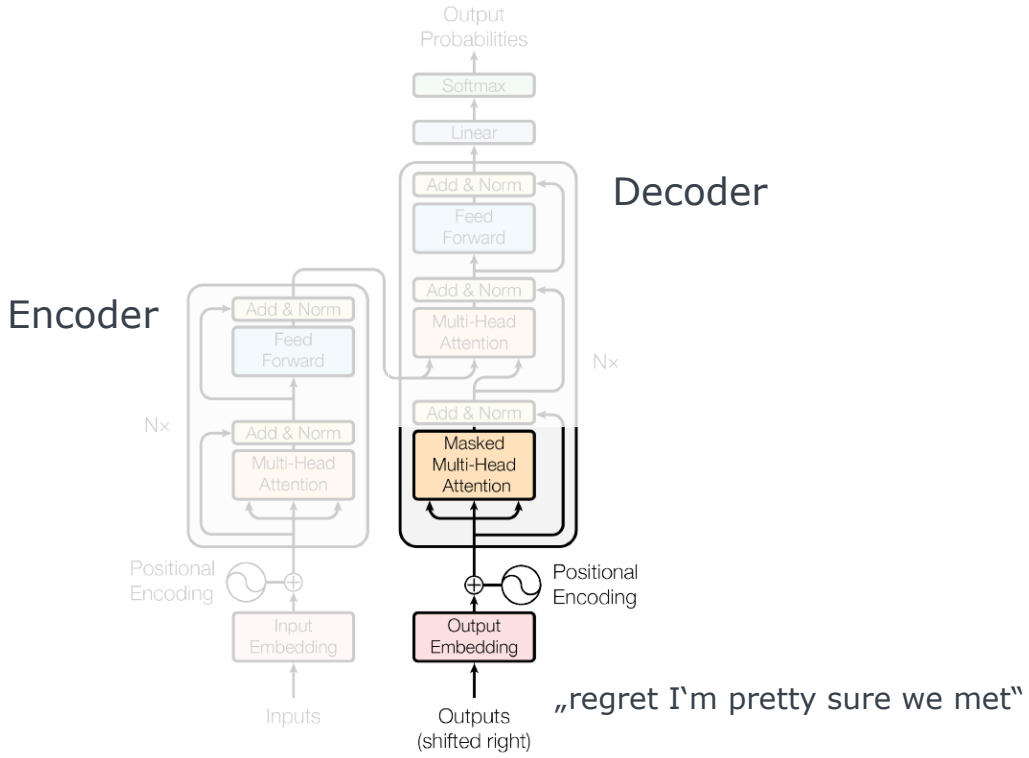
Positional Encoding:

- Analog zu Encoding.

Gleiches Vorgehen wie beim Encoding, nur für Decoding Satz. Dieser ist beim Trainieren bekannt, beim späteren Einsatz jedoch nicht.

Quellen: Link, Link, Link

TRANSFORMER IM DETAIL.
DECODER – MASKED MULTI-HEAD ATTENTION.

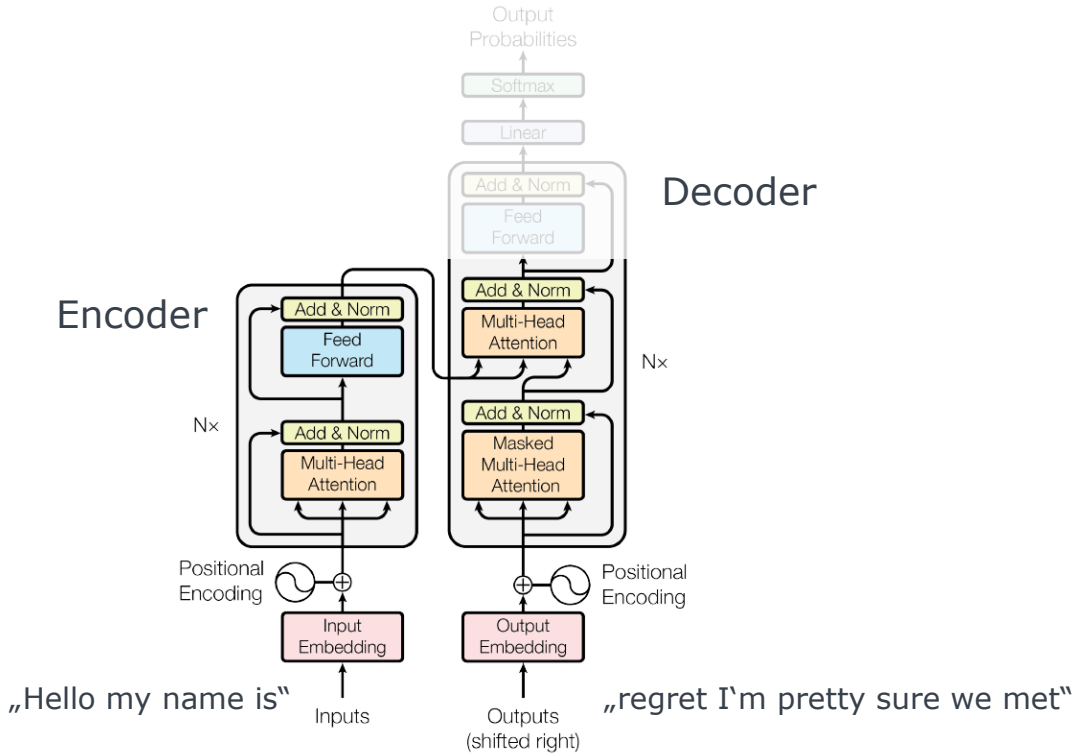


Masked Multi-head attention

- Ähnlicher Schritt wie Multi-Head Attention beim Encoding.
- Aber: beim Trainieren haben wir jeweils einen Satz für Input als auch für Output.
- Damit das Modell lernt und nicht einfach das nächste Wort "nachschaut", werden die restlichen Wörter des Vergleichssatzes ausmaskiert (auf 0 gesetzt), d.h. das Modell kann diese nicht nachsehen.
- Dadurch kann auch parallel trainiert werden und das Training somit beschleunigt werden.

Quellen: Link, Link, Link

TRANSFORMER IM DETAIL.
DECODER –MULTI-HEAD ATTENTION.



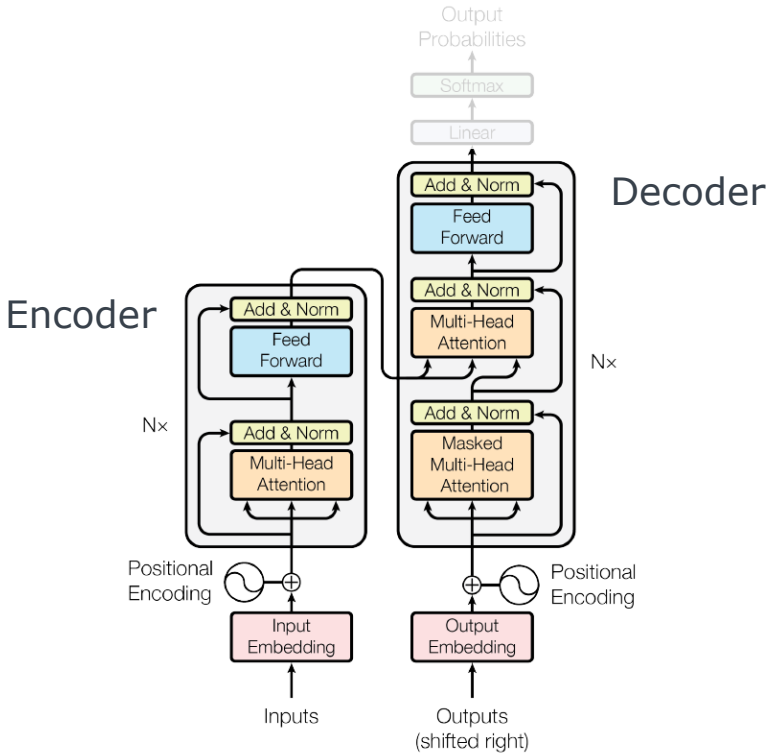
Multi-head attention (oder Encoder-Decoder Attention)

- In diesem Block kommen die Tokens vom Encoder und die maskierten Eingaben vom Decoder (inkl. Residual) zum ersten Mal zusammen.
- Dadurch kann das Modell für jeden Token vom Decoder lernen, welcher Input vom Encoder relevant ist.

Hier entsteht das Mapping von Input auf Output also bspw. ein Attention Vektor für jedes Wort

Quellen: Link, Link, Link

TRANSFORMER IM DETAIL. DECODER –FEED FORWARD.



Normalisierung:

- Stabilisieren Netzwerk und deutliche Reduktion Trainingszeiten.

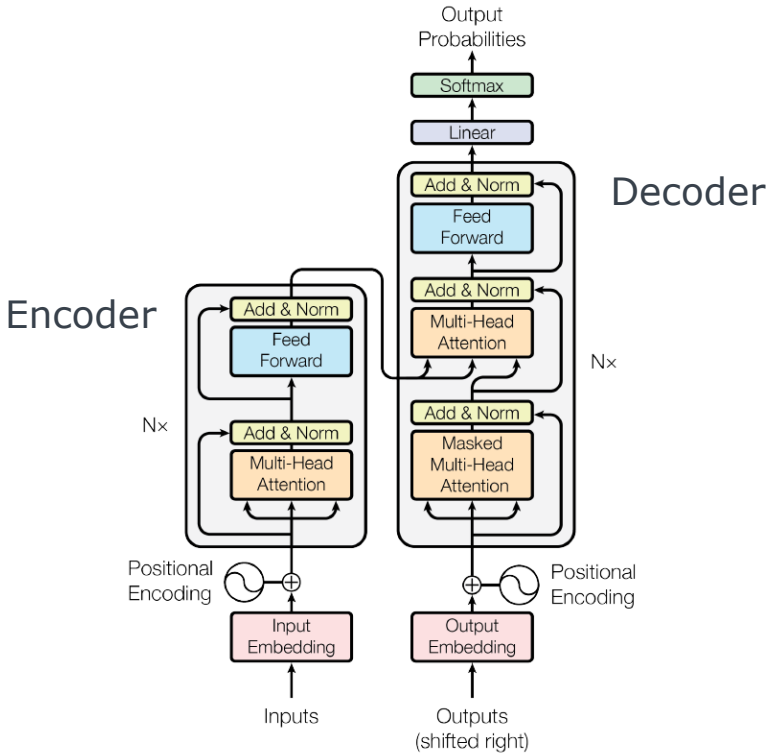
Feed forward:

- Einsatz eines einfachen, nicht-rekurrenten, neuronalen Netzwerks für Verarbeitung Ergebnisse.
- Speichern der Ergebnisse in einem Format für spätere Berechnung Wahrscheinlichkeit (analog Bilderkennung!)

Einsatz dieser Techniken ermöglichte Verbesserung der Ergebnisse sowie Reduzierung Ressourceneinsatz.

Quellen: Link, Link, Link

TRANSFORMER IM DETAIL. DECODER – OUTPUT.



Lineares Neuronales Netz

- so groß wie das gesamte Vokabular für Output, d.h. eine Zelle je Wort (gleiches Vorgehen wie Flatten-Vektor bei CNN!)

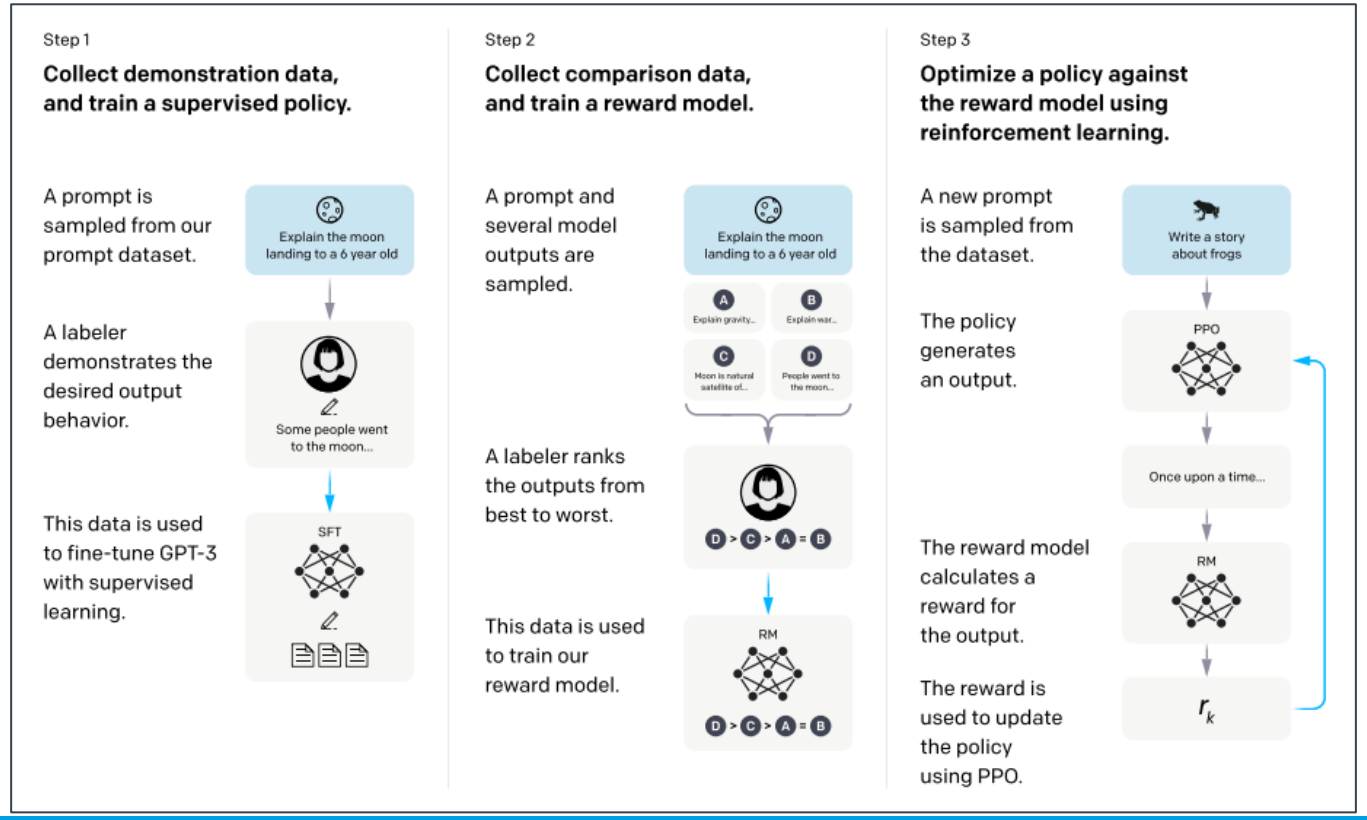
Softmax:

- Erstellt Wahrscheinlichkeitsverteilung für alle Zellen (d.h. über alle Wörter des gesamten Wörterbuchs).
- Die Zelle mit höchstem Wert für Wahrscheinlichkeit wird als vorhergesagtes nächstes Token genommen.

Zusammengefaßt ist die Aufgabe des Decoders das Sammeln aller notwendigen Informationen um mittels einer Wahrscheinlichkeitsverteilung aus dem gesamten Vokabular das nächste Ausgabewort möglichst genau vorherzusagen.

Quellen: Link, Link, Link

FINE-TUNING MODELL ERFOLGT MITTELS SUPERVISED UND REINFORCED LEARNING.



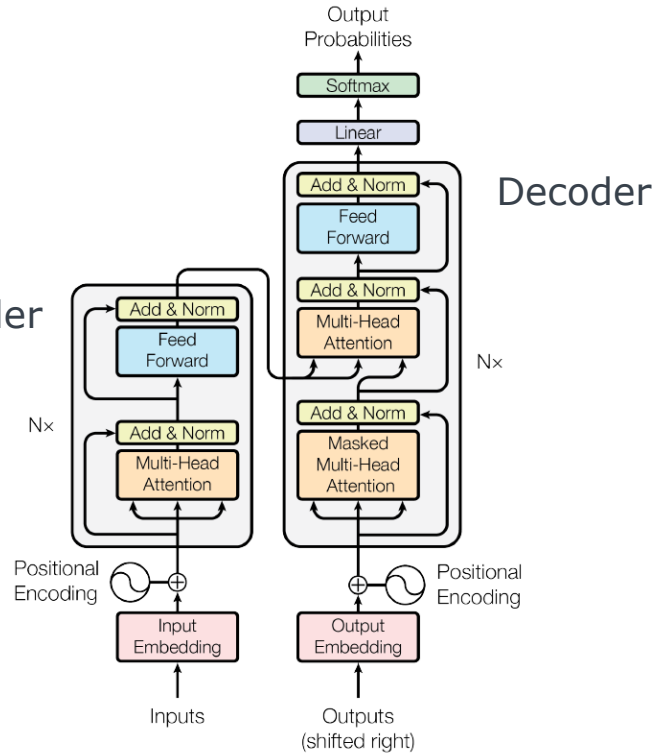
Detaillierung Schritte:

1. überwachtes Lernen: Menschen labeln/ bewerten fertige Chats als “Gut” oder “schlecht”.
2. Modell generiert verschiedene Outputs auf ausgewählte Prompts. Diese Outputs werden von Menschen gerankt. Dadurch lernt Modell eine Belohnungsfunktion für Bewertung seiner Ausgaben.
3. Belohnungsfunktion ermöglicht Modell, sich selber stets zu verbessern. Das heißt, es lernt, welche Ausgaben den höchsten Nutzen für Anwender und somit höchste Belohnung haben.

Schritte 1 und 2 sind aufgrund menschlicher Einbindung nicht beliebig skalierbar.
Das Lernen der Belohnungsfunktion emöglicht durch autonomes Verbessern Modell.

Quellen: Ouyang, Long, et al. “Training language models to follow instructions with human feedback“, 2022, Link

TRANSFORMER IM DETAIL. FALLBEISPIEL ZUR LAUFZEIT.



„Write a song about…"

„La la la…."

Inferenz/ Zur Laufzeit

- Start-Inputs:
  - Encoder: Input/ Prompt als Sequenz inclusive Start-of-Sentence und End-of-Sentence-Token.
  - Decoder: leere Sequenz mit "Start-of-Sentence"-Token.
- Diese beiden Sequenzen werden in das Modell eingegeben.
- Output Modell ist Wahrscheinlichkeitsverteilung für nächste Token.
- Das Token mit höchster Wahrscheinlichkeit wird genommen und an die Decoder-Sequenz angehängt (die jetzt 2 Tokens hat).
- Beide Sequenzen werden wieder ins Modell eingegeben, von der Decoder-Sequenz aber nur das letzte, gerade generierte, Token. Das Ergebnis des Modells wird wieder an die letzte Stelle der Decoder-Sequenz angehängt.
- Der vorige Schritt wird solange wiederholt bis entweder ein End-of-Sentence-Token vorhergesagt wird oder die maximale Satzlänge erreicht ist.

Quellen: Link, Link, Link

5. ZUSAMMENFASSUNG

CHAT GPT: VORTEILE UND HERAUSFORDERUNGEN.

Vorteile

- Ergebnisse sind oft richtig gut
- Unterstützt Menschen bei vielfältigen, repetitiven, aber auch kreativen Aufgaben
- Berücksichtigt Kontext
- Kontinuierliche Weiterentwicklung Modell
- Massiver Push für Thema AI

Herausforderungen

- Manchmal sind Ergebnisse komplett falsch, weil kein symbolisches Verständnis (Modell kennt nur was tausende im Netz schon so machten…)
- Sensitivität: ähnliche Inputwörter können zu sehr unterschiedlichen Ergebnissen führen.
- Es braucht weiter Wissen, um sicherzustellen, daß der generierte Text sinnvoll ist.
- Abhängigkeit Trainingsdaten: nicht kuratiert und deshalb ggf. problematisch (biased, nicht mehr gültig, ….)
- Sehr hoher Ressourcenbedarf
- Ersetzen von Menschen?
- Erklärbarkeit/ Modell ist nicht open source

LITERATUR UND WEITERE QUELLEN (AUSZUG).

LSTM:

- https://colah.github.io/posts/2015-08-Understanding-LSTMs/
- Hochreiter, Schmidhuber: Long short-term memory, 1997 (meistzitierte AI-Paper!).

Attention:

- https://distill.pub/2016/augmented-rnns/

Transformers:

- https://e2eml.school/transformers.html
- https://nlp.seas.harvard.edu/annotated-transformer/
- https://www.tensorflow.org/text/tutorials/transformer
- https://jalammar.github.io/illustrated-transformer/

Large Language Models:

- https://mark-riedl.medium.com/a-very-gentle-introduction-to-large-language-models-without-the-hype-5f67941fa59e

Stable Diffusion:

- https://bootcamp.uxdesign.cc/how-stable-diffusion-works-explained-for-non-technical-people-be6aa674fa1d
- https://medium.com/@steinsfu/diffusion-model-clearly-explained-cd331bd41166
- https://analyticsindiamag.com/10-awesome-google-colab-notebooks-on-stable-diffusion/
- https://jalammar.github.io/illustrated-stable-diffusion/

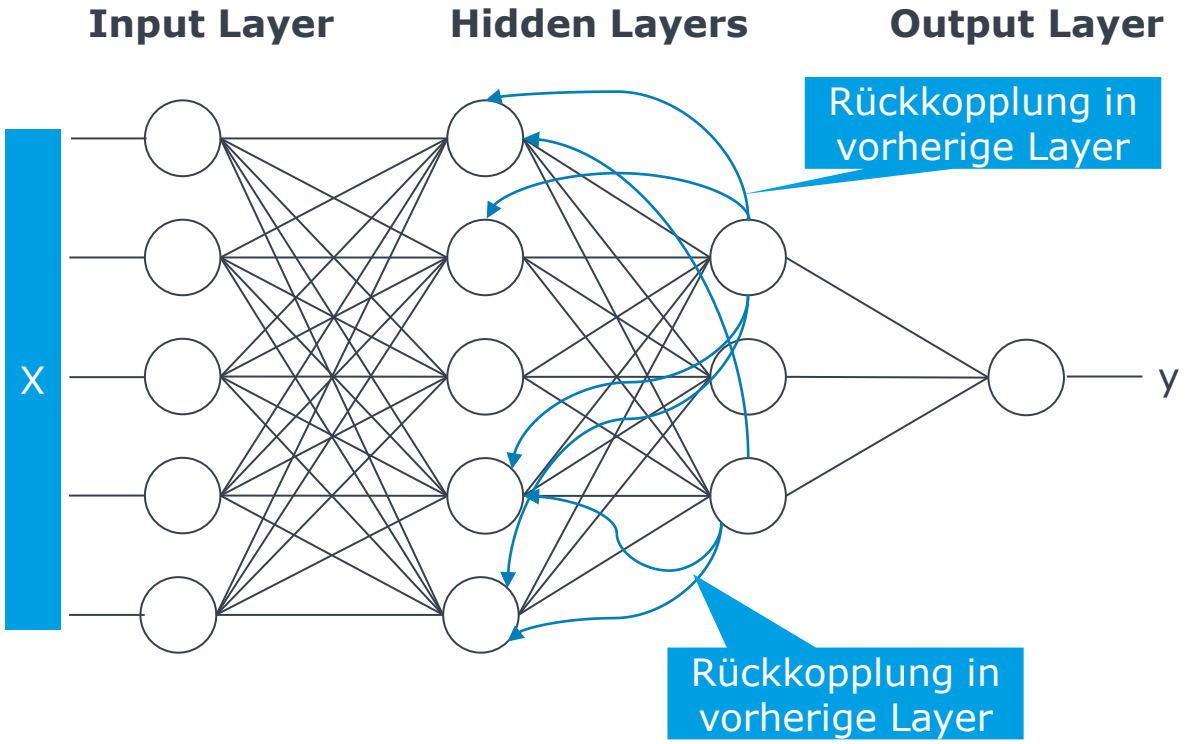
GPT:

- Karpathy: Entwicklung GPT2-Clone in 600 Zeilen Python: https://github.com/karpathy/nanoGPT und https://www.youtube.com/watch?v=kCc8FmEb1nY
- Jay Mody: https://jaykmody.com/blog/gpt-from-scratch/#putting-it-all-together und https://github.com/jaymody/picoGPT/blob/main/gpt2.py
- https://bootcamp.uxdesign.cc/how-chatgpt-really-works-explained-for-non-technical-people-71efb078a5c9
- https://medium.com/@fareedkhandev/create-gpt-from-scratch-using-python-part-1-bd89ccf6206a
- Meta Llama 2: Source Code und Paper verfügbar unter https://ai.meta.com/llama/

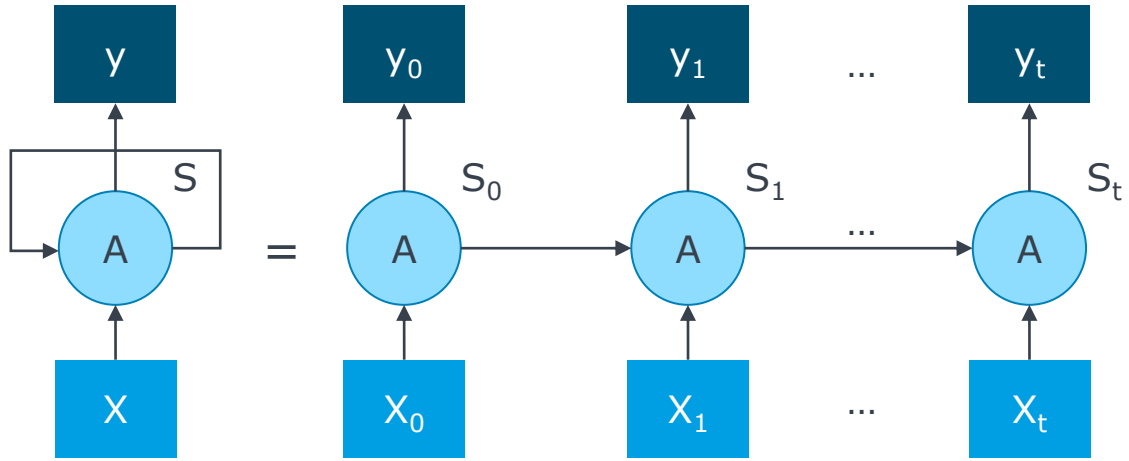
BACKUP

DETAILLIERUNG REKURRENTE NEURONALE NETZE

STRUKTUR REKURRENTE NEURONALE NETZE.



Detaillierung Struktur RNN über Zeitschritte t („Querschnittsbild")

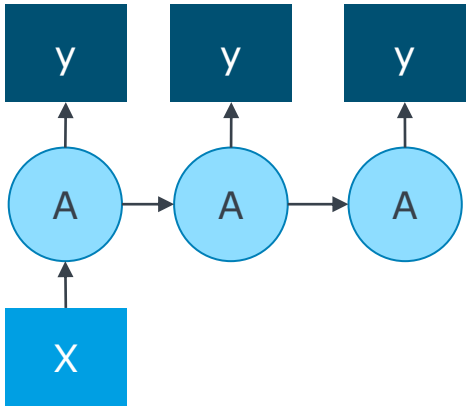


- Eingaben X_i: Sequenz wird auf Inputs aufgeteilt.
- Hidden State S_i: „Kurzzeitgedächtnis". Abhängig vom Input x_i sowie vorherigen Zustände S_i. Überträgt auch (gewichtet) Informationen zum nächsten Schritt.
- A: Neurales Netzwerk inklusive Rückkopplung Ergebnisse.
- Ausgaben y_i: Ausgaben/ Prädiktion des Netzwerks.

Quelle: rechtes Bild: Link und Link
Rekurrente Netzwerke definiert in: Elman, J.: "Finding structure in time", Cognitive Science, 14(2), 179–211, 1990.

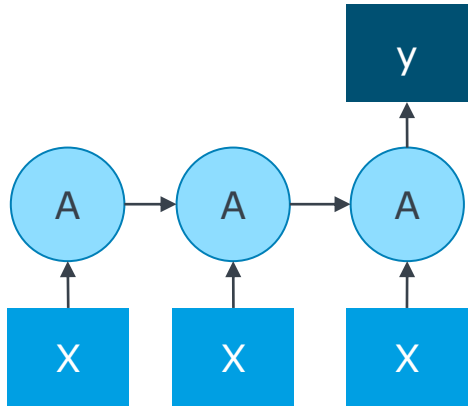
(GROB-)STRUKTUR RNN ABHÄNGIG VOM EINSATZGEBIET.

Ein Input, viele Outputs



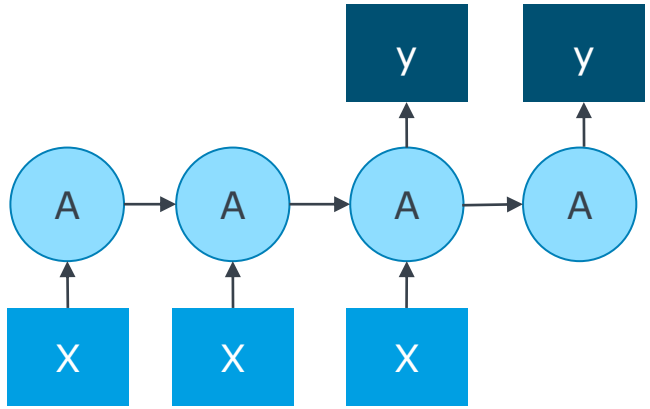
Sequence Output:
Bspw. automatisierte **Bilderkennung und -beschreibung**
(1 Bild mit n Wörten beschreiben)

Viele Inputs, ein Output



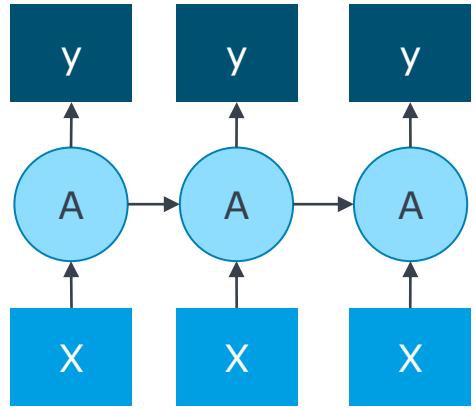
Sequence Input:
Bspw. automatisierte **Sentiment Analysis/ Gefühlsanalyse**
(1 Satz wird analysiert ob positive/ negative Aussagen)

Viele Inputs, viele Outputs



Sequence input and output:
Bspw. Automatisiertes **Übersetzen Texte** (Deutsch – Englisch), aber Input und Output können verschiedene Längen haben.

Viele Inputs, viele Outputs



Gleich lange Sequence input and output:
Bspw. Videoklassifikation, jeder Frame wird gelabelt

Frage: welche Struktur wird eingesetzt für
a. Bewerten ob positives/ negatives Kundenfeedback
b. Automatisiertes Übersetzen eines Satzes
c. Labeln Videoframes (je Frame)
d. Beschreiben eines Bildes mit Wörtern

Quelle: Link

BEKANNTE MACHINE LEARNING VERFAHREN FÜR TEXTE

ÜBERSICHT BEKANNTE MACHINE LEARNING VERFAHREN.

- **LSTM (1997)[1]:**
  - Rekurrentes Neuronales Netzwerk, entwickelt von Sepp Hochreiter und Jürgen Schmidhuber an der TU München.
  - aktuell (noch?) eines der häufigst eingesetzte Verfahren für sequentielle Daten, bspw. Spracherkennung in Handys (Link).
  - Kann viel längere Zeitsequenzen verarbeiten als normale RNN (keine Probleme mit Vanishing Gradient[2])
  - Ressourcenaufwendig: hoher Speicherbedarf und (sehr) aufwendige Trainingszeit.
  - Und neigt zum Overfitting (das man durch Tunen Hyperparameter und Struktur LSTM aber reduzieren kann).

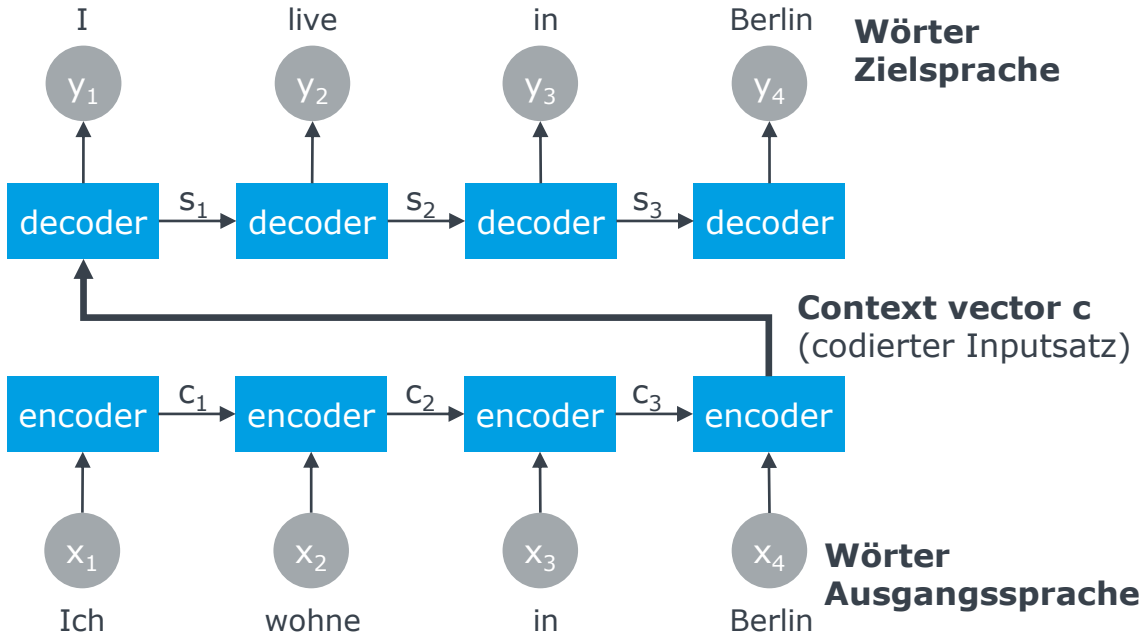
Wiederholungsfrage: was ist Overfitting?

- **Transformer (2017)[3]:**
  - Entwickelt von Google, basierend auf **Attention[4]-Verfahren** aus 2015 (basierend auf Schmidhuber 1992).
  - Kein RNN, sondern Kombination verschiedener sequentieller Verfahren ohne Rückkopplung (inkl. Transfer Learning).
  - Schnelleres Training als LSTM aufgrund Aufteilen Lernen auf mehrere Rechner (Parallelisieren).
  - Hat LSTM in vielen Gebieten abgelöst, bspw. für maschinelles Übersetzen, aufgrund höherer Flexibilität.
  - Einsatz analog Transfer Learning für State-of-the-Art Ansätze wie BERT (Bidirectional Encoder Representations from Transformers)[5] und GPT-3 (Generative Pre-trained Transformer 3)[6].

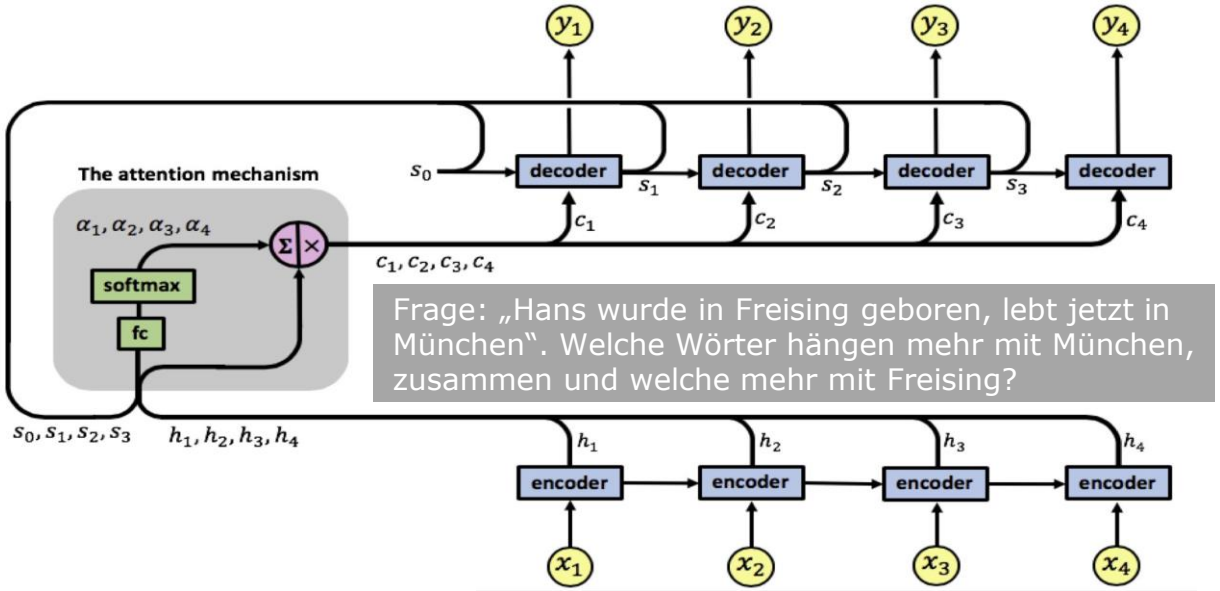
Quellen:
1 Hochreiter, Schmidhuber: "Long Short-Term Memory ", 1997. Link
2 Vanishing Gradient: beim Trainieren von Netzen mit vielen Schichten und Nutzen Expontentialfunktion als Aktivierungsfunktion kann bei Backpropagation Fall entstehen, daß Gradienten sehr, sehr klein werden und damit die Gewichte und Bias des Modells, v.a. der ersten Schichten, nicht mehr geändert/ trainiert werden. Vgl. Hochreiter: „Untersuchungen zu dynamischen neuronalen Netzen", 1991.
3 Vaswani et al.: "Attention is all you need ", 2017. Link
4 Bahdanau et al.: "Neural Machine Translation by Jointly Learning to Align and Translate", 2015. Link. Basierend auf Schmidhuber, "Learning to control fast-weight memories.", 1992.
5 Devlin et al.: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", 2019. Link
6 Brown et al:: "Language Models are Few-Shot Learners", 2020

(VEREINFACHTES) ATTENTION – MOTIVATION.

Übersetzen Ausgangs- in Zielsprache bei RNN



Optimierung durch Attention Mechanismus



Herausforderungen/ Probleme bisheriger Verfahren:

- letztes Wort Ausgangssprache ist Input erstes Wort für Zielsprache; dabei sind oft erste Wörter ähnlich.
- „Normales" RNN: Länge Decoder (Satz einer Zielsprache) festgelegt, was zu Problemen beim Lernen von langen Sätzen führt (umgangssprachlich: Modell „vergisst" Kontext).
- RNN/ LSTM: sehr hoher Rechenaufwand & nicht parallelisierbar.

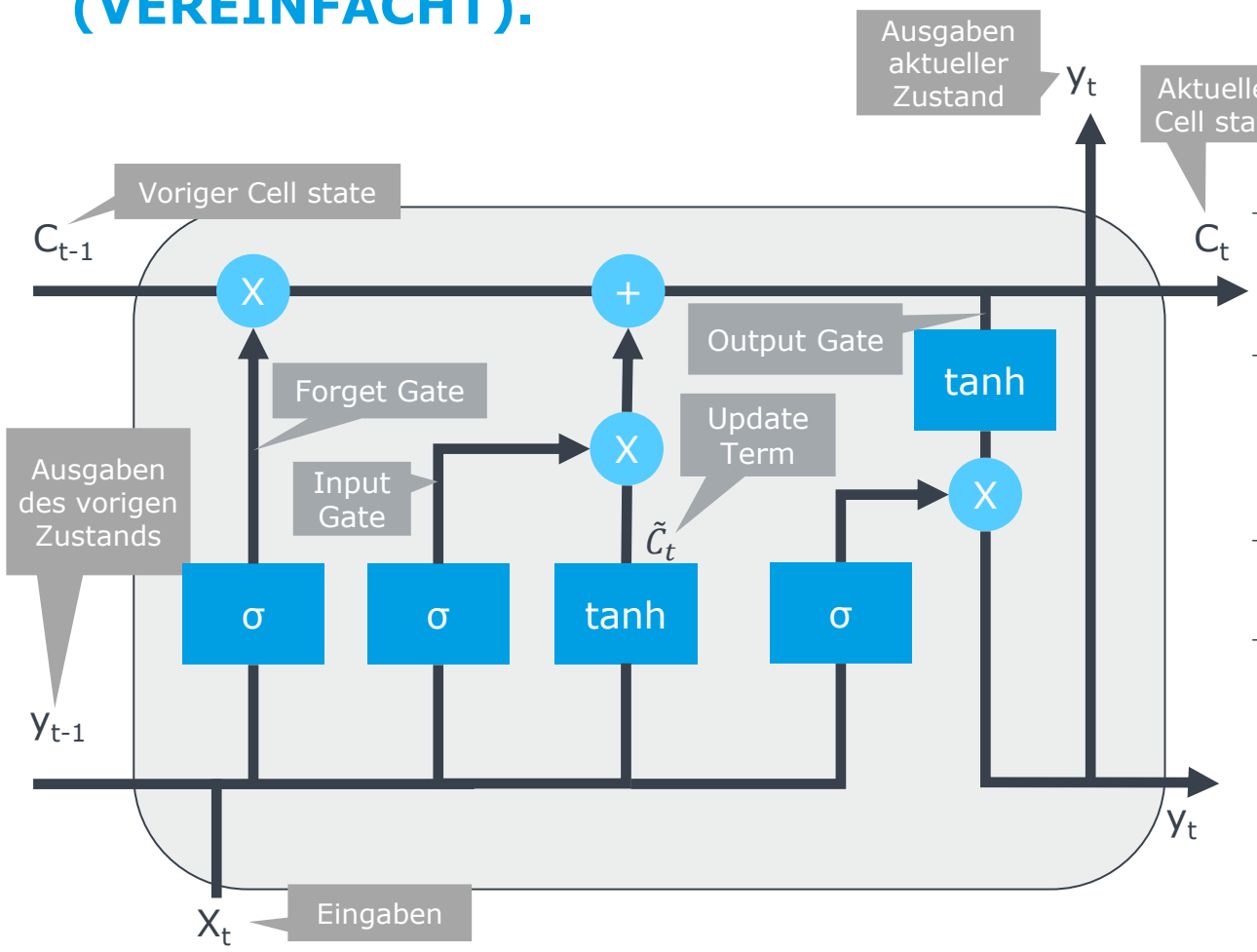
Attention löst die erwähnten Herausforderungen:

- Attention ermöglicht dem Decoder, bestimmte Teile des Inputs stärker zu gewichten bei der Vorhersage des Outputs (hier: über c_1 – c_4). Lernen dieser Attention-Gewichte (a_1 – a_4) per neuronalem Netz.
- Das Problem der Satzlänge wird durch individuelle Gewichte für jeden Output gelöst. Von dieser unterschiedlichen Fokussierung stammt der Name Attention.

Quelle Bild: Link

DETAILLIERUNG LSTM

LONG SHORT-TERM MEMORY: STRUKTUR UND ABLAUF (VEREINFACHT).



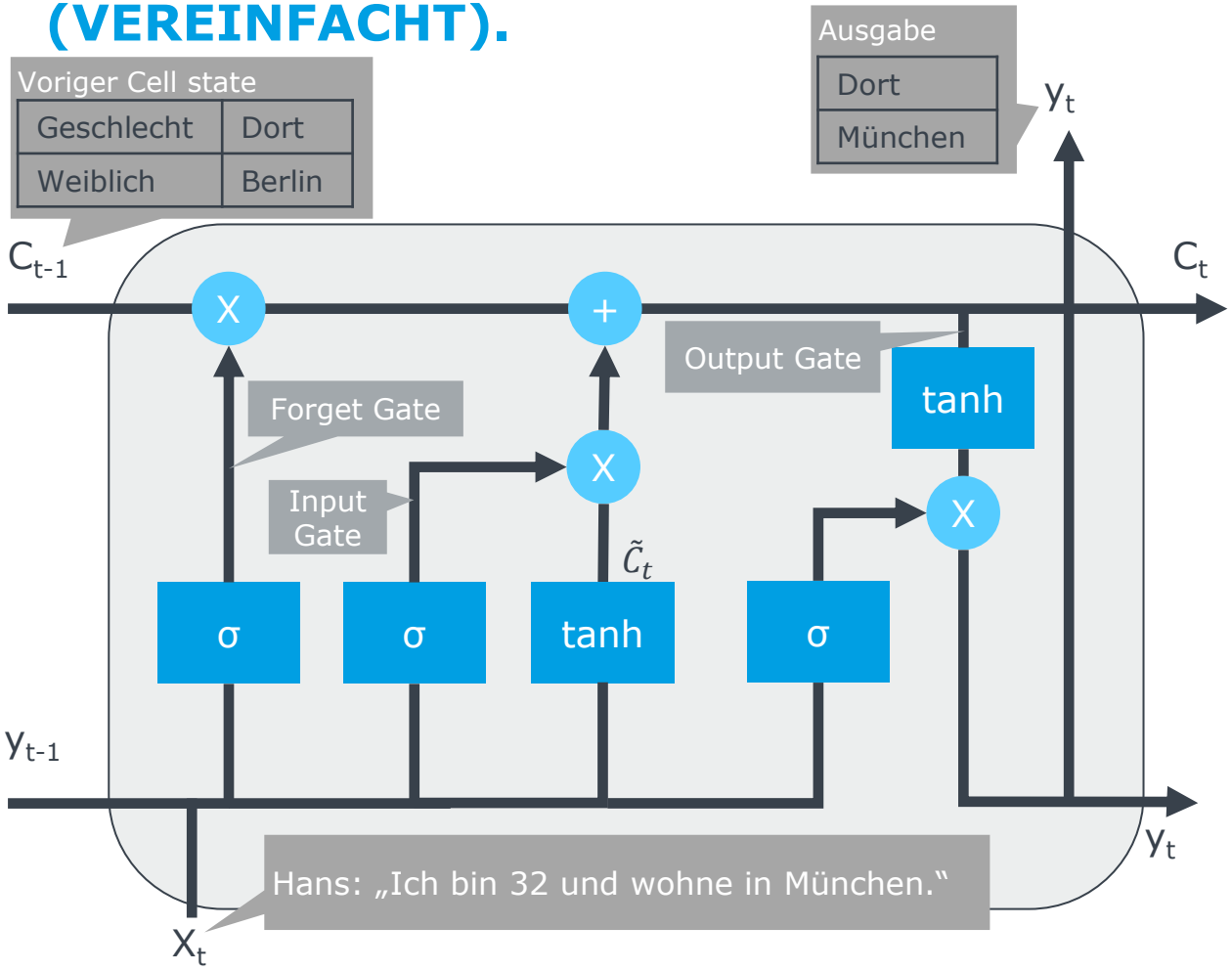
Wir schauen uns eine von mehreren LSTM-Zelle an, die miteinander verknüpft sind, d.h. die abgebildete Zelle hat „Nachbarzellen".

- **Forget Gate:** entscheidet, welche Infos des Cell State C_{t-1} zu vergessen oder weiter zu erinnern sind. Geschieht per Parameter mit Wert 0 für Vergessen, 1 für Erinnern oder Wert dazwischen, falls Gate unsicher ist.
- **Input Gate:** hat zwei Schritte.
  Schritt 1: Entscheidet, welche Infos für Cell State C_t zu aktualisieren sind. Erfolgt per Parameter mit Werten von 0 (irrelevant) bis 1 (relevant).
  Schritt 2: Bestimmt Update-Terme, d.h. die neuen Werte für den Cell State.
- **Cell State:** Hier wird die Zelle C_t aktualisiert, basierend auf Inputs des Forget Gate und Input Gates.
- **Output Gate:** entscheidet was aus der Zelle ausgegeben wird.

LSTM vermeidet Vanishing Gradient durch:
- Steuerung Verhalten Gradienten durch Werte Gates: Wert 0 verhindert bspw. Änderung Gradient.
- Werte des Gates für Vermeiden Explosion werden gelernt.
- Formel für Ausgabe C_t enthält Addition. Dadurch hat die den Gradienten erzeugende Ableitung ein „besseres Verhalten" als bei bspw. Multiplikation

Quellen:
Bild: Link.
LSTM: Hochreiter, Schmidhuber: „Long Short-Term Memory", 1997 sowie Link, Link.
Erklärung Vermeidung Vanishing Gradient: Link

LONG SHORT-TERM MEMORY: FALLBEISPIEL (VEREINFACHT).



Ziel: „Verstehen" einer Konversation durch Algorithmus
Anna: „Ich bin 30 Jahre alt und wohne in Berlin"
Hans: „Ich bin 32 und wohne in München."
Anna: „Wie viele Menschen leben dort?"

Frage: worauf bezieht sich dort?

Start mit Hans' Satz: Cell state C_{t-1}:

Geschlecht	Dort
Weiblich	Berlin

- **Forget Gate:** vergiss Werte für Geschlecht und Wohnort (Forget gate = 0), da mit Hans andere Person mit anderem Geschlecht und Wohnort spricht.
- **Input Gate:**
  - Schritt 1: bestimme zu aktualisierende Werte
  - Schritt 2: bestimme Änderungswerte $\tilde{C}_t$

Geschlecht	Dort
Männlich	München

- **Cell State:** schreibe die Werte in die Zelle C_t

Geschlecht	Dort
Männlich	München

- **Output Gate:** schreibe in Ausgabe y_t Wert für dort, da dieser Begriff für Annas nächsten Satz relevant ist. LSTM lernt also, daß Annas Frage sich auf München bezieht!

Dort
München

Quellen:
Bild: Link.
LSTM: Hochreiter, Schmidhuber: „Long Short-Term Memory", 1997 sowie Link, Link.
Erklärung Vermeidung Vanishing Gradient: Link

CASE STUDY LSTM

WHEN TAYLOR MEETS HELENE…

Use Case 1: Taylor Generator: basierend auf bestehenden Taylor Swift Lyrics automatisiert möglichst ähnliche Texte erstellt (Natural Language Generation)

Use Case 2: Helenes neue Zielgruppe Amerika: Automatisiertes Übersetzen Helene Fischer Lieder in Englisch (**maschinelles Übersetzen)**

When Taylor meets Helene: wir geben übersetzte Helene Texte in den Taylor-Generator ein.

WIEDERHOLUNG GENERISCHER ABLAUF SUPERVISED LEARNING. ÜBERSICHT.

1. Daten organisieren und hochladen
2. Daten aufbereiten/ Data cleaning
3. Daten aufteilen in Test- und Trainingsmenge (sowie ggf. Validierungsmenge)
4. Vorbereitungen: Machine Learning Verfahren wählen, Gewichte initialisieren, Kostenfunktion wählen
5. Training: Schrittweise Optimierung Modellparameter bis Modell möglichst gute Performance für die Trainingsmenge hat
6. Modell(-güte) validieren anhand der Testmenge
7. Deployment: Modell einsetzen im „Live"-Betrieb inkl. kont. Überprüfen Güte Modell und Aktualisierung

WHEN TAYLOR MEETS HELENE
SCHRITT 1: DATEN ORGANISIEREN.

Textgenerierung

"Shake it off"

"I knew you were trouble"

"Tim McGraw"
He said the way my blue eyes shined
Put those Georgia stars to shame that night
I said, "That's a lie"
Just a boy in a Chevy truck
That had a tendency of gettin' stuck
On back roads at night
And I was right there beside him all summer long
And then the time we woke up to find that summer gone

Automatisiertes Abgreifen der Texte aller Lieder und Speichern als Textdatei

Maschinelles Übersetzen

Tab-delimited Bilingual Sentence Pairs

These are selected sentence pairs from the Tatoeba Project.

Updated: 2020-08-23

- Afrikaans - English afr-eng.zip (807)
- Albanian - English sqi-eng.zip (451)
- Algerian Arabic - English arq-eng.zip (155)
- Arabic - English ara-eng.zip (11446)
- Assamese - English asm-eng.zip (1655)
- Azerbaijani - English aze-eng.zip (2208)
- Basque - English eus-eng.zip (677)
- Belarusian - English bel-eng.zip (3825)
- Bengali - English ben-eng.zip (4342)
- Berber - English ber-eng.zip (120079)
- Bosnian - English bos-eng.zip (102)
- Bulgarian - English bul-eng.zip (14979)
- Burmese - English mya-eng.zip (347)
- Cantonese - English yue-eng.zip (3213)
- Catalan - English cat-eng.zip (665)
- Cebuano - English ceb-eng.zip (205)
- Central Dusun - English dtp-eng.zip (1521)
- Chavacano - English cbk-eng.zip (1905)

Introducing Anki

- If you don't already use Anki, vist the website at http://ankisrs.net/ to download this free application for Macintosh, Windows or Linux.

About These Files

- Any flashcard program that can import tab-delimited text files, such as Anki (free) can use these files.
- **Warning!** There are errors in the Tatoeba Corpus. (Detailed Warning)
- **In order to minimize the number of errors**, I only used sentences that were owned by identified native speakers working on the Tatoeba Project and English sentences that I've personally checked and did not reject.
- **Warning!** Please remember that even doing this may not have eliminated all errors.

How the Data Looks

English + TAB + The Other Language + TAB + Attribution

```
This work isn't easy.    この仕事は簡単じゃない。    CC-BY 2.0 (France) Attributi
Those are sunflowers.    それはひまわりです。        CC-BY 2.0 (France) Attributi
Tom bought a new car.    トムは新車を買った。        CC-BY 2.0 (France) Attributi
This watch is broken.    この時計は壊れている。      CC-BY 2.0 (France) Attributi
```

The attribution gets imported into Anki as a tag, by default This attribution contains the

Herunterladen Deutsch – Englisch „Wörterbuch" mit ~221'000 Deutschen Sätzen.

WHEN TAYLOR MEETS HELENE
SCHRITT 2: DATEN SÄUBERN.

Textgenerierung

Embedding auf Basis Buchstaben

He said the way my blue eyes shined
Put those Georgia stars to shame that night
I said, "That's a lie"

he said the way my blue eyes shined
put those georgia stars to shame that night
i said thats a lie

{0: ' ', 1: 'a', 2: 'b', 3: 'c', 4: 'd', 5: 'e', 6: 'f', 7: 'g', 8: 'h', 9: 'i', 10: 'j', 11: 'k', 12: 'l', 13: 'm', 14: 'n', 15: 'o', 16: 'p', 17: 'q', 18: 'r', 19: 's', 20: 't', 21: 'u', 22: 'v', 23: 'w', 24: 'x', 25: 'y', 26: 'z'}

Maschinelles Übersetzen

Embedding auf Basis Wörter

Tom kommt heute.

tom kommt heute

Input Language embedding
1 ----> <start>
5 ----> tom
144 ----> kommt
114 ----> heute
3 ----> .
2 ----> <end>

Tom is coming today.

Tom is coming today

Target Language embedding
1 ----> <start>
5 ----> tom
8 ----> is
185 ----> coming
133 ----> today
3 ----> .
2 ----> <end>

- Alle Wörter in Kleinschreibung umwandeln
- Sonderzeichen entfernen
- Embedding für Wörter/ Buchstaben: zuweisen eindeutige Nummer

AUTOMATISIERTES ERSTELLEN TAYLOR SWIFT SONGS. SCHRITT 3: DATEN AUFTEILEN IN TRAININGS- UND TESTMENGE.

Textgenerierung

```
seq_length = 100

Data_X_Taylor = []
Data_y_Taylor = []

for i in range(0, n_chars_Taylor - seq_length, 1):
    # Input sequence(will be used as samples)
    seq_in  = Cleaned_Taylor_lyrics[i:i+seq_length]

    # Output sequence (will be used as target)
    seq_out = Cleaned_Taylor_lyrics[i + seq_length]

    # Store samples in data_X
    Data_X_Taylor.append([chars2int_Taylor[char] for char in seq_in])

    # Store targets in data_y
    Data_y_Taylor.append(chars2int_Taylor[seq_out])

n_patternsTaylor = len(Data_X_Taylor)

print( 'Total Patterns : ', n_patternsTaylor)
```

```
Total Patterns :  444330
```

```
# Vektor anpassen damit er in LSTM RNN eingespeist werden kann
X_Taylor = np.reshape(Data_X_Taylor, (n_patternsTaylor, seq_length, 1))

# Normalizing input dat
X_Taylor = X_Taylor/ float(n_vocab_Taylor)

# One hot encode the output targets :
y_Taylor = np_utils.to_categorical(Data_y_Taylor)
```

```
print(X_Taylor.shape[1], X_Taylor.shape[2])
```

Verhältnis 70% - 30%

Maschinelles Übersetzen

```
def load_dataset(path, num_examples=None):
  # creating cleaned input, output pairs
  #targ_lang, inp_lang = create_dataset(path, num_examples)
  eng, deu = create_dataset(path, num_examples)

  input_tensor, inp_lang_tokenizer = tokenize(deu)
  target_tensor, targ_lang_tokenizer = tokenize(eng)

  return input_tensor, target_tensor, inp_lang_tokenizer, targ_lang_tokenizer
```

```
num_examples = 50000
input_tensor, target_tensor, inp_lang, targ_lang = load_dataset(path2file, num_examples)

# Calculate max_length of the target tensors
max_length_targ, max_length_inp = target_tensor.shape[1], input_tensor.shape[1]
```

```
# Creating training and validation sets using an 80-20 split
input_tensor_train, input_tensor_val, target_tensor_train, target_tensor_val = train_test_split(input_tensor, target_tensor, test_size=0.2)

# Show length
print(len(input_tensor_train), len(target_tensor_train), len(input_tensor_val), len(target_tensor_val))
```

```
40000 40000 10000 10000
```

Verhältnis 80% - 20%

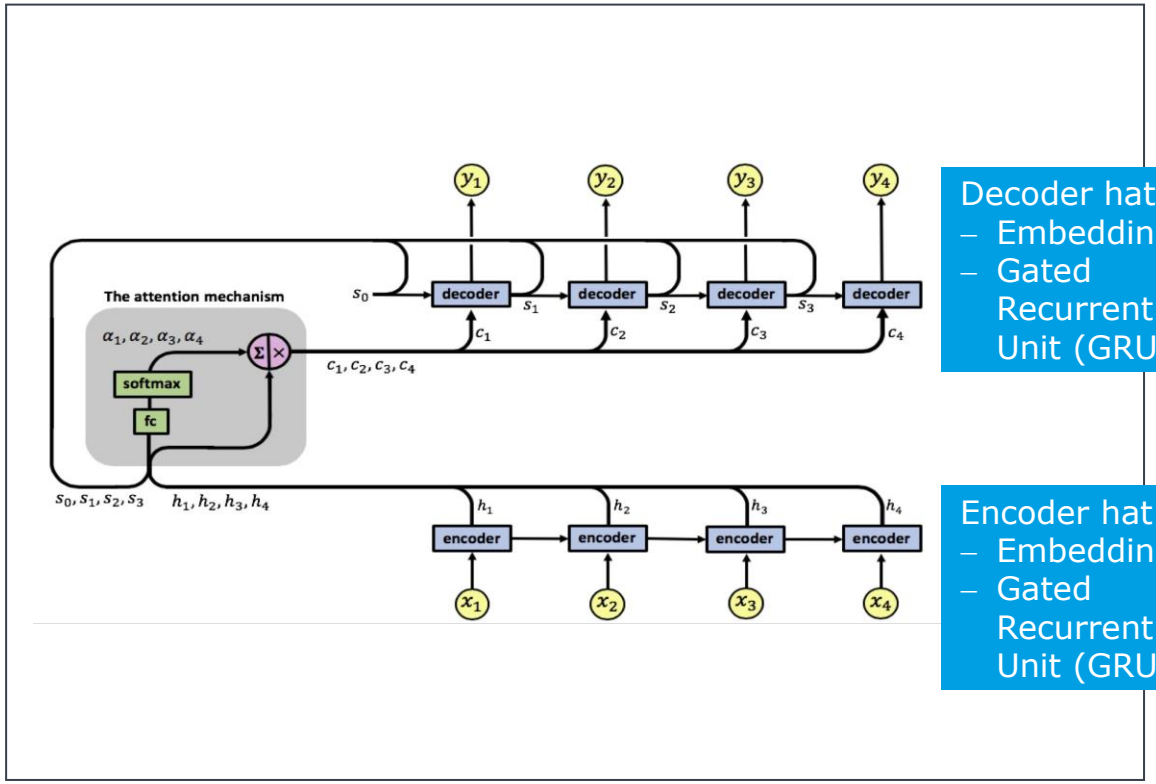
AUTOMATISIERTES ERSTELLEN TAYLOR SWIFT SONGS. SCHRITT 4: MACHINE LEARNING VERFAHREN WÄHLEN.

Textgenerierung

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_4 (LSTM)                (None, 100, 256)          264192
_________________________________________________________________
dropout_4 (Dropout)          (None, 100, 256)          0
_________________________________________________________________
lstm_5 (LSTM)                (None, 100, 256)          525312
_________________________________________________________________
dropout_5 (Dropout)          (None, 100, 256)          0
_________________________________________________________________
lstm_6 (LSTM)                (None, 100, 256)          525312
_________________________________________________________________
dropout_6 (Dropout)          (None, 100, 256)          0
_________________________________________________________________
lstm_7 (LSTM)                (None, 100, 256)          525312
_________________________________________________________________
dropout_7 (Dropout)          (None, 100, 256)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 25600)             0
_________________________________________________________________
dense_1 (Dense)              (None, 29)                742429
_________________________________________________________________
activation_1 (Activation)    (None, 29)                0
=================================================================
Total params: 2,582,557
Trainable params: 2,582,557
Non-trainable params: 0
_________________________________________________________________
```

4 gekoppelte LSTM mit jeweils einem Dropout zur Reduzierung Overfit

Maschinelles Übersetzen



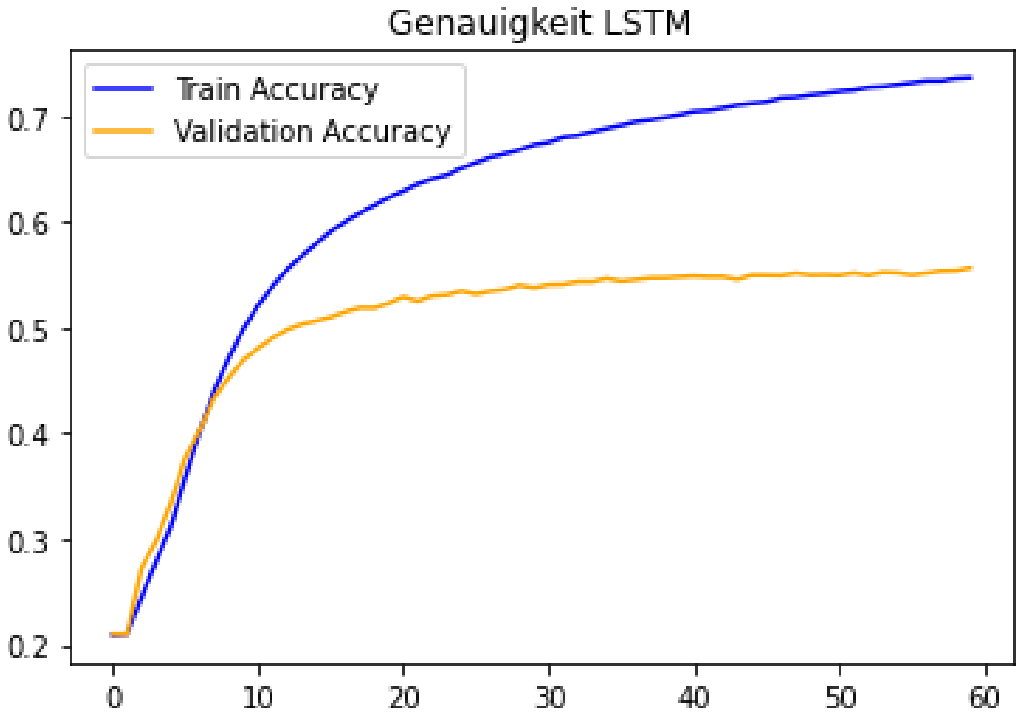
Decoder hat:
- Embedding
- Gated Recurrent Unit (GRU)

Encoder hat:
- Embedding
- Gated Recurrent Unit (GRU)

Vereinfachtes Beispiel: 1 GRU für En/Decoder und Attention mit 10x Fully Connected Layer

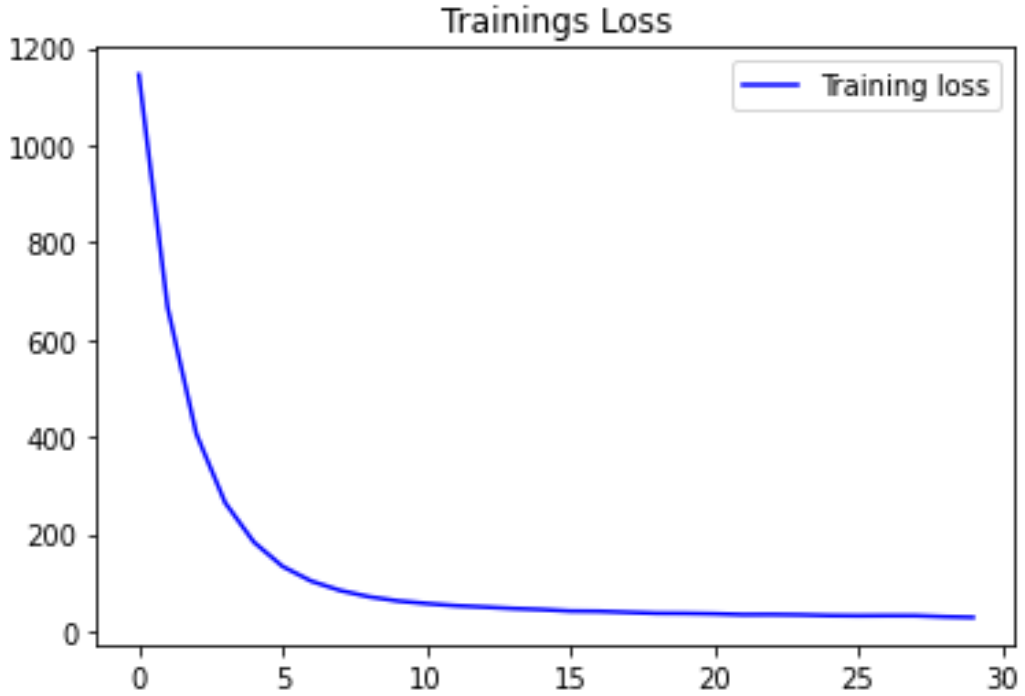
AUTOMATISIERTES ERSTELLEN TAYLOR SWIFT SONGS. SCHRITT 5: TRAINING

Textgenerierung



60 Epochen

Maschinelles Übersetzen



30 Epochen

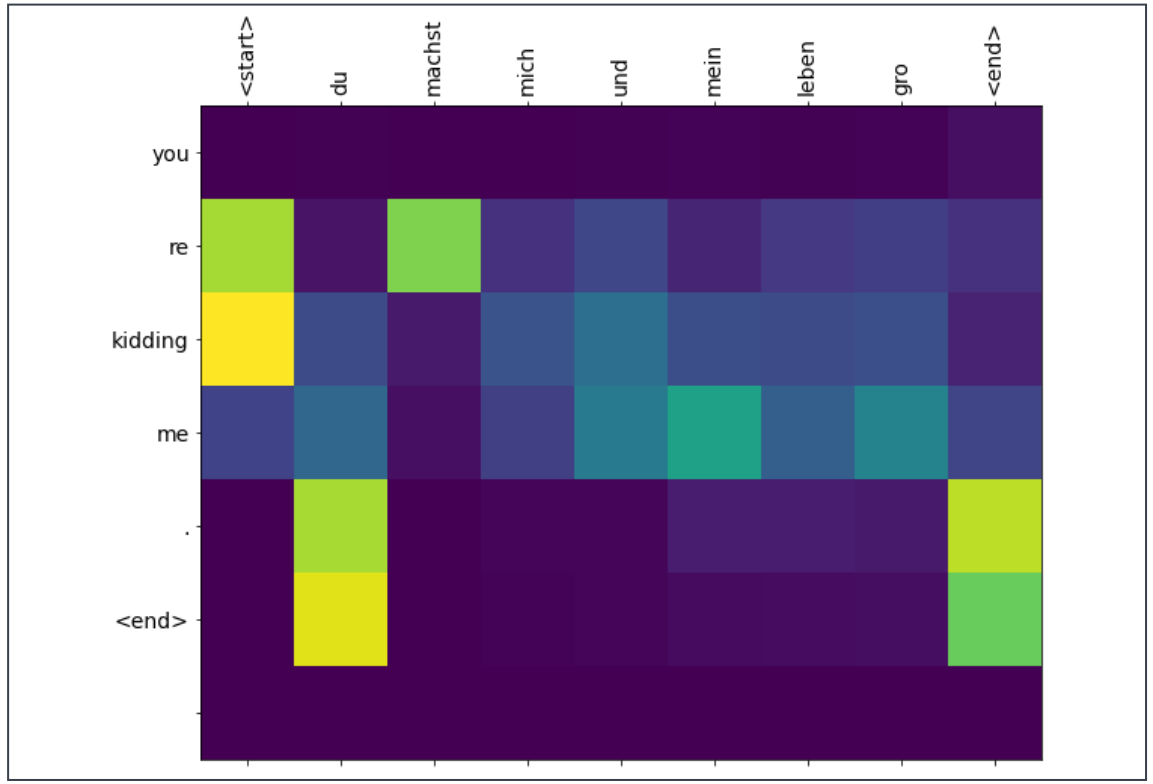
AUTOMATISIERTES ERSTELLEN TAYLOR SWIFT SONGS. SCHRITT 6: VALIDIERUNG MODELLGÜTE PER TESTDATEN.

Textgenerierung

e because i mean the crowd was going so loud that if he would have said no they would have probably " boy i had the time i was an american gor i lifht be the things they did to say that i dont know i want to be alone i need you and i can see you staring in the world was riding in your all and you dont leave me like the way you want me with me and you were the way you want to walk away the way at wo Done

hate hate hate hate hate baby im just gonna shake shake shake shake shake i shake it off i shake it off heartbreakers gonna break break break break break and the fakers gonna fake fake fake fake fake baby im just gonna shake shake shake shake shake i shake it off i shake it off heartbreakers gonna break break break break break and the fakers gonna fake fake fake fake fake baby im just gonna shake shake shake shake shake i shake it off i shake it off heartbreakers gonna break break break break br Done

Maschinelles Übersetzen



Qualität bei Textgenerierung und Übersetzen nicht ausreichend; deutlich längeres Trainieren als 60 Epochen notwendig