1. Better speech synthesis through scaling

Betker, J. (2023). *Better speech synthesis through scaling*. Verfügbar unter: https://github.com/neonbjb/tortoise-tts

Ziel des Projekts:

Die Intention bei der Implementierung dieses Projektes war, Fortschritte in der Bildgenerierung durch autoregressive Transformatoren und Denoising Diffusion Probabilistic Models (DDPMs) auf die Sprachsynthese anzuwenden. Der Autor wollte zeigen, dass Methoden, die im Bereich der Bildgenerierung erfolgreich waren, auch für die Verbesserung von Text-to-Speech (TTS)-Systemen genutzt werden können.

Wie werden die Stimmen generiert?

Begriffsklärung:

Autoregressives Modell (AR-Modell)	Erzeugt eine große Anzahl von Sprachkandidaten basierend auf den Eingabedaten und dem Text.
Contrastive Language-Voice Pretrained Transformer (CLVP)	Bewertet die Korrelation zwischen den generierten Sprachkandidaten und dem Text, um die besten Kandidaten auszuwählen.
Diffusionsmodell (hier: Denoising Diffusion Probabilistic Model)	Wandelt das ausgewählte Sprachmodell / Kandidat in ein MEL-Spektrogramm um.
Vocoder (Univnet)	Wandelt das MEL-Spektrogramm in eine Audio-Wellenform um.

Generierung von Stimmen:

1. Training des AR-Modells:

Das AR-Modell wird auf großen Datensätzen trainiert, um Sprachsequenzen zu generieren. Es verwendet dabei Transformer-Schichten, die sich durch ihre Effizienz und Leistungsfähigkeit auszeichnen.

2. Nutzung des CLVP-Modells:

Die generierten Sprachkandidaten werden vom CLVP bewertet und im Anschluss werden die besten Kandidaten davon ausgewählt. Der CLVP verwendet dabei eine kontrastive Lernmethode, um die besten Übereinstimmungen zwischen Text und Sprache zu finden, ohne dass das ressourcenintensive Diffusionsmodell eingesetzt werden muss.

3. Diffusionsmodell:

Die ausgewählten Sprachkandidaten werden dann durch das Diffusionsmodell in ein MEL-Spektrogramm umgewandelt. Das Diffusionsmodell beginnt mit einem verrauschten Spektrogramm und führt einen stufenweisen Entschlackungsprozess durch. In jedem Schritt entfernt das Modell etwas Rauschen und nähert sich immer weiter dem Zielsprachspektrogramm. Durch die schrittweise Entfernung des Rauschens wird ein hochdimensionales, realistisches Sprachspektrogramm rekonstruiert.

4. Vocoder (Univnet):

Das erzeugte Sprachspektrogramm wird schließlich durch einen Vocoder in eine Audio-Wellenform umgewandelt, die als endgültiges Sprachergebnis ausgegeben wird.

2. Abhängigkeiten und Installation (unter Windows)

Voraussetzungen:

- Anaconda ist installiert
- Repository wurde geklont

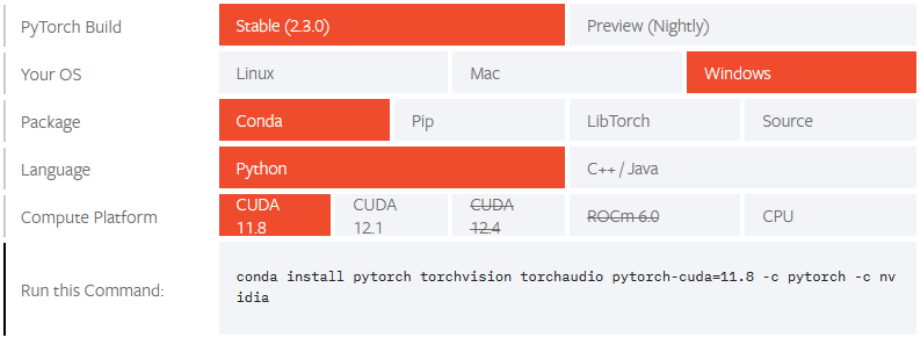
1. Python-Environment erstellen

- `conda create -n <name> python==3.9`

2. Environment aktivieren

- `conda activate <name>`

3. PyTorch installieren (https://pytorch.org/get-started/locally/#start-locally)



- NVIDEA Grafikkarte → Compute Platform: CUDA 11.8 / CUDA 12.1
- Keine NVIDEA Grafikkarte → Compute Platform: CPU
- Befehl kopieren und ausführen im Conda Environment

4. PySoundFile installieren (Windows spezifisch)

- `conda install -c conda-forge pysoundfile`

5. Projekt-Requirements installieren

- Im Terminal zum Tortoise Ordner (by default: „tortoise-tts-main“) navigieren
- `pip install -r requirements.txt`
- `python setup.py install`

6. Eigene Samples synthetisieren

- tortoise-tts-main/tortoise/voices
  → neuen Ordner (<my_voice>) mit Stimmsamples anlegen
  - ca. 10s lange Segmente
  - mind. 3 Clips
  - .wav Format
  - Aufnahmen mit möglichst wenig Hintergrundgeräuschen
- **One-Phrase-Generation**
  - `python tortoise/do_tts.py --text "<my_sentence>" --voice <my_voice> --preset fast`
- **Längere Texte vertonen**
  - tortoise-tts-main\tortoise\data
    → .txt Datei (<my_text>) anlegen mit Text, der gesprochen werden soll
  - `python tortoise/read.py --textfile <full_path_to_my_text.txt> --voice <my_voice> --preset standard`
- Ergebnisse sind in tortoise-tts-main/results zu finden

(Ein Video-Tutorial ist hier zu finden.)

Verschiedene Presets:

- **“ultra_fast”:** schnellste Methode, um Ergebnisse zu generieren
- **„fast“:** anständige Sprachqualität zu einer anständigen Generierungszeit; gut, wenn große Mengen an Text synthetisiert werden sollen
- **„standard“:** sehr gute Qualität, in dem meisten Fällen so gut wie es überhaupt geht
- **„high_quality“:** zu benutzen, wenn man das absolut beste Ergebnis haben möchte, („This is not really worth the compute though.“)

3. Eigene Experimente / Erfahrungen

Die Synthese hat je nach Menge des zu vertonenden Textes unterschiedlich lang gedauert. In meinem Fall waren es immer zwischen 4 und 6h. Ich habe dabei mehrere verschiedene Stimmen ausprobiert: männliche und weibliche Stimmen, mit und ohne Akzent, deutsch und englisch. Dabei hat sich herausgestellt, dass männliche Stimmen ohne Akzent, die einen englischen Text vertonen, in meinen Versuchen am besten funktioniert haben. Beim Versuch einen deutschen Text (mit deutschen Trainingssamples) zu vertonen, hat die Tonlage zwar gestimmt, aber der Text wurde eher „englisch" ausgesprochen. Wenn man wusste, wie der Basistext lautete, dann konnte man es halbwegs verstehen, aber andernfalls klang es eher wie eine komplett neue Sprache.

Bei Stimmen mit Akzent in den Trainingsdaten sind die Resultate gut ausgefallen, jedoch war der Akzent nicht mehr vorhanden, wodurch sie doch recht stark vom Original abgewichen sind.

Das beste Ergebnis hat sich bei englischsprachigen Stimmen ohne Akzent herausgestellt. Auch wenn diese in manchen Fällen regeneriert werden mussten, sind am Ende sehr gute Ergebnisse erzielt worden. Die beiden Besten habe ich einmal hier verlinkt:

- Video (dt. Synchro von Ryan Reynolds, [failed Example])
- Video (Angelina Jolie erzählt den Anfang des Märchens „Rotkäppchen")
- Video (Benedict Cumberbatch im Kontext einer Pottwal-Doku)

4. Möglichkeiten der Stimmsynthee

Die Stimmsynthese mithilfe von künstlicher Intelligenz ermöglicht es mittlerweile realistische Stimmen mit hoher Klangqualität zu erzeugen. Die Ergebnisse können in verhältnismäßig kurzer Zeit erzeugt werden und bleiben dabei von der Intonation konstant. Gerade wenn ruhige Stimmen benötigt werden und es nicht auf Synchronität mit Lippenbewegungen ankommt, sind die KI-generierten Stimmen sehr gut geeignet. In der Cumberbatch-Doku sind die Ergebnisse z.B. kaum zu unterscheiden von herkömmlichen Dokumentationen. Und dabei sind die Ergebnisse nur in einem Bruchteil der Zeit entstanden. Auf diese Weise kann jede geeignete Stimme für alles Mögliche verwendet werden, ohne dass man bspw. Termine im Tonstudio ausmachen muss und dort mehrere Takes aufnehmen muss. Allerdings ist das gleichzeitig wohl auch das größte Problem / der größte Kritikpunkt dieser Technologie. Heutzutage existieren schon Deepfakes von allen möglichen Stimmen, z.B. von US-Präsidenten oder auch von berühmten Schauspieler:innen. Diese können dann sehr leicht für unethische Dinge wie z.B. die Manipulation von Wahlkämpfen oder auch die Erstellung von bspw. Pornografischen Inhalten missbraucht werden, ohne dass die eigentlichen Stimmbesitzer dem zugestimmt haben. Auch bei Telefonbetrug könnte diese Technologie missbraucht werden.

Ich denke nicht, dass diese Technologie allen Synchronsprechern den Job wegnehmen wird. Dafür ist sie noch zu unausgereift und kommt beispielsweise mit der Darstellung von unterschiedlichen Emotionen wie Aufregung, Trauer oder Freude nicht zurecht und kann auch verschiedene Dialekte / Akzente nicht perfekt nachahmen oder nur für sehr spezifische Stimmen. Des Weiteren hat ein berühmter Name als Synchronsprecher aus Marketingsicht auch mehr Einfluss als ein „KI-Sprecher" und kann dadurch wohl mehr Zuschauer anziehen. Auch bei bspw. Musical-Produktionen gibt es Probleme, da Gesang aktuell noch nicht perfekt nachgeahmt werden kann, zumindest nicht von den Systemen, die auch die Sprechsynthese unterstützen. Hier wären wieder verschiedene Systeme notwendig, die in ihrer Anschaffung und in ihrem Training recht kosten- und ressourcenintensiv werden. Und über all dem steht natürlich auch ein Konflikt mit dem persönlichen Datenschutz eines jeden, denn die unerlaubte Verwendung von Samples einer Person zum Training des Modells ist mehr als nur problematisch. Für kleinere Produktionen, wie bspw. meiner Pottwal-Dokumentation kommt diese Technologie allerdings sehr gelegen.

5. Diskussion

Im Folgenden werden die Kommentare aus dem OPAL-Kurs aufgelistet und beantwortet:

> „Ich bin ziemlich beeindruckt von den englischen Demos (vor allem Cumberbatch). Weißt du, woran es liegt, dass die deutschen Stimmen im Gegensatz dazu so schlecht klingen / kaum zu verstehen sind? An welchem Layer in dem Prozess könnte das liegen?
>
> Ich würde mal vermuten, dass der Pretrained Transformer nur auf englischen Samples trainiert wurde, aber vielleicht hast du da noch mehr Infos."

Es ist so wie du sagst. James Betker hat das Modell ausschließlich auf englischen Datensätzen trainiert. Diese hat er selbst aus Audiobüchern und Podcasts, die er im Internet gesammelt hat, zusammengebaut. Der Datensatz umfasste am Ende ca. 49.000 Stunden an bereinigten Audioclips. Dadurch hat der CLVP entsprechende Schwierigkeiten qualitativ hochwertige Sprachproben zu erzeugen. Auch das Diffusionsmodell hat dadurch Probleme dann präzise und natürliche Sprachspektrogramme für die deutsche Sprache zu erzeugen.

> „Ich habe mir das Repo kurz angeschaut und es gibt zwar im Repo eine Anleitung, die enthält aber wenig Info zum Trainieren von neuen Stimmen. Mich würde noch interessieren, wie viele Samples man braucht, damit eine Stimme gut wird, bzw. wie viel das Modell schon an Sprachwissen mitbringt. Du schreibst, dass die Vertonung von Text immer 4-6h gedauert hat. Das ist schon eher lange, oder? Mich würde auch interessieren, welcher der vier Schritte im Prozess am aufwändigsten ist."

Es gibt im Github-Repository tatsächlich eine sehr genaue Anleitung wie man eigene Stimmen einbinden kann. Ein paar Anforderungen an die bereitzustellenden Samples hatte ich auch in meiner Anleitung zusammengefasst. Da die Anleitung im Repo evtl. nicht so leicht zu finden ist, habe ich den Link dazu hier mal angehängt.
Die Generierung hat unter anderem sehr lang gedauert, da ich das Modell bei mir auf der CPU laufen lassen musste, da ich Probleme damit hatte das mit CUDA zum Laufen zu bringen. Der Schritt der bei der Generierung immer am längsten gedauert hatte, war der erste Schritt also die Generierung von vielen verschiedenen Sprachsequenzen auf der Basis der bereitgestellten Samples. Da aber in allen Schritten sehr viele Berechnungen durchgeführt werden, bspw. auch beim iterativen Entfernen des Rauschens durch das Diffusionsmodell (3. Schritt), ist der Prozess an sich natürlich sehr umfangreich.

> „Ich persönlich finde Sprachsynthese die realistische Ergebnisse liefert, sehr befremdlich, kann mir aber vorstellen, dass sie für kleinere Projekte ohne großes Budget ein sehr hilfreiches Tool sein kann. Auch finde ich sehr interessant, dass hier Vocoder Verwendung finden, ich kannte diese bisher nur für einen charakteristischen Klang in elektronischer Musik. Mich würde auch interessieren, was genau an dem Prozess der Synthese dafür sorgt, dass sie 4-6h dauert."

Warum der Prozess so lange dauert, habe ich bereits im vorigen Kommentar näher erläutert. Zum Thema mit den Vocodern möchte ich sagen, dass Vocoder in beiden Fällen dafür verwendet werden, um Frequenzbänder (bzw. hier Sprachspektrogramme) zu analysieren und die

Signalfolgen zu modifizieren, um einen bestimmten (künstlerischen) Effekt zu erzeugen oder auch wie hier die Natürlichkeit und Verständlichkeit der erzeugten Sprache zu verbessern.

„Mich würde interessieren, ob du auf Basis der Repositorys etwas über die Datenmenge erfahren konntest, also wie groß die Anzahl an Trainingsdaten ist und ob hier vielleicht auch selbst mit synthetischen Daten trainiert wird. Das Ganze hat ja auch ein Stück weit einen ethischen Aspekt."

"I independently built an extended TTS dataset composed of audiobooks and podcasts scraped from the web. This data was split on 500ms silences, and any audio clip between 5-20 seconds was kept. I then fed the resulting clips through a pipeline of classifiers that I trained which remove any audio with background noise, music, poor quality (such as phone calls), multiple voices speaking at once and reverb. Due to disk space limitations, I was forced to limit the amount of scraping. The end result was 49,000 hours of cleaned audio clips.

I transcribed this dataset using a wav2vec2-large model. I personally fine-tuned this model to predict punctuation, as quotation marks, commas and exclamation marks are important for the purposes of generating speech but are not generally included in the training of speech recognition models. Fine-tuning was performed on LibriTTS and HiFiTTS and the pretrained model weights and transcription scripts can be found here"

Diesen Abschnitt habe ich eins zu eins aus dem zugrundeliegenden Paper entnommen und ist unter „A Extended Dataset Collection" zu finden. Das Paper kann hier nachgelesen werden. Wie man lesen kann, wurde hier nicht mit selbstsynthetisierten Daten trainiert, sondern mit Hörbüchern und Podcasts aus dem Internet. Ob dafür die Rechte überall vorhanden sind, kann ich dir aber nicht sagen.

„Ich finde sehr gut, dass du in deinem Diskussions-Abschnitt auf die gesellschaftlichen Risiken dieser Technologie eingegangen bist. Ich persönlich sehe da auch große Herausforderungen, die geregelt werden müssen.
Ansonsten finde ich auch sehr interessant, dass eine englische, männliche Stimme am besten klappt. Das wird sicherlich an den Trainingsdaten liegen."

Tatsächlich gibt es auch einige Beispiele aus dem Repo mit weiblichen Stimmen, die sehr gut funktioniert haben, wie bspw. die von Angelina Jolie oder auch von Emma Stone gibt es Beispiele im Originalrepository. Es kann daher auch gut sein, dass die Stimmen, die ich ausgewählt habe einfach nicht geeignet waren oder die Trainingsdaten nicht ausreichend bzw. qualitativ nicht hochwertig genug waren, um daraus künstliche Stimmen zu synthetisieren. Warum die Synthese mit englischen Stimmen wesentlich besser funktioniert, habe ich in einem vorigen Kommentar bereits erklärt.

„Sehr spannendes Thema, mich würde interessieren, ob man mit einem großen Datensatz bestehend aus deutschen Samples, ein gutes Endergebnis bekommen würde. Bzw. du sagtest, dass die Texte trotz deutschen Testsamples eher Englisch ausgesprochen wurden. Denkst du, ein größerer Datensatz mit längerer Trainingszeit hätte dieses Problem gelöst?"

Kurzum, ja. Die deutsche Sprache hat eine andere Phonetik als die englische Sprache. Dadurch hat der CLVP bspw. Probleme wirklich geeignete Stimmkandidaten auszuwählen. Das Training auf mehr deutschsprachigen Daten hätte natürlich die Ergebnisse verbessert.

„Guter Bericht. Alles verständlich und präzise erklärt. Interessieren würde mich, wie das Problem von fehlenden Emotionen in Stimmen momentan angegangen wird bzw. was es da an Entwicklungen und Forschung gibt. Ich tippe mal, dass die Vertonungszeit sehr von der Hardware abhängt. Wie schneidet denn dein verwendetes Verfahren gegenüber anderen ab oder gibt es nicht wirklich sinnvolle Alternativen?“

Modelle wie der Autoregressive Decoder oder das Denoising Diffusion Probabilistic Modell können durch den Einsatz von Sprachkonditionierungseingaben verbessert werden, die es dem Modell ermöglichen, Vokaleigenschaften wie Ton und Prosodie (komplexe Spracheigenschaften oberhalb der Lautebene, z.B. die Intonation, die Satzmelodie, das Sprechtempo, der Sprechrhythmus und Pausen) zu inferieren. Dies reduziert den Suchraum möglicher Sprachoutputs und verbessert die Natürlichkeit bzw. Emotionalität der erzeugten Ergebnisse. Es gibt viele andere Ansätze für TTS-Synthese, z.B.

- WaveNet (von DeepMind)
  basiert vollständig auf Convolutional Neural Networks (CNN), kann extrem natürliche Sprachwellen erzeugen, fordert aber auch sehr hohen Rechenaufwand
- Tacotron und Tacotron 2
  kombinieren Recurrent Neural Networks und CNNs zur Erzeugung von Mel-Spektrogrammen, die anschließend von einem Vocoder wie WaveNet in Sprachwellenformen umgewandelt werden.
- Variational Autoencoders, GAN-TTS (Generative Adversarial Networks), …

Alle haben ihre eigenen Vorteile und Nachteile. Da die Vertonungszeit stark von der Hardware abhängig ist, lässt sich die Vertonungszeit nur miteinander vergleichen, wenn man exakt die gleiche Hardware für die verschiedenen Modelle verwendet. Das konnte ich nur leider nicht testen und es wäre auch außerhalb des Rahmens dieses Projektes. Kurzum lässt sich aber sagen, dass leistungsfähigere Hardware schnellere Inferenzzeiten ermöglicht und damit auch kürzere Vertonungszeiten.

„Ich kann mich Max nur anschließen, ich bin ebenfalls überrascht, dass die Synthese so lange dauert bei, wenn man bedenkt dass APIs (zB elevenlabs) dies inzwischen in Sekunden können. Bezieht sich die Zeit von 4-6h auch auf das Erzeugen des Gesagten oder nur die Erzeugung dieser Voice? Bei den fantastischen Ergbnissen im Kontext der Pottwal-Doku, könnte man die lange Wartezeit durchaus in Kauf nehmen, wenn dabei die eigentliche Erzeugung der Audio nicht mehr so lange dauert.“

Die Synthesezeit von 4-6 Stunden bei der Verwendung von TorToise bezieht sich auf den gesamten Prozess der Audioerzeugung, nicht nur auf die Erzeugung der Stimme. Wenn man das Modell auf einer guten GPU laufen lassen kann, kann sich die Dauer natürlich nochmal erheblich reduzieren. Im Repository steht geschrieben, dass das Modell mittlerweile so optimiert werden konnte, dass die Generierung einen RTF (Real-Time-Factor) von 025 – 0,3 auf einem 4GB VRAM erreichen kann. In einem Beispiel heißt das, dass die Generierung von 2 bis 3 Sätzen nicht einmal eine Minute dauert. Und selbst auf seinem Ursprungsstadium hatte die Generierung eines Satzes mit einer

K80 ca. 2min gedauert. Wenn man also die geeignete Hardware und ein funktionierendes Setup hat, kann sich diese Generierungszeit massiv verkürzen, wodurch es wesentlich marktfähiger ist.

6. Nachsatz

Ich habe alle meine zur Synthese verwendeten Daten und die Ergebnisse für die hier genannten aber auch für zwei weitere fehlgeschlagene Experimente in einem Github-Repository hochgeladen. Dieses ist hier erreichbar.