EN

Musical Voice Synthesis at the Midpoint: Where Text Meets Sound

Dr. Olivier Pasquet
Goldsmiths, University of London
o.pasquet@gold.ac.uk

ABSTRACT

This research explores the application of language models and machine-learning techniques to generate musical voices with personality and control. By utilizing autoregressive transformers, specifically the Bark model, tokens are generated from input text to produce a unique invented and undefinable language.
Composition is being made between text and sound using various control techniques, including tokens repetition and windowing, Lempel-Ziv-Welch compression, and token clustering from acoustic feature extraction, to regulate the output voice's granularity, intelligibility, and meaning. A recursive generation system using token analysis is also introduced, allowing for the creation of a large series of interrelated voices.
The research is used in various artistic applications, including music remixing and theater productions. It explores other forms of expressive voices and storytelling seamlessly lying right in the middle between text and sound.

1. INTRODUCTION

Many have previously played with Text-To-Speech (TTS) synthesis engines by typing nonsensical text as input. We even built loops and random texts, which we then used to create thousands of audio files for post-processing in previous productions using synthetic voice. The results were very satisfying, but they had a tendency to feel banal unless we introduced additional musical effects later on. We never seamlessly balanced between text and sound; it was inherently a back-and-forth process between those two elements.

This paper addresses a key challenge by exploring various concepts and techniques to produce genuine artificial vocal techniques, voices, and abstractions that blur the line between text and pure music. By diving deep into a TTS voice synthesis engine, we can operate at the intersection of text and sound, where this unique fusion takes shape. We showcase how this approach integrates seamlessly into a broader compositional workflow and ultimately introduce five creative hacks specifically designed for Suno's system, *Bark*.

Copyright: ©2025 Dr. Olivier Pasquet et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License 3.0 Unported, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

2. VOICE GENERATION WITH PERSONALITY

2.1 Artificial "musical" voices

Georges Aperghis's work obviously influenced our research. His music-theatre pieces defy categorization. But it belongs to a kind of vocal composition relying heavily on a virtuosic manipulation of "phonemes", marked by rapid tempos, repetitive patterns, and accumulative techniques, all of which generate intense rhythmic energy [1]. [1] The creation of an "imaginary language" gives rise to a soundscape that is both ambiguous and playfully humorous, evoking the illusion of communication while remaining music. This blurs the line between linguistic expression and musical composition.

This blurred line can usually be explored and used with symbolic text and signal processing separately. However, there have been exceptions merging those steps such as Sprechgesang or Sprechstimme's expressionist musical vocal techniques. They lie in the middle but also follow the chain of being first defined symbolically and then interpreted by vocalists.

Hip-hop and R&B have continually pushed the boundaries of vocal expression through various techniques that blur the line between literalism and abstraction [2]. Notable vocal techniques are added to audio techniques such as Auto-tune [3, 4]. Since the late 1990s, Auto-tune has evolved from being merely a vocal correction tool into a cultural phenomenon. This effect is based on re-synthesis and can be extended as an artificial voice controlled by voice. This instrument is able to transform both the voice itself and the meaning it conveys.

2.2 Search for a synthesis with personality

Voice synthesis controlled by a Large Language Model (LLM) allows for generating a wide range of text and vocal techniques that are different when asking a performer. Depending on the models and used techniques, synthesis can bring a wide range of variability, subversion and inspirations. Such synthesis allows emotional detachment, gender, and neutrality that we can hybridize at will.

Moreover, it allows composing at the exact place of the blurred line between literalism and abstraction; and between symbolic text and signal.

However, most synthesis engines' quality has become too good to be mere instruments. Their lack of glitches and inconsistency does not enhance the creativity of the tool. Moreover, they significantly lose character, and the voice

[1] "Phonemes" does not refer to proper linguistics but rather portions of words or voice techniques.

is far less creative than that of a real actor, for instance. Finally, it is challenging to compose using voice synthesis without an architecture that we can fully control, avoiding babbling effects too closely tied to early-2020s aesthetics.

3. WORKFLOW

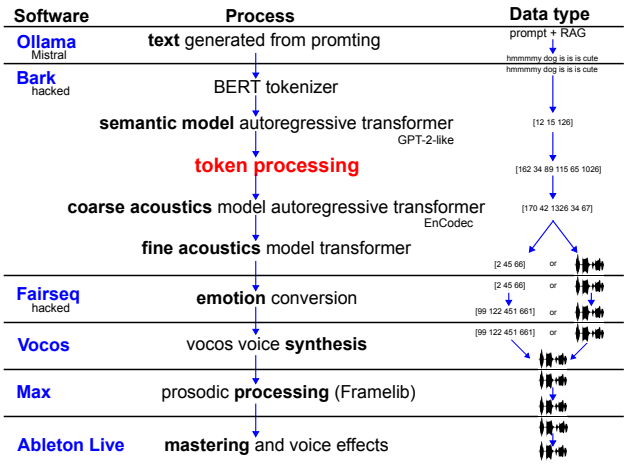

Figure 1. Proposed workflow with variables that can be utilized for musical purposes. This paper will focus specifically on the *Bark* part, particularly the red section, which has been demonstrated to be the most effective and expressive.

We propose a workflow offering control at each stage, from the initial text generation to the final voice (fig. 1):

- We start generating text using LLMs in Ollama, making various refinements, segmenting into units and loops [5, 6].
- We generate a series of audio tokens employing a voice synthesis engine named *Bark*. [2]
- We convert them to directly utilize the FairSeq library for expression conversion [7, 8]. [3]
- The final synthesis is done using the Vocos neural vocoder and upscaled to the right sampling rate. [4]
- The outcome is sent to Max for prosodic processing and Ableton Live for mastering and voice doubling.

We will here focus on one part of the voice synthesis although all components are interdependent and sometimes necessarily influenced by one another. We then only concentrate on the red section called token processing shown in fig. 1. This processes *semantic token* that lie exactly where we want: in the middle between text and audio generation.

4. HACKED BARK

4.1 Adapting Bark to composition

We decided to start with an adapted version of *Bark*, a text-prompted Generative Pre-trained Transformer-style (GPT-style) model that takes creative liberties in its generation. Suno's program, from 2023, does not offer the best sound quality but this has no impact on the overall quality of our system, as we employ numerous other processes and syntheses afterward. Moreover, the step using Max at the end of the workflow is based on re-synthesis using *FrameLib* [9, 10]. [5] It significantly alters voice qualities for aesthetic purposes.

Bark is made of a series of auto-regressive transformers using a semantic model, a coarse acoustics model, and a fine acoustics model:

- The *fine acoustics model* takes as input predicted tokens generated from the coarse model and iteratively predicts tokens ready for the audio synthesis. The use of EnCodec neural codec permits coding and hooking *Bark* to other libraries [11]. [6]
- The *coarse acoustic model* is a GPT-2-style causal transformer converting semantic tokens into coarse acoustic ones.
- The *semantic model* is also a GPT-2-like causal auto-regressive transformer model with a language modeling head on top. It takes in tokenized text (from a BERT tokenizer) as input and then predicts the semantic tokens that encode the audio to be generated. This part is the most important for speaker's identity. We can here add prompts that will most define speakers' personality traits with their intonation and prosodic patterns.

Primarily working with tokens from the *semantic model* allowed us to maintain our original intention of balancing composition between text and audio.

4.2 *Hack 1*: Variable token windowing

Bark's architecture is powerful for creativity thanks to its GPT architecture extending beyond only voice. Its models encompasses a wide variety of nonverbal communications like laughing, sighing, crying, and other surprises that can be called by prompting depending on the model used. However, this strength also brings the weakness of outputs quickly getting unpredictable if not properly inferred. The maximum length of audio in *Bark* is 756 semantic tokens, equivalent to approximately 15 seconds. This is due to the model's context window size being capped at 1024, similar to how text language models have limited context sizes today (e.g. 4096). It is worth noting that if *Bark* utilized relative positioning instead of absolute positioning, it might have been possible to achieve larger context sizes, such as 2048 or 4096. However, currently, we have not found techniques for achieving this with absolute positioning.

Instead, we created a system to play with variable windows of tokens allowing the control of sound granularity and thus voice intelligibility. It aligns with the infinitesimal rhythmical aesthetic we envisioned in our music. However, using a set of random tokens at this stage rapidly disrupts the model's predictive capabilities, typically resulting in a degenerate output: a converged, monotonous pitch with filtering and noise (fig. 2). The choice of variable window size and the randomness of token sequences defines how much prediction there will be.

[2] Suno's initial Bark lib: https://github.com/suno-ai/bark
[3] Fairseq lib: https://github.com/facebookresearch/fairseq
[4] Vocos lib: https://github.com/gemelo-ai/vocos
[5] FrameLib: https://github.com/AlexHarker/FrameLib
[6] EnCodec codec: https://github.com/facebookresearch/encodec

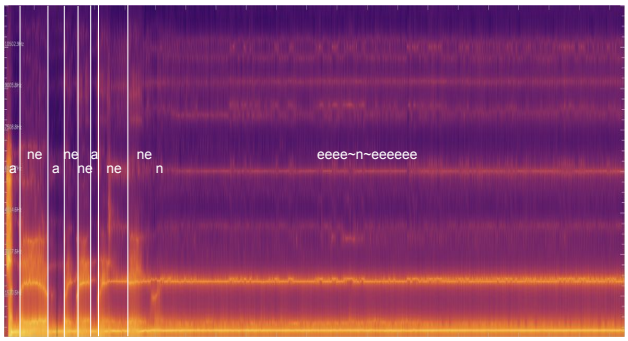

Figure 2. LPC spectrogram showing the predictive limits of the model. Using low temperature makes it diverge when not using unconventional sequences of tokens or when asking for a longer duration than what the system was designed for. We see here that it starts with the intended text then gets into a pitched loop. This inherent characteristic can be transformed into artistic control.

4.3 *Hack 2*: Control of noise variables

Each of those three layers has the following standard controls seen in such models. These include temperature, Top-p, and Top-k controls:

- *Temperature* setting governs the degree of randomness in word selection during text generation. Lower temperatures yield more predictable and consistent outputs, whereas higher temperatures introduce greater freedom and creativity, albeit at the cost of consistency.
- *Top p* setting determines the number of probable words considered by the model. Higher values enable the model to examine a broader range of possibilities, including less likely words, resulting in more diverse generated text.
- Adjusting the *Top k* setting influences response repetitiveness and complexity, notably in vocabulary and phrasing.

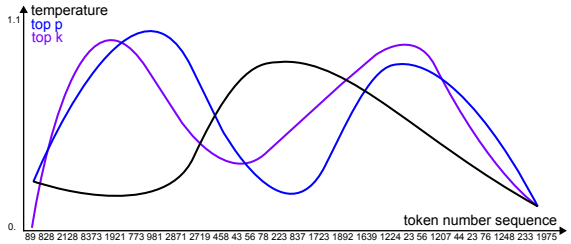

Figure 3. An interpolated break-point function (BPF) has demonstrated effectiveness in controlling temperature, top p, and top k, across token sequences, enabling articulation and temporal variations; simple and perceptively efficient. As often, using 3d simplex noise to correlate those parameters together resulted in greater consistency.

We have implemented a method to control these values using break-point functions (BPF), which enable us to regulate amounts of randomness within each sequence of given tokens (fig. 3). This helps play with the varying intensity of expression throughout a sentence. We added several random and quantizing engines akin to those found in Ableton Live's *Beat Repeat* feature [12]. Randomly repeating tokens this way sounds very much like granular synthesis. But employing such transformers with temperatures as described earlier yields a more dynamic and human-like output. It offers greater control and expressiveness compared to only directly concatenating grains. The articulation between "phonemes" aims for naturalness and may sometimes evoke the additive interpolation taste found in *Diphone* [13, 14].

Randomness plays a valuable role in facilitating serendipitous composition experiments, although it here falls short of enabling nuanced musical sequence composition. The best way to increase control over discreet token sequence generation is simply to use markovian techniques.

5. TOKEN ENGINEERING

In order to get interesting results, we have to generate subsequently more sentences than needed, thereby producing a large set of tokens that can then be organized using probabilities. We therefore use Ollama's models to produce a set of paraphrased sentences that share a sufficient number of common words, "phonemes", and meaning. [7] This consistency enables us to achieve more meaningful textual output and brings us closer to our original goal of creating an invented language. The fact our current "language" and voice synthesis system makes use of probabilities and neural networks together is interesting from a historical point of view [15].

5.1 *Hack 3*: LZ decoding of token

We subsequently employed the multi-scalar Lempel-Ziv-Welch (LZ) compression, acknowledging that windowing was playing a critical role (cf. 4.2). The LZ encoding algorithm compresses sequences of variable-sized tokens by mapping them to a dictionary and emitting references to that dictionary as a string [16].

We can subsequently decode arbitrary-length sequences from the string, generating concatenated sequences indefinitely [17]. Using LZ with weighted probability onto the string has been demonstrated to be the most effective method for regulating the level of meaning and abstraction in our voices.

LZ performs optimally on data containing repetitive patterns, making it well-suited for paraphrased sentences. Its multi-scalar nature allows us to choose specific sequence lengths from the LZ dictionary and play with our model's context window, as described earlier (fig. 4).

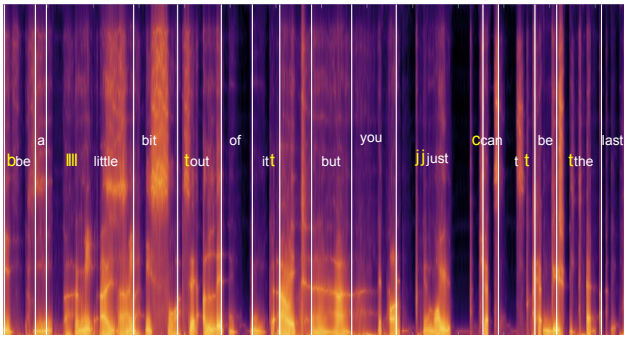

Figure 4. Example use: generation of semi-words and semi-sentences in English using an LZ sequence (white) overlapped with accidental repetitions of particular sets of semantic token (yellow).

[7] https://ollama.com - We mostly prompt Mistral models and widely use Retrieval-Augmented Generation (RAG) technique.

5.2 *Hack 4*: Acoustic properties sequencing

We now aim to achieve token segmentation enabling us to extract specific sounds or patterns from within the model's output. The inherent unpredictability of autoregressive transformers, particularly when using models not trained by ourselves, necessitates a thorough analysis of the model's behavior: *We query the model before using it*. To minimize uncertainty, we employ a deterministic strategy by using low-temperature settings and a unique seed number (fig. 5).

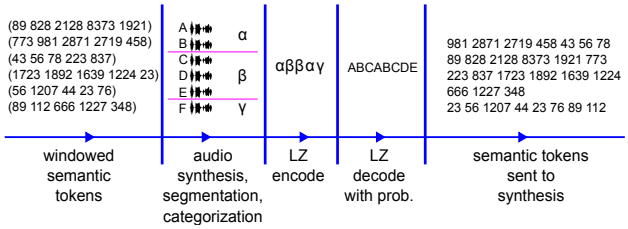

Figure 5. Audio analysis of *Bark* model's behavior using *Flucoma*'s descriptors: We query the model before using it. To minimize uncertainty, we employ a deterministic strategy by using low-temperature settings and a unique seed number.

We initiate our analysis with a comprehensive dataset of paraphrased sentences, which we then subject to sound-descriptors extraction using the *Flucoma* Python library and simple house-made k-means clustering [18, 19]. The multilingual feature of *Bark* models allows for even greater timbral variety. Specifically, we analyze pitch sequences and Mel-Frequency Cepstral Coefficients (MFCCs) derived from the output, enabling us to perform pitch-based or timbre-based clustering of tokens. [8] This approach proves highly effective for token sequence segmentation (fig. 6). We then use the simple but effective weighted LZ method described earlier to query token and infer then from the model. We can eventually algorithmically compose with recurrences a sequence containing a succession of voice descriptors such as vowels, fricatives, nasals, and transients (fig. 7).

We remain focused on the initial idea of using text-to-speech synthesis for the moment. However, analyzing sounds derived from token chains can also guide inference and converge on diverse sound targets, akin to concatenative synthesis [20].

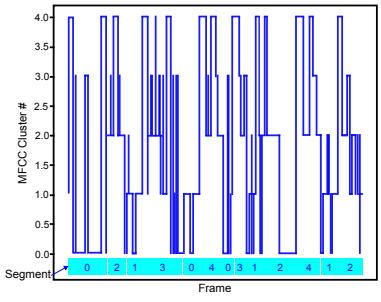

Figure 6. Simple k-mean clustering of MFCC segments before using them as token with our Markovian methods. We have here zoomed into a larger dataset to enhance visibility. Nevertheless, it is evident that the quantity of clusters has a significant impact on the intelligibility of the synthesized voice.

[8] https://www.flucoma.org and Python Flucoma

5.3 *Hack 5*: Long generation using fine-tuning

We observed that the maximum length of audio in *Bark* was 756 semantic tokens, equivalent to approximately 15 seconds. In contrast, the coarse acoustic model transformer has no such limit. We then recursively feed the latter with sequences of semantic tokens and ensure that we can replicate the same sound characteristics as for previous iterations. We first fine-tuned the coarse acoustic model. But we had better results fine-tuning all the three models using a parameter we call *history-prompt*. Tuning all three models together at the same time is also used to target the personality of a specific actor or singer (deepfake).

We designed a method that dynamically controls long sets of tokens by guiding inferences from latent space to a desired target. The path is made by controlling the probability weights of previous tokens, selecting appropriate history-prompt and manipulating temperature to enhance novelty and propositions. We can also use both textual prompts and the sound-descriptor method described above to automate the evolution of long, composed sequences:

1. We iterate our system while having a relatively high temperature in order to widen predictions.
2. When the sound characteristics reach a satisfactory target, we use the results as history-prompt and keep the same seed numbers to re-inject them in the subsequent steps.
3. We use seed and history-prompt to generate new versions with low temperature this time.
4. We gradually increase the temperature, step by step, until reaching a new target defined automatically by the sound-descriptors' analysis.
5. We repeat from point 2.

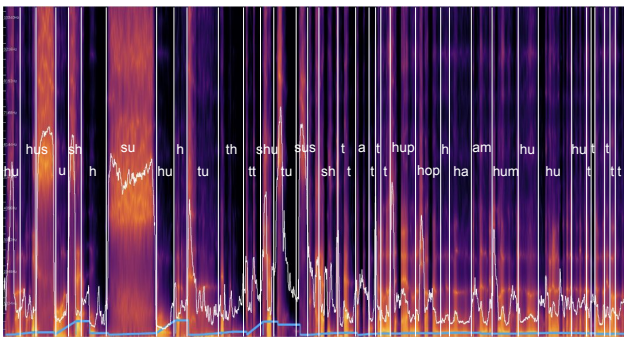

Figure 7. LPC spectrogram showing a generation of `hu`, `tu`, `shu`... Temperature introduced into the coarse acoustic model adds greater variety. It also avoids repetition in the sequence by adding more phonemes related to those given by the input tokens. Initially articulated as voice, those segments sound very much like "electro-acoustic music's type" articulated transitions. A pitch curve in blue and centroid in white have been added over to better visualize intensity and inflections in this phrase. Notice the end converging toward only one pitch, in blue.

Each of the versions generated above can be used independently or together to create polyphony. We have also used this interplay to generate hip-hop stems by trying to control points of convergence at specific moments in a song. By selecting an optimal size for the windowing and the group of tokens at the input of the system, this approach produces an indescribable assemblage where the dramatic intensity of the voice, and the genre, dissolve into enigmatic meanderings, creating an unsettling disorientation. This approach

highlights the power of using the voice from a dramatic point of view.

6. FUTURE AND MUSICAL APPLICATIONS

We used these techniques in the production of remixes for popular music singers with the authorization of Warner Music. Those methods are also going to be widely used for a musical English and theatrical French version of John Fosse's play *And We'll Never Be Parted* premiered at T2G National Theatre in September 2025.
The part using GPT-2-like transformers is satisfying for our needs. However, training a large personal Bark model has proved difficult, if not impossible. In the near future, we will simplify the workflow using fewer external libraries and easily port the whole system to FairSeq 2. We want to increase variety and style using Low-Rank Adaptation (LoRA) onto much larger models [21]. We should also be able to merge those models as easily as we do in graphic stable diffusion tools.

7. CONCLUSION

The integration of transformer-based TTS synthesis and machine learning into production permits a creative expression between text, sound, literalism, and abstraction. We can generate long unique vocal sequences using token engineering controlled by sound-descriptors, and fine-tuning models. This research shows the utility of using simple concepts to achieve intuitive controls. Utilizing transformers might initially seem counterproductive to novelty and creativity. However, integrating parametric processes enables unexpected textual and sonic surprises, distinct from those derived solely from acting. We can seamlessly integrate text and music together in a personalized and uniquely crafted manner, ensuring creative independence and originality without relying entirely on externally controlled online platforms or products.

A Jupyter notebook with all the sequences and audio examples is available here on GitHub.

8. REFERENCES

[1] M. Woo, "*L'interprétation musicale des phonèmes, des gestes et des images dans Machinations de Georges Aperghis*," 2011.

[2] P. Shapiro and I. Lee, "*Modulations: a history of electronic music: throbbing words on sound*," 2000.

[3] A. Gayraud, R. Mackay, D. Miller, and N. Power, "*Dialectic of Pop*," 2019.

[4] *Rubin, Rick and Strauss, Neil*, "The Creative Act: A Way of Being," 2023.

[5] "*DeepRapper: Neural Rap Generation with Rhyme and Rhythm Modeling*."

[6] C. Durt and T. Fuchs, "*Large Language Models and the Patterns of Human Language Use*," 2024.

[7] F. Kreuk, A. Polyak, J. Copet, E. Kharitonov, T.-A. Nguyen, M. Rivière, W.-N. Hsu, A. Mohamed, E. Dupoux, and Y. Adi, "*Textless Speech Emotion Conversion using Discrete and Decomposed Representations*," 2022.

[8] L. Barrault, P.-A. Duquenne, M. Elbayad, and A. Kozhevnikov, "*Large Concept Models - Language Modeling in a Sentence Representation Space*," 2024.

[9] A. Harker, "*FrameLib: Audio DSP using Frames of Arbitrary Length and Timing*," 2016.

[10] N. Schnell and D. Schwarz, "*Gabor, Multi-representation Real-Time Analysis/Synarticle*," 2005.

[11] N. Obin, "*Cries and Whispers - Classification of Vocal Effort in Expressive Speech*," 2012.

[12] J. Kammerer, "*Unleashing Creativity with Ableton's Beat Repeat: A Comprehensive Guide*," 2014.

[13] G. Loizillon, "*Diphone Studio*," 1999.

[14] J. Bachan, "*Efficient Diphone Database Creation for MBROLA, a Multilingual Speech Synarticleer*," 2010.

[15] R. Shwartz-Ziv and Y. LeCun, "*To Compress or Not to Compress- Self-Supervised Learning and Information Theory: A Review*," 2023.

[16] J. Ziv and A. Lempel, "*Compression of Individual Sequences via Variable-Rate Coding*," 1978.

[17] O. Lartillot, "*OpenMusic LZ 2.2 Library*," 2001.

[18] P. A. Tremblay, O. Green, G. Roma, and A. Harker, "Fr*om Collections to Corpora: Exploring Sounds through Fluid Decomposition*," 2019.

[19] T. Moore, J. Bradbury, and P. A. Trembly, "*FluCoMa for Pedagogues*," 2022.

[20] B. Hackbarth, N. Schnell, P. Esling, and D. Schwarz, "*Composing Morphology: Concatenative Synarticle as an Intuitive Medium for Prescribing Sound in Time*," 2013.

[21] C. V. Nguyen, X. Shen, R. Aponte, Y. Xia, S. Basu, Z. Hu, J. Chen, M. Parmar, S. Kunapuli, J. Barrow, J. Wu, A. Singh, Y. Wang, J. Gu, F. Dernoncourt, N. K. Ahmed, N. Lipka, R. Zhang, X. Chen, T. Yu, S. Kim, H. Deilamsalehy, N. Park, M. Rimer, Z. Zhang, H. Yang, R. A. Rossi, and T. H. Nguyen, "*A Survey of Small Language Models*," 2024.

DE

Musikalische Stimmsynthese in der Mitte: Wo Text auf Ton trifft

Dr. Olivier Pasquet
Goldsmiths, University of London
o.pasquet@gold.ac.uk

ABSTRACT

Diese Forschungsarbeit untersucht die Anwendung von Sprachmodellen und Techniken des maschinellen Lernens, um musi- kalische Stimmen mit Persönlichkeit und Kontrolle zu erzeugen. Durch die Verwendung von autoregressiven Transformers, insbesondere des Bark-Modells, werden aus dem Eingabetext Token generiert, die eine einzigartige erfundene und undefinierbare Sprache erzeugen.
Die Komposition zwischen Text und Ton erfolgt unter Verwendung verschiedener Kontrolltechniken, einschließlich Token-Wiederholung und Fensterung, Lempel-Ziv-Welch Kompression und Token-Clustering aus der Extraktion akustischer Merkmale, um die Granularität, Verständlichkeit und Bedeutung der ausgegebenen Stimme zu regulieren. Außerdem wird ein rekursives Generierungssystem vorgestellt, das die Token-Analyse nutzt und die Erstellung einer großen Reihe von miteinander verbundenen Stimmen ermöglicht.
Die Forschung wird in verschiedenen künstlerischen Anwendungen eingesetzt, darunter Musik-Remixing und Theaterproduktionen. Sie erforscht andere Formen ausdrucksstarker Stimmen und des Geschichtenerzählens, die nahtlos in der Mitte zwischen Text und Ton liegen.

1. INTRODUCTION

Viele haben bereits mit Text-To-Speech (TTS)-Synthese-Engines gespielt, indem sie unsinnigen Text als Eingabe eintippten. Wir haben sogar Loops und Zufallstexte erstellt, aus denen wir dann Tausende von Audiodateien für die Nachbearbeitung in früheren Produktionen mit synthetischer Stimme erstellt haben. Die Ergebnisse waren sehr befriedigend, aber sie wirkten eher banal, wenn wir nicht später zusätzliche musikalische Effekte einfügten. Wir haben nie nahtlos zwischen Text und Ton balanciert; es war von Natur aus ein Hin und Her zwischen diesen beiden Elementen.

Dieser Beitrag befasst sich mit einer zentralen Herausforderung, indem er verschiedene Konzepte und Techniken zur Erzeugung echter künstlicher Gesangstechniken, Stimmen und Abstraktionen untersucht, die die Grenze zwischen Text und reiner Musik verwischen. Indem wir tief in eine TTS-Stimmensynthese-Engine eintauchen, können wir an der Schnittstelle von Text und Klang arbeiten, wo diese einzigartige Fusion Gestalt annimmt. Wir zeigen, wie sich dieser Ansatz nahtlos in einen breiteren kompositorischen Arbeitsablauf einfügt, und stellen schließlich fünf kreative Hacks vor, die speziell für das Suno-System *Bark*.

Copyright: ©2025 Dr. Olivier Pasquet et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License 3.0 Unported, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

2. STIMMERZEUGUNG MIT PERSÖNLICHKEIT

2.1 Künstliche „musikalische“ Stimmen

Das Werk von Georges Aperghis hat unsere Forschung offensichtlich beeinflusst. Seine Musiktheaterstücke lassen sich nicht kategorisieren. Aber es gehört zu einer Art von Vokalkomposition, die sich stark auf eine virtuose Manipulation von Phonemen stützt, die durch schnelle Tempi, sich wiederholende Muster und akkumulative Techniken gekennzeichnet ist, die allesamt eine intensive rhythmische Energie erzeugen [1]. [1] Durch die Schaffung einer ïmaginären Spracheëntsteht eine Klanglandschaft, die sowohl mehrdeutig als auch spielerisch-humorvoll ist, die die Illusion von Kommunikation hervorruft und gleichzeitig Musik bleibt. Die Grenze zwischen sprachlichem Ausdruck und musikalischer Komposition verschwimmt dabei.

Diese unscharfe Grenze kann in der Regel mit symbolischem Text und Signalverarbeitung getrennt erforscht und genutzt werden. Es gibt jedoch Ausnahmen, bei denen diese Schritte zusammengeführt werden, wie z. B. beim Sprechgesang oder der Sprechstimme in der expressionistischen Vokalmusik. Sie liegen in der Mitte, folgen aber auch der Kette, dass sie zunächst symbolisch definiert und dann von den Sängern interpretiert werden.

Hip-Hop und R&B haben die Grenzen des stimmlichen Ausdrucks durch verschiedene Techniken, die die Grenze zwischen Wörtlichkeit und Abstraktion verschwimmen las- sen, immer weiter verschoben [2]. Bemerkenswerte Gesangstechniken kommen zu Audiotechniken wie Auto-tune hinzu [3, 4]. Seit den späten 1990er Jahren hat sich Auto-tune von einem reinen Stimmkorrekturwerkzeug zu einem kulturellen Phänomen entwickelt. Dieser Effekt basiert auf der Re-Synthese und kann zu einer künstlichen Stimme erweitert werden, die von der Stimme gesteuert wird. Dieses Instrument ist in der Lage, sowohl die Stimme selbst als auch die Bedeutung, die sie vermittelt, zu verändern.

2.2 Suche nach einer Synthese mit Persönlichkeit

Die von einem Large Language Model (LLM) gesteuerte Sprachsynthese ermöglicht die Erzeugung einer breiten

[1] „Phoneme“ bezieht sich nicht auf die eigentliche Linguistik, sondern eher auf Teile von Wörtern oder Stimmtechniken.

Pa- lette von Texten und Gesangstechniken, die sich unterscheiden, wenn man einen Interpreten fragt. Je nach den verwendeten Modellen und Techniken kann die Synthese ein breites Spektrum an Variabilität, Subversion und Inspirationen bieten. Eine solche Synthese ermöglicht emotionale Distanz, Gender und Neutralität, die wir nach Belieben hybridisieren können.

Darüber hinaus ermöglicht es das Komponieren genau an der unscharfen Grenze zwischen Wörtlichkeit und Abstraktion, zwischen symbolischem Text und Signal.

Die Qualität der meisten Synthese-Engines ist jedoch zu gut geworden, um nur Instrumente zu sein. Das Fehlen von Fehlern und Unstimmigkeiten trägt nicht zur Kreativität des Tools bei. Außerdem verlieren sie deutlich an Charakter, und die Stimme ist weit weniger kreativ als die eines echten Schauspielers beispielsweise. Und schließlich ist es eine Herausforderung, mit Hilfe der Sprachsynthese zu komponieren, ohne dass wir eine Architektur haben, die wir vollständig kontrollieren können, und die zu sehr an die Ästhetik der frühen 20er Jahre gebundene, plappernde Effekte vermeidet.

3. WORKFLOW

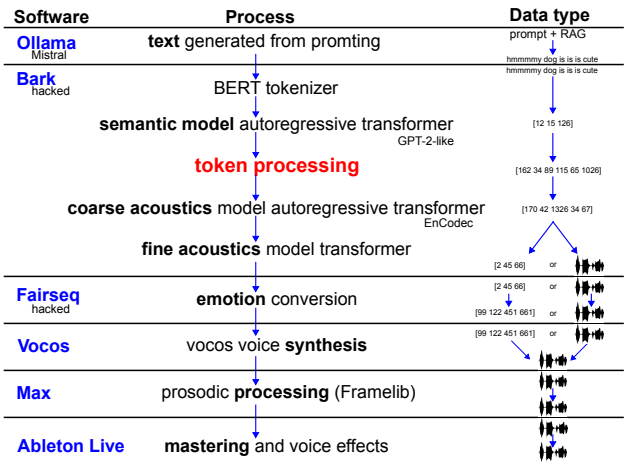

Figure 1. Vorgeschlagener Arbeitsablauf mit Variablen, die für musikalische Zwecke genutzt werden können. Dieser Beitrag wird sich speziell auf den TextitBark-Teil konzentrieren, insbesondere auf den roten Abschnitt, der sich als der effektivste und ausdrucksstärkste erwiesen hat.

Wir schlagen einen Arbeitsablauf vor, der in jeder Phase Kontrolle bietet, von der anfänglichen Texterstellung bis zur endgültigen Sprachausgabe (fig. 1):

- Wir beginnen mit der Generierung von Text unter Verwendung von LLMs in Ollama, nehmen verschiedene Verfeinerungen vor, segmentieren in Einheiten und Schleifen[5, 6].
- Wir erzeugen eine Reihe von Audio-Tokens mit Hilfe einer Sprachsynthese-Engine namens*Bark*. [2]
- Wir konvertieren sie, um die FairSeq-Bibliothek direkt für die Expressionskonvertierung zu nutzen [7, 8]. [3]
- Die endgültige Synthese erfolgt mit dem neuronalen Vocoder von Vocos und wird auf die richtige Abtastrate hochskaliert. [4]
- Das Ergebnis wird für die prosodische Bearbeitung an Max und für das Mastering und die Stimmverdopplung an Ableton Live gesendet.

Wir werden uns hier auf einen Teil der Sprachsynthese konzentrieren, obwohl alle Komponenten voneinander abhängig sind und sich manchmal zwangsläufig gegenseitig beeinflussen. Wir konzentrieren uns daher nur auf den roten Abschnitt namens token processing, der in Abb. 1. Dieser verarbeitet *semantische Token*, die genau dort liegen, wo wir sie haben wollen: in der Mitte zwischen Text- und Audiogenerierung.

4. HACKED BARK

4.1 Anpassung Bark an die Zusammensetzung

Wir haben uns entschieden, mit einer angepassten Version von *Bark* zu beginnen, einem textgesteuerten Modell im Stil von Generative Pre-trained Transformer (GPT), das sich bei der Erzeugung kreative Freiheiten nimmt. Das Programm von Suno aus dem Jahr 2023 bietet nicht die beste Klangqualität, aber das hat keinen Einfluss auf die Gesamtqualität unseres Systems, da wir zahlreiche andere Prozesse und Synthesen im Anschluss daran einsetzen. Außerdem basiert der Schritt mit Max am Ende des Arbeitsablaufs auf einer Re-Synthese mit *FrameLib* [9, 10]. [5] Sie verändert die Stimmqualität zu ästhetischen Zwecken erheblich.

Bark besteht aus einer Reihe von autoregressiven Transformers, die ein semantisches Modell, ein grobes Akustikmodell und ein feines Akustikmodell verwenden:

- Das *fine acoustics model* nimmt als Eingabe vorhergesagte Token, die vom groben Modell generiert wurden, und sagt iterativ Token voraus, die für die Audiosynthese bereit sind. Die Verwendung des neuronalen Codecs EnCodec ermöglicht die Kodierung und Anbindung von *Bark* an andere Bibliotheken [11]. [6]
- Das grobe akustische Modell ist ein kausaler Transformator im GPT-2-Stil, der semantische Token in grobe akustische Token umwandelt.
- Das semantische Modell ist ebenfalls ein GPT-2-ähnliches kausales, autoregressives Transformationsmodell mit einem Sprachmodellierungskopf an der Spitze. Es nimmt tokenisierten Text (von einem BERT-To- kenizer) als Eingabe auf und sagt dann die semantischen Token voraus, die das zu erzeugende Audio kodieren. Dieser Teil ist für die Identität des Sprechers am wichtigsten. Hier können wir Prompts hinzufügen, die die Persönlichkeitsmerkmale des Sprechers durch ihre Intonation und prosodischen Muster am besten definieren.

Indem wir hauptsächlich mit Tokens aus dem TextitSemantikmodell arbeiteten, konnten wir unsere ursprüngliche Absicht einer ausgewogenen Komposition zwischen Text und Audio beibehalten.

[2] Sunos anfängliche Bark lib: https://github.com/suno-ai/bark
[3] Fairseq lib: https://github.com/facebookresearch/fairseq
[4] Vocos lib: https://github.com/gemelo-ai/vocos
[5] FrameLib: https://github.com/AlexHarker/FrameLib
[6] EnCodec codec: https://github.com/facebookresearch/encodec

4.2 *Hack 1*: Variable Token-Fensterung

Die Architektur von *Bark* ist dank ihrer GPT-Architektur, die über die reine Stimme hinausgeht, sehr kreativ. Seine Modelle umfassen eine Vielzahl nonverbaler Kommunikationen wie Lachen, Seufzen, Weinen und andere Überraschungen, die je nach verwendetem Modell durch Eingabeaufforderungen aufgerufen werden können. Diese Stärke bringt jedoch auch die Schwäche mit sich, dass die Ausgaben schnell unvorhersehbar werden, wenn sie nicht richtig abgeleitet werden. Die maximale Länge des Audios in *Bark* beträgt 756 semantische Token, was etwa 15 Sekunden entspricht. Dies ist darauf zurückzuführen, dass die Größe des Kontextfensters des Modells auf 1024 begrenzt ist, ähnlich wie Text-Sprachmodelle heute begrenzte Kontextgrößen haben (z. B. 4096). Es ist erwähnenswert, dass es möglich gewesen wäre, größere Kontextgrößen wie 2048 oder 4096 zu erreichen, wenn *Bark* eine relative Positionierung anstelle einer absoluten Positionierung verwendet hätte. Derzeit haben wir jedoch keine Techniken gefunden, um dies mit absoluter Positionierung zu erreichen.

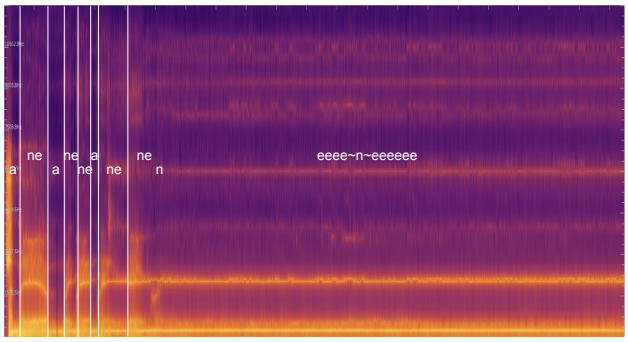

Figure 2. LPC-Spektrogramm, das die Vorhersagegrenzen des Modells zeigt. Die niedrige Temperatur führt dazu, dass das Modell abweicht, wenn keine unkonventionellen Token-Sequenzen verwendet werden oder wenn die Abfrage länger dauert als das System vorgesehen ist. Wir sehen hier, dass es mit dem beabsichtigten Text beginnt und dann in eine Tonhöhenschleife gerät. Diese inhärente Eigenschaft kann in künstlerische Kontrolle umgewandelt werden.

Stattdessen haben wir ein System entwickelt, das mit variablen Fenstern von Token spielt und die Kontrolle über die Klanggranularität und damit die Sprachverständlichkeit ermöglicht. Dies steht im Einklang mit der infinitesimalen rhythmischen Ästhetik, die wir uns für unsere Musik vorge- stellt haben. Die Verwendung eines Satzes zufälliger Token in dieser Phase stört jedoch schnell die Vorhersagefähigkeiten des Modells und führt in der Regel zu einer degenerierten Ausgabe: eine konvergierte, monotone Tonhöhe mit Filterung und Rauschen (fig. 2). Die Wahl der Größe des variablen Fensters und die Zufälligkeit der Token-Sequenzen bestimmen, wie viel Vorhersage es geben wird.

4.3 *Hack 2*: Kontrolle der Lärmvariablen

Jede dieser drei Ebenen verfügt über die folgenden Standardsteuerungen, die bei solchen Modellen üblich sind. Dazu gehören Temperatur, Top-p- und Top-k-Kontrollen:

- *Temperature* regelt den Grad der Zufälligkeit bei der Wortauswahl während der Texterstellung. Niedrigere Temperaturen führen zu vorhersehbaren und konsistenten Ergebnissen, während höhere Temperaturen mehr Freiheit und Kreativität ermöglichen, wenn auch auf Kosten der Konsistenz.
- Die Einstellung *Top p* bestimmt die Anzahl der wahrscheinlichen Wörter, die vom Modell berücksichtigt werden. Höhere Werte ermöglichen es dem Modell, ein breiteres Spektrum an Möglichkeiten zu untersuchen, einschließlich weniger wahrscheinlicher Wörter, was zu einem vielfältigeren generierten Text führt.
- Die Einstellung von *Top k* beeinflusst die Wiederholbarkeit und Komplexität der Antworten, insbesondere in Bezug auf Wortschatz und Phrasierung.

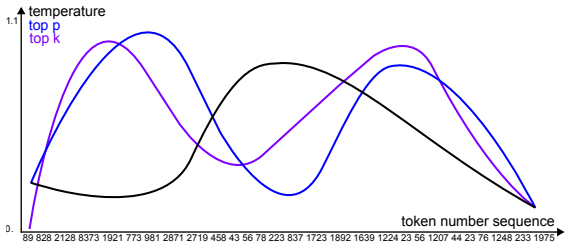

Figure 3. Eine interpolierte Bruchpunktfunktion (BPF) hat sich bei der Kontrolle der Temperatur, des oberen p und des oberen k über Token-Sequenzen hinweg als wirksam erwiesen, da sie Artikulation und zeitliche Variationen ermöglicht; sie ist einfach und wahrnehmungsmäßig effizient. Wie so oft führte die Verwendung von 3d-Simplex-Rauschen, um diese Parameter miteinander zu korrelieren, zu größerer Konsistenz.

Wir haben eine Methode zur Steuerung dieser Werte mit Hilfe von Break-Point-Funktionen (BPF) implementiert, die es uns ermöglichen, die Menge an Zufälligkeit innerhalb jeder Sequenz von gegebenen Token zu regulieren (Abb. 3). Auf diese Weise können wir mit der unterschiedlichen Intensität des Ausdrucks innerhalb eines Satzes spielen. Wir haben mehrere Zufalls- und Quantisierungs-Engines hinzugefügt, die denen der Funktion *Beat Repeat* von Ableton Live ähneln. [12]. Die zufällige Wiederholung von Token auf diese Weise klingt sehr ähnlich wie Granularsynthese. Der Einsatz solcher Transformers mit Temperaturen wie oben beschrieben führt jedoch zu einer dynamischeren und menschenähnlicheren Ausgabe. Im Vergleich zur direkten Verkettung von Grains bietet sie mehr Kontrolle und Ausdruckskraft. Die Artikulation zwischen „Phonemen“ zielt auf Natürlichkeit ab und kann manchmal an den Geschmack der additiven Interpolation erinnern, wie er in *Diphone* [13, 14].

Der Zufall spielt eine wertvolle Rolle bei der Erleichterung zufälliger Kompositionsexperimente, obwohl er hier nicht ausreicht, um eine nuancierte musikalische Sequenzkomposition zu ermöglichen. Die beste Möglichkeit, die Kontrolle über die Erzeugung diskreter Token-Sequenzen zu verbessern, ist die Verwendung markovianischer Techniken.

5. TOKEN-TECHNIK

Um interessante Ergebnisse zu erhalten, müssen wir in der Folge mehr Sätze generieren als nötig, wodurch eine große Menge von Token entsteht, die dann mit Hilfe von Wahrscheinlichkeiten organisiert werden können. Wir verwenden daher die Modelle von Ollama, um eine Menge umschriebener Sätze zu erzeugen, die eine ausreichende An-

zahl gemeinsamer Wörter, Phoneme und Bedeutungen aufweisen.[7] Diese Konsistenz ermöglicht uns eine aussagekräftigere Textausgabe und bringt uns unserem ursprünglichen Ziel, eine erfundene Sprache zu schaffen, näher. Die Tatsache, dass unser aktuelles System für die Zitat- und Sprachsynthese Wahrscheinlichkeiten und neuronale Netze zusammen verwendet, ist aus historischer Sicht interessant[15].

5.1 *Hack 3*: LZ-Dekodierung von Token

Anschließend haben wir die multiskalare Lempel-Ziv-Welch (LZ)-Komprimierung eingesetzt, wobei wir festgestellt haben, dass die Fensterung eine entscheidende Rolle spielt (cf. 4.2). Der LZ-Kodierungsalgorithmus komprimiert Sequenzen von Token variabler Größe, indem er sie auf ein Wörterbuch abbildet und Verweise auf dieses Wörterbuch als String ausgibt [16].

Anschließend können wir Sequenzen beliebiger Länge aus der Zeichenkette dekodieren und so unendlich viele verkettete Sequenzen erzeugen [17]. Die Verwendung von LZ mit gewichteter Wahrscheinlichkeit auf die Zeichenkette hat sich als die effektivste Methode zur Regulierung des Bedeutungs- und Abstraktionsniveaus in unseren Stimmen erwiesen.

LZ funktioniert optimal bei Daten, die sich wiederholende Muster enthalten, und eignet sich daher gut für umschreibende Sätze. Seine multiskalare Natur ermöglicht es uns, bestimmte Sequenzlängen aus dem LZ-Wörterbuch auszuwählen und mit dem Kontextfenster unseres Modells zu spielen, wie zuvor beschrieben (fig. 4).

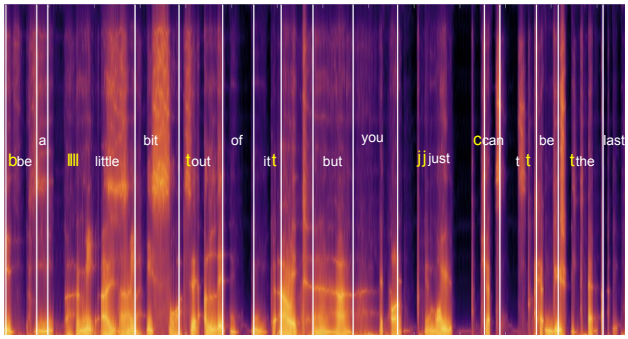

Figure 4. Anwendungsbeispiel: Generierung von Halbwörtern und Halbsätzen im Englischen unter Verwendung einer LZ-Sequenz (weiß), überlagert mit zufälligen Wiederholungen bestimmter Sätze von semantischen Token (gelb).

5.2 *Hack 4*: Sequenzierung der akustischen Eigenschaften

Unser Ziel ist es nun, eine Token-Segmentierung zu erreichen, die es uns ermöglicht, bestimmte Klänge oder Muster aus der Ausgabe des Modells zu extrahieren. Die inhärente Unvorhersehbarkeit von autoregressiven Transformers, insbesondere bei Verwendung von Modellen, die nicht von uns selbst trainiert wurden, erfordert eine gründliche Analyse des Modellverhaltens: Wir fragen das Modell ab, bevor wir es verwenden. Um die Unsicherheit zu minimieren, wenden wir eine deterministische Strategie an, indem wir niedrige Temperaturen und eine eindeutige Seed-Nummer verwenden (fig. 5).

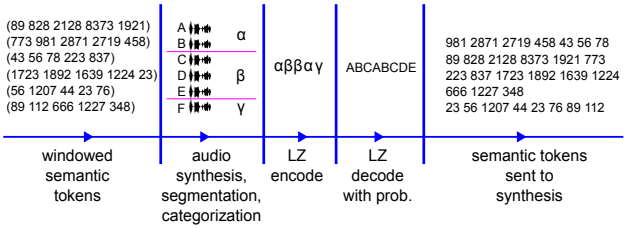

Figure 5. Audioanalyse des Verhaltens des Modells *Bark* unter Verwendung der Deskriptoren von *Flucoma*: Wir fragen das Modell ab, bevor wir es verwenden. Um Unsicherheiten zu minimieren, verwenden wir eine deterministische Strategie mit Niedrigtemperatur-Einstellungen und einer eindeutigen Startnummer.

Wir beginnen unsere Analyse mit einem umfassenden Datensatz paraphrasierter Sätze, die wir dann mit Hilfe der Python-Bibliothek *Flucoma* und einem einfachen hausgemachten k-means Clustering [18, 19] einer Klangdeskriptorextraktion unterziehen. Die mehrsprachige Funktion der *Bark*-Modelle ermöglicht eine noch größere klangliche Vielfalt. Insbesondere analysieren wir Tonhöhenfolgen und Mel-Frequency Cepstral Coefficients (MFCCs), die aus der Ausgabe abgeleitet werden, und können so ein tonhöhen- oder klangfarbenbasiertes Clustering von Token durchführen.[8] Dieser Ansatz erweist sich als sehr effektiv für die Segmentierung von Token-Sequenzen (fig. 6). Anschließend verwenden wir die zuvor beschriebene einfache, aber effektive gewichtete LZ-Methode, um Token abzufragen und dann aus dem Modell abzuleiten. Schließlich können wir algorithmisch mit Rekursionen eine Sequenz zusammenstellen, die eine Reihe von Stimmdeskriptoren wie Vokale, Frikative, Nasale und Transienten enthält (fig. 7).

Wir konzentrieren uns vorerst auf die ursprüngliche Idee, die Text-to-Speech-Synthese zu verwenden. Die Analyse von Klängen, die aus Token-Ketten abgeleitet werden, kann jedoch auch die Inferenz leiten und zu verschiedenen Klangzielen konvergieren, ähnlich wie bei der konkatenativen Synthese [20].

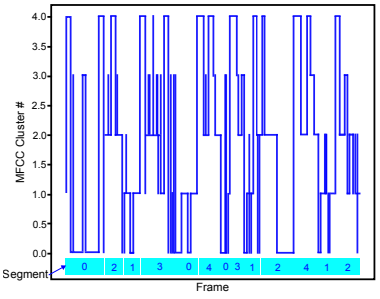

Figure 6. Einfaches k-mean Clustering von MFCC-Segmenten, bevor sie als Token mit unseren Markov-Methoden verwendet werden. Wir haben hier in einen größeren Datensatz gezoomt, um die Sichtbarkeit zu verbessern. Dennoch ist es offensichtlich, dass die Anzahl der Cluster einen erheblichen Einfluss auf die Verständlichkeit der synthetisierten Stimme hat.

[7] https://ollama.com - Wir fordern meist Mistral-Modelle an und verwenden häufig die Retrieval-Augmented Generation (RAG)-Technik.

[8] https://www.flucoma.org und Python Flucoma

5.3 *Hack 5*: Lange Generierung durch fine-tuning

Wir haben festgestellt, dass die maximale Länge des Audios in *Bark* 756 semantische Token beträgt, was etwa 15 Sekunden entspricht. Im Gegensatz dazu hat der grobe akustische Modelltransformator keine solche Grenze. Anschließend füttern wir ihn rekursiv mit Sequenzen von semantischen Token und stellen sicher, dass wir dieselben Klangeigenschaften wie bei den vorherigen Iterationen reproduzieren können. Wir haben zunächst eine Feinabstimmung des groben akustischen Modells vorgenommen. Bessere Ergebnisse erzielten wir jedoch bei der Feinabstimmung aller drei Modelle unter Verwendung eines Parameters, den wir *history-prompt* nennen. Die gleichzeitige Abstimmung aller drei Modelle wird auch verwendet, um die Persönlichkeit eines bestimmten Schauspielers oder Sängers zu ermitteln (deepfake).

Wir haben eine Methode entwickelt, die lange Sätze von Token dynamisch steuert, indem sie Schlussfolgerungen aus dem latenten Raum zu einem gewünschten Ziel führt. Der Weg dorthin wird durch die Kontrolle der Wahrscheinlichkeitsgewichte früherer Token, die Auswahl geeigneter historischer Prompts und die Manipulation der Temperatur zur Verbesserung der Neuheit und der Propositionen eingeschlagen. Wir können auch sowohl textuelle Aufforderungen als auch die oben beschriebene Sound-Deskriptor-Methode verwenden, um die Entwicklung langer, komponierter Sequenzen zu automatisieren:

1. Wir iterieren unser System bei einer relativ hohen Temperatur, um die Vorhersagen zu erweitern.
2. Wenn die Klangeigenschaften ein zufriedenstellendes Ziel erreichen, verwenden wir die Ergebnisse als History-Prompt und behalten dieselben Seed-Nummern, um sie in den nachfolgenden Schritten erneut zu injizieren.
3. Wir verwenden Seed und History-Prompt, um diesmal neue Versionen mit niedriger Temperatur zu erzeugen.
4. Wir erhöhen die Temperatur schrittweise, bis wir ein neues Ziel erreichen, das automatisch durch die Analyse der Geräuschdeskriptoren definiert wird.
5. Wir wiederholen ab Punkt 2.

Jede der oben erzeugten Versionen kann unabhängig oder zusammen verwendet werden, um Mehrstimmigkeit zu erzeugen. Wir haben dieses Zusammenspiel auch genutzt, um Hip-Hop-Stems zu erzeugen, indem wir versucht haben, die Konvergenzpunkte in bestimmten Momenten eines Liedes zu kontrollieren. Durch die Auswahl einer optimalen Größe für die Fensterung und die Gruppe von Token am Eingang des Systems erzeugt dieser Ansatz eine unbeschreibliche Assemblage, bei der sich die dramatische Intensität der Stimme und das Genre in rätselhafte Mäander auflösen und eine beunruhigende Desorientierung erzeugen. Diese Herangehensweise unterstreicht die Kraft der Stimme aus einem dramatischen Blickwinkel heraus.

6. ZUKUNFT UND MUSIKALISCHE ANWENDUNGEN

Wir haben diese Techniken bei der Produktion von Remixen für populäre Musiksänger mit der Genehmigung von Warner Music eingesetzt. Diese Methoden werden auch für eine englische und französische Theaterfassung von John Fosses Stück *And We'll Never Be Parted*, das im September 2025 im T2G National Theatre uraufgeführt wird, verwendet werden.

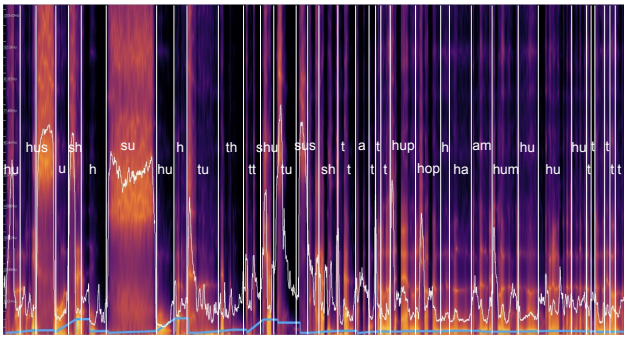

Figure 7. LPC-Spektrogramm mit einer Erzeugung von `hu`, `tu`, `shu`... Die in das grobe akustische Modell eingeführte Temperatur sorgt für mehr Vielfalt. Es vermeidet auch Wiederholungen in der Sequenz, indem es mehr Phoneme hinzufügt, die mit den von den Eingabe-Tokens gegebenen verwandt sind. Ursprünglich als Stimme artikuliert, klingen diese Segmente sehr ähnlich wie die artikulierten Übergänge der elektro-akustischen Musik. Eine Tonhöhenkurve in Blau und ein Schwerpunkt in Weiß wurden hinzugefügt, um die Intensität und die Beugungen in dieser Phrase besser zu visualisieren. Beachten Sie das Ende, das nur zu einer Tonhöhe konvergiert, in blau.

Der Teil, der GPT-2-ähnliche Transformers verwendet, ist für unsere Bedürfnisse zufriedenstellend. Allerdings hat sich das Training eines großen persönlichen Bark-Modells als schwierig, wenn nicht gar unmöglich erwiesen. In naher Zukunft werden wir den Arbeitsablauf vereinfachen, indem wir weniger externe Bibliotheken verwenden und das gesamte System einfach auf FairSeq 2 portieren. Wir wollen die Vielfalt und den Stil mit Hilfe der Low-Rank-Adaptation (LoRA) auf viel größere Modelle ausweiten [21]. Wir sollten auch in der Lage sein, diese Modelle so einfach zu fusionieren, wie wir es mit grafischen, stabilen Diffusionswerkzeugen tun.

7. CONCLUSION

Die Integration von transformatorbasierter TTS-Synthese und maschinellem Lernen in die Produktion ermöglicht einen kreativen Ausdruck zwischen Text, Klang, Wörtlichkeit und Abstraktion. Wir können lange, einzigartige Sprachsequenzen mit Hilfe von Token-Engineering, gesteuert durch Sound-Deskriptoren, und Feinabstimmungsmodellen erzeugen. Diese Forschung zeigt, wie nützlich die Verwendung einfacher Konzepte ist, um eine intuitive Steuerung zu erreichen. Die Verwendung von Transformers mag zunächst kontraproduktiv für Neuartigkeit und Kreativität erscheinen. Die Integration parametrischer Prozesse ermöglicht jedoch unerwartete textliche und klangliche Überraschungen, die sich von denen unterscheiden, die sich allein aus dem Schauspiel ergeben. Wir können Text und Musik nahtlos in einer personalisierten und einzigartig gestalteten Weise zusammenführen und so kreative Unabhängigkeit und Originalität gewährleisten, ohne uns vollständig auf extern kontrollierte Online-Plattformen oder Produkte zu verlassen.

Ein Jupyter-Notizbuch mit allen Sequenzen und Audiobeispielen ist hier auf GitHub verfügbar.

8. REFERENCES

[1] M. Woo, "*L'interprétation musicale des phonèmes, des gestes et des images dans Machinations de Georges Aperghis*," 2011.

[2] P. Shapiro and I. Lee, "*Modulations: a history of electronic music: throbbing words on sound*," 2000.

[3] A. Gayraud, R. Mackay, D. Miller, and N. Power, "*Dialectic of Pop*," 2019.

[4] *Rubin, Rick and Strauss, Neil*, "The Creative Act: A Way of Being," 2023.

[5] "*DeepRapper: Neural Rap Generation with Rhyme and Rhythm Modeling*."

[6] C. Durt and T. Fuchs, "*Large Language Models and the Patterns of Human Language Use*," 2024.

[7] F. Kreuk, A. Polyak, J. Copet, E. Kharitonov, T.-A. Nguyen, M. Rivière, W.-N. Hsu, A. Mohamed, E. Dupoux, and Y. Adi, "*Textless Speech Emotion Conversion using Discrete and Decomposed Representations*," 2022.

[8] L. Barrault, P.-A. Duquenne, M. Elbayad, and A. Kozhevnikov, "*Large Concept Models - Language Modeling in a Sentence Representation Space*," 2024.

[9] A. Harker, "*FrameLib: Audio DSP using Frames of Arbitrary Length and Timing*," 2016.

[10] N. Schnell and D. Schwarz, "*Gabor, Multi-representation Real-Time Analysis/Synarticle*," 2005.

[11] N. Obin, "*Cries and Whispers - Classification of Vocal Effort in Expressive Speech*," 2012.

[12] J. Kammerer, "*Unleashing Creativity with Ableton's Beat Repeat: A Comprehensive Guide*," 2014.

[13] G. Loizillon, "*Diphone Studio*," 1999.

[14] J. Bachan, "*Efficient Diphone Database Creation for MBROLA, a Multilingual Speech Synarticleer*," 2010.

[15] R. Shwartz-Ziv and Y. LeCun, "*To Compress or Not to Compress- Self-Supervised Learning and Information Theory: A Review*," 2023.

[16] J. Ziv and A. Lempel, "*Compression of Individual Sequences via Variable-Rate Coding*," 1978.

[17] O. Lartillot, "*OpenMusic LZ 2.2 Library*," 2001.

[18] P. A. Tremblay, O. Green, G. Roma, and A. Harker, "Fr*om Collections to Corpora: Exploring Sounds through Fluid Decomposition*," 2019.

[19] T. Moore, J. Bradbury, and P. A. Trembly, "*FluCoMa for Pedagogues*," 2022.

[20] B. Hackbarth, N. Schnell, P. Esling, and D. Schwarz, "*Composing Morphology: Concatenative Synarticle as an Intuitive Medium for Prescribing Sound in Time*," 2013.

[21] C. V. Nguyen, X. Shen, R. Aponte, Y. Xia, S. Basu, Z. Hu, J. Chen, M. Parmar, S. Kunapuli, J. Barrow, J. Wu, A. Singh, Y. Wang, J. Gu, F. Dernoncourt, N. K. Ahmed, N. Lipka, R. Zhang, X. Chen, T. Yu, S. Kim, H. Deilamsalehy, N. Park, M. Rimer, Z. Zhang, H. Yang, R. A. Rossi, and T. H. Nguyen, "*A Survey of Small Language Models*," 2024.

FR

SYNTHÈSE VOCALE MUSICALE AU POINT MÉDIAN : AU CROISEMENT DU TEXTE ET DU SON

Dr Olivier Pasquet
Goldsmiths, University of London

RÉSUMÉ

Cette recherche explore l'application de modèles de langage et de techniques d'apprentissage automatique pour générer des voix contrôlées musicalement et dotées de personnalité, d'expressivité. En utilisant des transformeurs autorégressifs, en particulier le modèle Bark, des tokens sont générés à partir d'un texte d'entrée pour produire un langage unique, inventé et indéfinissable.
La génération de voix musicale se réalise ici entre le texte et le son à l'aide de diverses techniques de contrôle, notamment la répétition et le fenêtrage des tokens, la compression Lempel-Ziv-Welch et le regroupement de tokens à partir de l'extraction de descripteurs sonores, afin de réguler la granularité, l'intelligibilité et la signification de la voix en sortie. Une méthode de génération récursive est également introduite, permettant la création et le control de séries de voix originales.
Cette recherche est utilisée dans divers projets artistiques, y compris des remix de musiques populaires et des productions théâtrales. Elle explore d'autres formes d'expression, de contrôle vocal et de narration, à mi-chemin entre le texte et le son.

1. INTRODUCTION

Nombreux sont ceux qui ont déjà joué avec des moteurs de synthèse vocale en tapant des textes absurdes en entrée. Nous avons même construit des boucles et des textes aléatoires, que nous avons ensuite utilisés pour créer des milliers de fichiers audio pour traitements dans des projets artistiques précédents utilisant aussi la voix synthétique. Les résultats étaient très satisfaisants, mais ils avaient tendance à devenir banals, à moins que nous n'introduisions des effets sonores supplémentaires en aval. Nous n'avons jamais réussi à trouver un équilibre parfait entre le texte et le son ; il s'agissait toujours d'un processus de travail en va-et-vient entre ces deux éléments.

2. GÉNÉRATION DE VOIX AVEC PERSONNALITÉ

2.1. Voix artificielles « musicales »

L'œuvre de Georges Aperghis a évidemment influencé notre recherche. Ses pièces de théâtre-musical échappent à toute catégorisation. Mais elle appartient à un genre de composition vocale qui s'appuie fortement sur une manipulation virtuose des « phonèmes », marquée par des tempos rapides, des motifs répétitifs et des techniques d'accumulation, le tout générant une énergie rythmique intense[12]. [1] La création d'un « langage imaginaire » donne lieu à un paysage sonore à la fois ambigu et humoristique, évoquant l'illusion de la communication tout en restant musical. La frontière entre l'expression linguistique et la composition musicale est ainsi brouillée.

Cette ligne floue peut généralement être explorée et utilisée séparément du texte symbolique et du traitement du signal. Toutefois, il existe des exceptions qui fusionnent ces étapes, comme les techniques vocales musicales expressionnistes de Sprechgesang ou Sprechstimme. Elles se situent au milieu, mais suivent également une chaîne de production dirigée qui commence avec une partition symbolique, puis continue par l'interprétation des chanteurs.

Le hip-hop et le R&B ont continuellement repoussé les limites de l'expression vocale grâce à diverses techniques brouillant la frontière entre le littéralisme et l'abstraction [3]. Les techniques vocales notables s'ajoutent aux techniques audio telles que l'Auto-tune [20, 10]. Depuis la fin des années 1990, l'Auto-tune est passé du statut de simple outil de correction vocale à celui de phénomène culturel. Cet effet basé sur de la re-synthèse peut être étendu à une voix artificielle contrôlée par de la voix. Cet instrument est capable de transformer à la fois la voix elle-même et le sens qu'elle véhicule.

2.2. Recherche d'une synthèse avec de la personnalité

La synthèse vocale contrôlée par un Large Language Model (LLM) permet de générer un large éventail de textes et de techniques vocales qui sont différents lorsqu'ils sont demandés à un interprète. En fonction des modèles et des techniques utilisés, la synthèse peut apporter un large éventail de variabilité, de subversion et d'inspirations. Cette synthèse permet aussi le détachement émotionnel, le genre et la neutralité que nous pouvons hybrider à volonté.

En outre, il permet de composer à l'endroit exact de la ligne floue entre le littéralisme et l'abstraction, et entre le texte symbolique et le signal.

Cependant, la qualité de la plupart des moteurs de synthèse est devenue trop bonne pour être de simples instruments. L'absence de problèmes et d'incohérences n'amé-

1 . Les « phonèmes » ne font pas référence à la linguistique proprement dite, mais plutôt à des portions de mots ou à des techniques vocales.

liore pas la créativité de l'outil. De plus, ils perdent considérablement en caractère, et la voix est bien moins créative que celle d'un véritable acteur, par exemple. Enfin, il est souvent difficile de maitriser un style personnel sans une architecture que nous pouvons contrôler, et en évitant les effets classiques de balbutiement vocal liés une l'esthétique du début des années 2020.

3. WORKFLOW

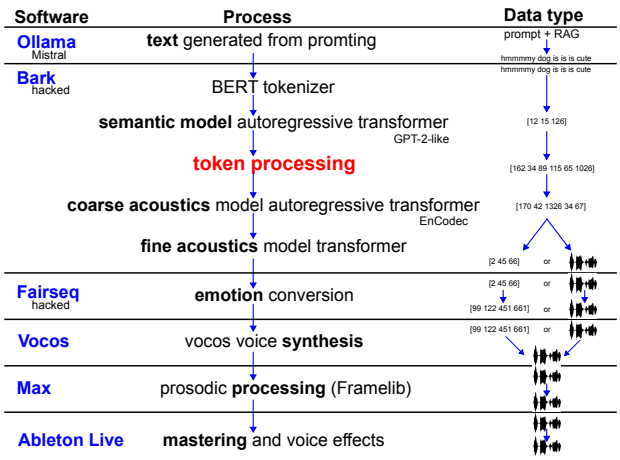

Figure 1. Workflow proposé avec des variables qui peuvent être utilisées à des fins musicales. Ce document se concentrera spécifiquement sur la partie *Bark*, en particulier la section rouge, qui s'est avérée être la plus efficace et la plus expressive.

Nous proposons dans ce papier un workflow offrant un contrôle à chaque étape, de la génération initiale du texte à la voix finale. (Fig. 1) :

- Nous commençons par générer du texte à l'aide de LLM, en procédant à divers raffinements, en segmentant en unités et créant des boucles [21, 4].
- Nous générons des séries de token à l'aide d'un moteur de synthèse vocale appelé *Bark*. [2]
- Nous les convertissons afin d'utiliser directement la bibliothèque FairSeq pour pouvoir contrôler l'expressivité. [3] [8, 9]
- La synthèse finale est réalisée à l'aide du vocodeur neuronal Vocos et « upscalée » à la bonne fréquence d'échantillonnage. [4]
- Le résultat est envoyé à Max pour le traitement prosodique et à Ableton Live pour le mastering et le doublage des voix.

Bien que tous les composants soient interdépendants et parfois nécessairement influencés les uns par les autres. Nous nous concentrerons ici que sur une seule partie du workflow : la section rouge appelée token processing, illustrée par Fig. 1, qui traite les *token sémantiques*. C'est cette partie rouge qui travaille précisément dans le domaine qui nous interesse ici : à mi-chemin entre le texte et la génération audio.

2 . Bark lib initiale de Suno : https://github.com/suno-ai/bark
3 . Fairseq lib : https://github.com/facebookresearch/fairseq
4 . Vocos lib : https://github.com/gemelo-ai/vocos

4. BARK HACKÉ

4.1. Adapter Bark à la composition

Nous avons décidé de transformer une version de *Bark*, un modèle de type GPT (Generative Pre-trained Transformer) qui prend des libertés créatives dans sa génération. Ce programme de Suno, datant de 2023, n'offre pas la meilleure qualité sonore, mais cela n'a pas d'impact sur la qualité globale de notre système, car nous utilisons de nombreux autres processus et synthèses par la suite. De plus, l'étape utilisant Max à la fin du workflow est basée sur de la re-synthèse utilisant *FrameLib* et change la qualité sonore de toute manière. [5] [7, 2] Cela modifie considérablement les qualités de la voix à des fins esthétiques.

Bark est constitué d'une série de transformeurs autorégressifs utilisant un modèle sémantique (semantic model), un modèle acoustique grossier (coarse acoustics model) et un modèle acoustique fin (fine acoustics model) :

- Le *fine acoustics model* prend en entrée les tokens prédits par le coarse acoustics model et prédit itérativement les tokens prêts pour la synthèse audio. L'utilisation du codec neuronal EnCodec permet de coder puis de connecter *Bark* à d'autres bibliothèques. [6] [19]
- Le *coarse acoustic model* est un transformeur causal de type GPT-2 qui convertit les tokens sémantiques en token coarse acoustic.
- Le *semantic model* est également un modèle de transformeur auto-régressif causal de type GPT-2, agrémenté d'une modélisation linguistique. Il prend en entrée un texte tokenisé (à partir d'un tokenizer BERT) et prédit ensuite les tokens sémantiques qui codent l'audio à générer [22]. Cette partie est la plus importante pour l'identité du locuteur. Nous pouvons ici ajouter des prompts sonores qui définiront le mieux les traits de personnalité grâce à leur intonation et à leur prosodie.

Travailler principalement avec des tokens du semantic model nous permet de conserver l'intention initiale d'équilibrer la composition entre le texte et l'audio.

4.2. *Hack 1* : Fenêtrage variable des token

L'architecture de *Bark* est pratique pour la créativité grâce à son architecture GPT qui s'étend au-delà de la voix seule. Ses modèles englobent une grande variété de communications non verbales telles que les rires, les soupirs, les pleurs et d'autres surprises qui peuvent être appelées par un message-guide en fonction du modèle utilisé. Toutefois, cette force s'accompagne également d'une faiblesse : les résultats deviennent rapidement imprévisibles s'ils ne

5 . FrameLib : https://github.com/AlexHarker/FrameLib
6 . EnCodec codec : https://github.com/facebookresearch/encodec

sont pas correctement déduits. La longueur maximale de l'audio dans Bark est de 756 tokens sémantiques, ce qui équivaut à environ 15 secondes. Cela est dû au fait que la taille de la fenêtre contextuelle du modèle est plafonnée à 1024, de la même manière que les modèles de langage textuel ont des tailles de contexte limitées aujourd'hui (par exemple 4096). Il convient de noter que si *Bark* utilisait un positionnement relatif au lieu d'un positionnement absolu, il aurait été possible d'obtenir des tailles de contexte plus importantes, telles que 2048 ou 4096 tokens. Toutefois, nous n'avons pas encore trouvé de techniques permettant d'atteindre cet objectif avec un positionnement absolu.

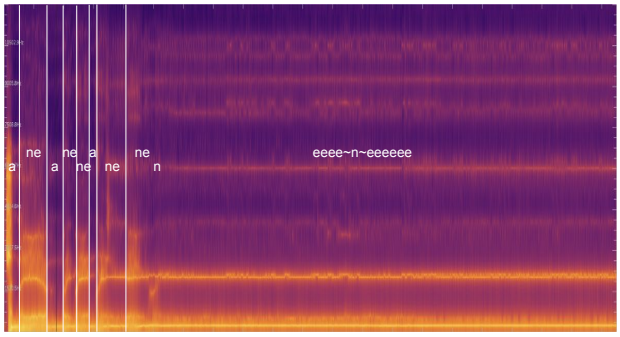

Figure 2. Spectrogramme LPC montrant les limites prédictives du modèle. L'utilisation d'une température basse fait diverger le modèle lorsqu'il n'utilise pas de séquences de tokens non conventionnelles ou lorsqu'il demande une durée plus longue que celle pour laquelle le système a été conçu. Nous voyons ici qu'il commence par le texte prévu, puis entre dans une boucle. Cette caractéristique inhérente peut cependant être utilisée artistiquement.

Au lieu de cela, nous avons créé un système permettant de jouer avec des fenêtres variables de tokens, ce qui permet de contrôler la granularité du son et donc l'intelligibilité de la voix. Cela correspond à l'esthétique rythmique infinitésimale que nous envisageons dans notre musique de manière réccurente. Cependant, l'utilisation d'un ensemble de tokens aléatoires à ce stade perturbe rapidement les capacités prédictives du modèle, ce qui se traduit généralement par un résultat dégénéré : une hauteur convergente et monotone avec du filtrage et du bruit (Fig. 2). Le choix de la taille de la fenêtre variable et le caractère aléatoire des séquences de tokens non entrainées définissent le degré de prédiction.

4.3. *Hack 2* : Contrôle des variables de bruit

Chacune de ces trois couches est dotée des commandes standard suivantes, que l'on retrouve dans ce type de modèles. Il s'agit des commandes de température, Top-p et Top-k :

— Le réglage de la *température* régit le degré d'aléatoire dans la sélection des mots lors de la génération de textes. Des températures plus basses produisent des résultats plus prévisibles et cohérents, tandis que des températures plus élevées introduisent une plus grande liberté et une plus grande créativité, au détriment toutefois de la cohérence.
— Le paramètre *Top p* détermine le nombre de mots probables pris en compte par le modèle. Des valeurs élevées permettent au modèle d'examiner un plus large éventail de possibilités, y compris des mots moins probables, ce qui se traduit par une plus grande diversité du texte généré.
— Le réglage du paramètre *Top k* influence la répétitivité et la complexité des réponses, notamment au niveau du vocabulaire et du phrasé.

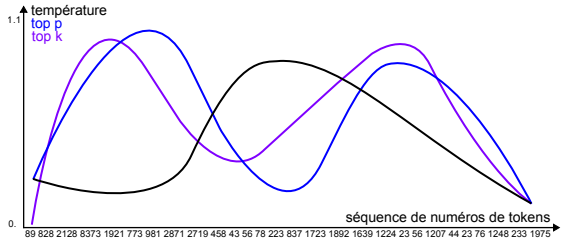

Figure 3. Une « break-point function » interpolée a démontré son efficacité dans le contrôle de la température, du top p et du top k, à travers les séquences de tokens, permettant articulation et variations temporelles ; elle est simple et efficace sur le plan perceptif. L'utilisation d'un bruit simplex 3D pour corréler ces paramètres permet d'obtenir une plus grande cohérence.

Nous avons mis en œuvre une méthode de contrôle de ces valeurs à l'aide de « break-point functions » (BPF), qui nous permettent de réguler les quantités d'aléatoire dans chaque séquence de tokens donnés (Fig. 3). Cela permet de jouer avec l'intensité variable de l'expression tout au long d'une phrase ou d'une série de tokens.

Nous avons ajouté plusieurs moteurs aléatoires et de quantification semblables à ceux que l'on trouve dans la fonction *Beat Repeat* d'Ableton Live [15]. La répétition aléatoire de tokens de cette manière ressemble beaucoup à de la synthèse granulaire. Mais utiliser des transformeurs avec température offre un résultat plus dynamique et plus humain. Elle offre un plus grand contrôle et une plus grande expressivité que la simple concaténation directe de grains [23]. L'articulation entre les « phonèmes » vise le naturel et peut parfois évoquer le goût de l'interpolation additive que l'on trouve dans *Diphone* [13, 14].

Ici aussi, le hasard joue un rôle précieux car il facilite les trouvailles fortuites. On peut donc contrôler la génération de séquences de tokens discrets en utilisant simplement des techniques markoviennes et ainsi jouer avec la chance.

5. INGÉNIERIE DES TOKENS

Afin d'obtenir des résultats intéressants, nous devons générer plus de phrases que nécessaire, produisant ainsi un grand ensemble de tokens qui peuvent ensuite être organisés à l'aide de probabilités. Nous utilisons donc les modèles d'Ollama pour produire un ensemble de phrases pa-

raphrasées qui partagent un nombre suffisant de mots et de sens communs. [7] Cette cohérence nous permet d'obtenir des résultats textuels plus significatifs et nous rapproche de notre objectif initial de création d'une langue inventée. Le fait que notre système actuel de synthèse « linguistique » et vocale utilise des probabilités et des réseaux neuronaux est intéressant d'un point de vue historique [1].

5.1. *Hack 3* : Décodage LZ de tokens

Nous utilisons ensuite une méthode de compression multiscalaire basée sur l'algorithme de Lempel-Ziv-Welch (LZ), qui fonctionne efficacement avec des séquences, en sachant que, comme nous l'avons vu, le fenêtrage joue un rôle essentiel. L'algorithme d'encodage LZ compresse des séquences de tokens de taille variable en les mettant en correspondance avec un dictionnaire et en émettant des références à ce dictionnaire sous la forme d'une chaîne de caractères [18].

Nous pouvons ensuite décoder des séquences de longueur arbitraire à partir de la chaîne, en générant des séquences concaténées à l'infini [11]. L'utilisation de LZ avec une probabilité pondérée sur la chaîne s'est avérée être la méthode la plus efficace pour réguler le niveau de signification et d'abstraction de nos voix.

LZ fonctionne de manière optimale sur des données contenant des motifs répétitifs, ce qui le rend bien adapté aux paraphrases décrites plus haut. Sa nature multiscalaire nous permet de choisir des longueurs de séquence spécifiques dans le dictionnaire LZ et de jouer avec la fenêtre contextuelle de notre modèle, comme décrit précédemment (Fig. 4).

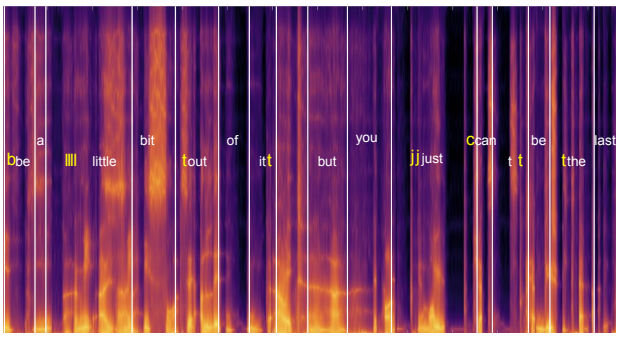

Figure 4. Exemple d'utilisation : génération de semi-mots et de semi-phrases en anglais à l'aide d'une séquence LZ (en blanc) superposée à des répétitions accidentelles d'ensembles particuliers de tokens sémantiques (en jaune).

5.2. *Hack 4* : Séquencement de propriétés acoustiques

Nous visons maintenant à réaliser une segmentation des tokens qui nous permette d'extraire des sons ou des motifs spécifiques de la sortie du modèle. L'imprévisibilité inhérente aux transformeurs autorégressifs, en particulier lors de l'utilisation de modèles non entrainés par nos soins, nécessite une analyse approfondie du comportement du modèle : *Nous interrogeons le modèle avant de l'utiliser*. Pour minimiser l'incertitude, nous employons une stratégie déterministe en utilisant des réglages avec basse température et un numéro de « seed » unique (Fig. 5).

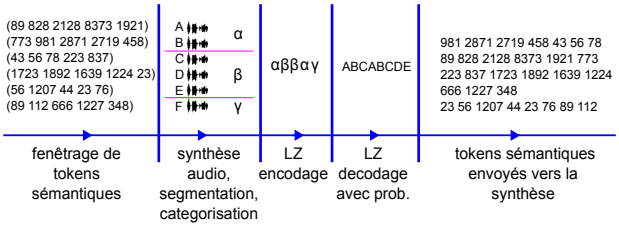

Figure 5. Analyse audio du comportement du modèle Bark à l'aide des descripteurs de Flucoma : Nous interrogeons le modèle avant de l'utiliser. Pour minimiser l'incertitude, nous employons une stratégie déterministe en utilisant des réglages à basse température et un numéro de « seed » unique.

Nous commençons notre analyse par un ensemble complet de phrases paraphrasées, que nous soumettons ensuite à l'extraction de descripteurs sonores à l'aide de la bibliothèque Python *Flucoma* et d'un simple regroupement (clustering) k-means fait maison [8] [17, 6]. La caractéristique multilingue des modèles *Bark* permet une variété timbrale encore plus grande. Plus précisément, nous analysons les séquences de hauteur et les coefficients cepstraux de fréquence Mel (MFCC) dérivés de la sortie, ce qui nous permet d'effectuer un regroupement des tokens basé sur la hauteur ou sur le timbre. [9] Cette approche s'avère très efficace pour la segmentation des séquences de mots (Fig. 6). Nous utilisons ensuite la méthode simple mais efficace de LZ pondérée décrite plus haut pour placer chaque segment de token en fonction de propriétés analysées. Nous pouvons finalement composer algorithmiquement et récursivement une séquence contenant une succession de descripteurs vocaux tels que les voyelles, les fricatives, les nasales, les transitoires, etc.(Fig. 7).

Pour l'instant, nous restons concentrés sur l'idée initiale d'utiliser la synthèse de la parole à partir du texte. Cependant, l'analyse des sons dérivés des chaînes de tokens peut également guider l'inférence et converger vers diverses cibles sonores, à l'instar de la synthèse concaténative [16].

5.3. *Hack 5* : Génération longue par « fine-tuning »

Nous avons observé que la longueur maximale de l'audio dans *Bark* était de 756 tokens sémantiques, ce qui équivaut à environ 15 secondes. En revanche, le transformeur du *coarse acoustic model* n'a pas de limite de ce type. Nous alimentons donc ce dernier avec un large nombre de séquences de tokens sémantiques et nous nous assurons que nous pouvons reproduire les mêmes caractéristiques

7 . https://ollama.com - Nous promtons principalement les modèlesMistral et utilisons allègrement la technique du Retrieval-Augmented Generation (RAG) technique.

8 . https://github.com/jamesb93/python-flucoma

9 . https://www.flucoma.org

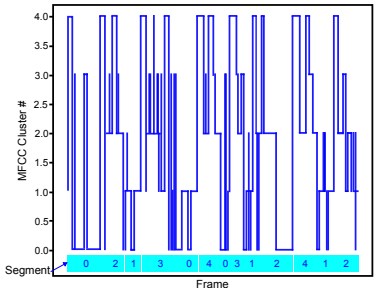

Figure 6. Clustering simple par k-mean des segments MFCC avant de les utiliser comme tokens avec nos méthodes markoviennes. Nous avons ici zoomé sur un ensemble de données plus important pour améliorer la visibilité. Néanmoins, il est évident que la quantité de clusters a un impact significatif sur l'intelligibilité de la voix synthétisée..

pour chacunes des itérations du processus. Nous affinions (fine-tune) uniquement le *coarse acoustic model* jusqu'à ce qu'on remarque qu'on obtient de meilleurs résultats en affinant les trois modèles à la fois, à l'aide d'un paramètre que nous appelons « history-prompt ». L'ajustement des trois modèles est également utilisé pour cibler la personnalité d'un acteur ou d'un chanteur spécifique (deepfake).

Nous avons conçu une méthode qui contrôle dynamiquement des longues séries de tokens en guidant les inférences de l'espace latent vers une cible souhaitée. Le chemin se fait en contrôlant des poids de probabilité des tokens précédents, en sélectionnant des history-prompt appropriés et en manipulant la température. Nous pouvons aussi utiliser à la fois des prompts textuels et la méthode des descripteurs sonores décrite précédemment pour automatiser l'évolution de séquences longues et composées :

1. Nous itérons notre système avec une température relativement élevée afin d'élargir les prédictions.
2. Lorsque les caractèristiques sonores atteignent un objectif satisfaisant, nous utilisons les resultats comme history-prompt et gardons les mêmes numéros de seed pour les réinjecter dans les étapes suivantes.
3. Nous utilisons ces seeds et history-prompts pour générer de nouvelles versions à basse température cette fois-ci.
4. Nous augmentons progressivement la température, étape après étape, jusqu'à ce que nous atteignions un nouvel objectif défini automatiquement par les descripteurs sonores.
5. Nous reprenons à partir du point 2.

Chacune des versions générées ci-dessus peut être utilisée indépendamment ou ensemble pour créer de la polyphonie. Nous avons également utilisé cette interaction pour générer des stems hip-hop en essayant de contrôler des points de convergences à des moments spécifiques d'une chanson. En sélectionnant une taille optimale pour le fenêtrage et le groupe de tokens en entrée du système, cette approche produit un assemblage indescriptible où l'intensité dramatique de la voix, et le genre, se dissolvent dans des méandres énigmatiques, suscitant une déstabilisation troublante. Cette approche met en lumière toute la puissance de l'utilisation de la voix d'un point de vue dramatique.

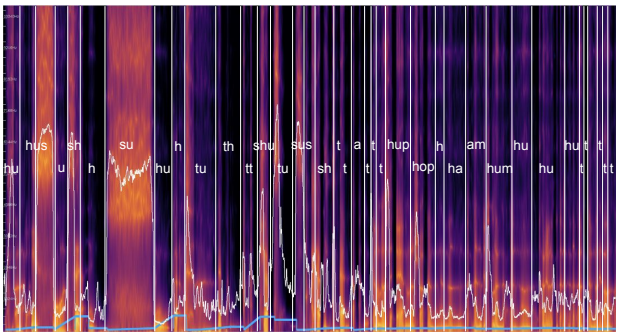

Figure 7. Spectrogramme LPC montrant une génération de `hu`, `tu`, `shu`... La température introduite dans le modèle acoustique grossier ajoute une plus grande variété. Elle permet également d'éviter les répétitions dans la séquence en ajoutant davantage de phonèmes liés à ceux donnés par les tokens d'entrée. Initialement articulés comme de la voix, ces segments ressemblent beaucoup à des transitions articulées de type « musique électroacoustique ». Une courbe de hauteur en bleu et un centroïde en blanc ont été ajoutés pour mieux visualiser l'intensité et les inflexions de cette phrase. Remarquez que la fin converge vers une seule hauteur, en bleu.

6. FUTURE ET APPLICATIONS MUSICALES

Nous avons utilisé ces techniques dans la production de plusieurs remixes pour des chanteurs de musique populaire avec l'autorisation de Warner Music. Ces méthodes seront également largement utilisées pour une version musicale anglaise et théâtrale française de la pièce *Et nous ne serons jamais séparés* de John Fosse, dont la première aura lieu au T2G National Theatre en septembre 2025.

La partie utilisant des transformeurs de type GPT-2 est satisfaisante pour nos besoins. Cependant, l'entraînement d'un grand modèle *Bark* personnel s'est avéré difficile, voire impossible. Nous simplifierons, dans un future proche, le flux de travail en utilisant moins de bibliothèques externes et en portant l'ensemble du système sur FairSeq 2. Nous voulons augmenter la variété et le style en affinant des modèles beaucoup plus grands que ceux utilisés actuellement avec des Low-Rank Adaptation (LoRA) [5]. Nous devrions également être en mesure de fusionner ces modèles aussi facilement que nous le faisons dans les outils visuels de stable diffusion.

7. CONCLUSION

L'intégration de la synthèse TTS basée sur des transformeurs et de l'apprentissage automatique permet une expression créative entre le texte, le son, le littéralisme et l'abstraction.

Nous pouvons générer de longues séquences vocales uniques en utilisant l'ingénierie des tokens contrôlée par des descripteurs sonores et en faisant du fine-tuning sur divers modèles . Cette recherche montre l'utilité d'utiliser des concepts simples pour obtenir des contrôles intuitifs.

L'utilisation de transformeurs peut initialement sembler contre-productive pour la nouveauté et la créativité. Cependant, l'intégration de processus paramétriques permet d'obtenir des surprises textuelles et sonores inattendues, distinctes de celles dérivées uniquement du jeu. Nous pouvons ainsi intégrer le texte et la musique de manière personnalisée.

Un notebook Jupyter avec toutes les séquences et exemples sonores est disponible ici sur GitHub.

8. REFERENCES

[1] Shwartz-Ziv, Ravid and LeCun, Yann, *To Compress or Not to Compress- Self-Supervised Learning and Information Theory : A Review,* 2023.

[2] Schnell, Norbert and Schwarz, Diemo, *Gabor, Multi-representation Real-Time Analysis/Synthesis,* 2005.

[3] Shapiro, Peter and Lee, Iara, *Modulations : a history of electronic music : throbbing words on sound,* 2000.

[4] Durt, Christoph and Fuchs, Thomas, *Large Language Models and the Patterns of Human Language Use,* 2024.

[5] Nguyen, Chien Van et al., *A Survey of Small Language Models,* 2024.

[6] Moore, Ted et al., *FluCoMa for Pedagogues,* 2022.

[7] Harker, Alex, *FrameLib : Audio DSP using Frames of Arbitrary Length and Timing,* 2016.

[8] Kreuk, Felix et al., *Textless Speech Emotion Conversion using Discrete and Decomposed Representations,* 2022.

[9] Barrault, Loic et al., *Large Concept Models - Language Modeling in a Sentence Representation Space,* 2024.

[10] Rubin, Rick and Strauss, Neil, *The Creative Act : A Way of Being,* 2023.

[11] Lartillot, Olivier, *OpenMusic LZ 2.2 Library,* 2001.

[12] Woo, Minjung, *L'interprétation musicale des phonèmes, des gestes et des images dans Machinations de Georges Aperghis,* 2011.

[13] Loizillon, Guillaume, *Diphone Studio,* Ircam, 1999.

[14] Bachan, Jolanta, *Efficient Diphone Database Creation for MBROLA, a Multilingual Speech Synthesiser,* Institute of Linguistics, Adam Mickiewicz University, 2010.

[15] Kammerer, John, *Unleashing Creativity with Ableton's Beat Repeat : A Comprehensive Guide,* 2014.

[16] Hackbarth, Benjamin et al., *Composing Morphology : Concatenative Synthesis as an Intuitive Medium for Prescribing Sound in Time,* Contemporary Music Review, Vol. 32, No. 1, 49-59. 2013.

[17] Tremblay, Pierre Alexandre et al., *From Collections to Corpora : Exploring Sounds through Fluid Decomposition,* 2019.

[18] Ziv, J. and Lempel, A., *Compression of Individual Sequences via Variable-Rate Coding,* 1978.

[19] Obin, Nicolas, *Cries and Whispers - Classification of Vocal Effort in Expressive Speech,* 2012.

[20] Gayraud, Agnès et al., *Dialectic of Pop,* 2019.

[21] XLanqing, *DeepRapper : Neural Rap Generation with Rhyme and Rhythm Modeling,* 2021.

[22] Devlin, Jacob and Chang, Ming-Wei and Lee, Kenton and Toutanova, Kristina, *BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding,* 2018.

[23] Schwarz D, Cahen R, and Britton S. *Principles and Applications of Interactive Corpus-Based Concatenative Synthesis,* 2008.

JP

中間点での音楽音声合成:
テキストとサウンドの出会い

Dr. Olivier Pasquet

Goldsmiths, University of London
o.pasquet@gold.ac.uk

ABSTRACT

この研究では、言語モデルと機械学習技術を応用して、個性と制御性を備えた音楽的な声を生成することを探求している。自己回帰変換器、特に*Bark*モデルを利用することで、入力テキストからトークンを生成し、独自の発明された定義不可能な言語を生成する。\\

トークンの反復とウィンドウ化、*Lempel-Ziv-Welch*圧縮、音響特徴抽出からのトークンのクラスタリングなど、様々な制御技術を用いてテキストと音声の合成を行い、出力音声の粒度、明瞭度、意味を調整する。また、トークン分析を用いた再帰的生成システムも紹介され、相互に関連する大規模な一連の音声を作成することができる。\\

この研究は、音楽のリミックスや演劇作品など、さまざまな芸術的応用に活用されている。テキストとサウンドの中間にシームレスに横たわる、他の形の表現音声やストーリーテリングを探求している。

1. はじめに

以前、多くの人が無意味なテキストを入力することで、音声合成（TTS）エンジンで遊んだことがある。私たちは、ループやランダムなテキストを作成し、それを使って合成音声を使った過去の作品の後処理用に何千もの音声ファイルを作成したこともある。結果はとても満足のいくものでしたが、後から音楽的な効果を加えなければ、平凡に感じてしまう傾向がありました。私たちは、テキストとサウンドをシームレスにバランスさせることはなく、本質的にこの2つの要素の間を行ったり来たりする作業だったのです。

この論文では、テキストと純粋な音楽の境界線を曖昧にする、本物の人工的な歌唱技術、声、抽象化を生み出すための様々なコンセプトとテクニックを探求することで、重要な課題に取り組む。TTS音声合成エンジンに深く潜り込むことで、テキストとサウンドの交差点で操作することができ、そこでこのユニークな融合が形作られる。このアプローチが、より広範な作曲ワークフローにどのようにシームレスに統合されるかを紹介し、最終的にSunoのシステム *bark* のために特別に設計された5つのクリエイティブ・ハックを紹介します。

Copyright: © 2025 Olivier Pasquet. This is an open-access article distributed under the terms of the Creative Commons Attribution License 3.0 Unported, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

2. 個性的な音声生成

1. 人工楽音

ジョルジュ・アペルギスの作品は、明らかに私たちの研究に影響を与えた。彼の音楽劇の作品は分類が難しい。しかしこの作品は、急速なテンポ、反復パターン、積み重ねの技法を特徴とし、強烈なリズム・エネルギーを生み出す「音素」の名人芸的操作に大きく依存した、一種の声楽作曲に属するものである [1]。[1]想像上の言語」を創り出すことで、曖昧で遊び心のあるユーモラスなサウンドスケープが生まれ、音楽でありながらコミュニケーションの錯覚を呼び起こす。これは言語表現と作曲の境界線を曖昧にする。

この曖昧な境界線は、通常、記号的なテキストと信号処理を別々に探求し、使用することができる。しかし、SprechgesangやSprechstimmeの表現主義的な音楽発声法のように、これらのステップを融合させる例外もある。それらは中間に位置するが、最初に記号的に定義され、次に声楽家によって解釈されるという連鎖をたどる。

ヒップホップとR&Bは、文字通りの表現と抽象的な表現の境界線を曖昧にする様々なテクニックによって、ヴォーカル表現の限界を押し広げ続けてきた [2]。オートチューンなどのオーディオ技術に、注目すべきボーカル技術が加わる [1, 4] 1990年代後半から、オートチューンは単なるボーカル修正ツールから文化的現象へと進化した。このエフェクトは再合成に基づいており、声によってコントロールされる人工音声として拡張することができる。この楽器は、声そのものとそれが伝える意味の両方を変えることができる。

2. 個性との融合を模索

ラージ・ランゲージ・モデル（LLM）によって制御された音声合成は、演奏者に依頼する際に異なる幅広いテキストや発声テクニックを生成することができる。モデルや使用するテクニックによって、合成は幅広い可変性、転覆性、インスピレーションをもたらすことができる。このような合成は、感情の剥

[1] 音素とは言語学的な意味ではなく、単語の一部分や音声テクニックのことである。

離、ジェンダー、中立性を可能にし、私たちが自由にハイブリッドすることができる。

さらに、文字主義と抽象主義、象徴的なテキストと信号の間の曖昧な境界線の正確な位置で作曲することができる。

しかし、ほとんどの合成エンジンの品質は、単なる楽器というにはあまりにも良くなりすぎている。不具合や一貫性のなさは、ツールの創造性を高めるものではない。さらに、キャラクターが著しく失われ、例えば本物の俳優の声よりもはるかに創造性に欠ける。最後に、私たちが完全にコントロールできるアーキテクチャがなければ、音声合成を使って作曲することは困難である。

3. ワークフロー

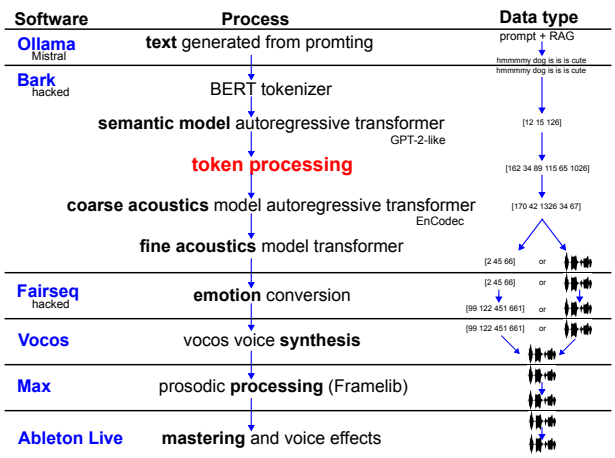

Figure 1. 音楽目的に活用できる変数を使ったワークフローを提案。本稿では特に、最も効果的で表現力豊かであることが実証されている、*Bark*の部分、特に赤い部分に焦点を当てる。

私たちは、最初のテキスト生成から最終的な音声まで、各段階での制御を提供するワークフローを提案する。(fig. 1):

- OllamaのLLMを使ってテキスト生成を開始し、様々な改良を加え、ユニットやループに分割する [5, 6]。
- という音声合成エンジンを使って一連の音声トークンを生成する*Bark*。[2]
- FairSeqライブラリを直接利用できるように変換し、発現を変換する [7, 8]。[3]
- 最終的な合成は、Vocosニューラル・ボコーダーを使って行われ、適切なサンプリング・レートにアップスケールされる。[4]
- 結果はマックスに送られ、韻律的処理が施され、マスタリングとヴォイス・ダブリングのためにエイブルン・ライブに送られる。

ここでは、音声合成の一部分に焦点を当てるが、すべての構成要素は相互に依存し、時には必然的に影響し合う。と呼ばれる赤い部分だけに集中する。token processing 図 fig. 1. このプロセス *semantic token* それは、まさに私たちが望むところ、つまりテキストと音声生成の中間に位置するものだ。

4. HACKED BARK

1. 適応 Bark 構成

私たちは、テキストを促音するGenerative Pre-trained Transformer-style（GPT-style）モデルであり、その生成において創造的な自由裁量を持つ、適応バージョンの *Bark* から始めることにした。Sunoのプログラムは2023年のもので、最高の音質を提供するものではありませんが、その後に多数の他の処理と合成を採用しているため、このことは私たちのシステムの全体的な品質には影響しません。さらに、ワークフローの最後にMaxを使用するステップは、以下のものを使用した再合成に基づいています *FrameLib* [9, 10]。[5] 美的な目的で声質を大きく変える。

Bark は、意味モデル、粗い音響モデル、細かい音響モデルを使った一連の自己回帰変換器からできている：

- 細かい音響モデルは、粗いモデルから生成された予測トークンを入力として受け取り、音声合成の準備ができたトークンを繰り返し予測します。EnCodecニューラル・コーデックを使用することで、ë(•̀ᴗ•́)ë を他のライブラリにフックすることができます [11]。[6]
- 粗音響モデルは、意味トークンを粗音響トークンに変換するGPT-2スタイルの因果変換器である。

[2] Suno's 初回 Bark lib: https://github.com/suno-ai/bark
[3] Fairseq lib: https://github.com/facebookresearch/fairseq
[4] Vocos lib: https://github.com/gemelo-ai/vocos
[5] FrameLib: https://github.com/AlexHarker/FrameLib
[6] EnCodec codec: https://github.com/facebookresearch/encodec

- 意味モデルもまた、GPT-2 のような因果的自己回帰変換モデルであり、その上に言語モデリングヘッドが搭載されている。トークン化されたテキスト（BERTトークナイザから）を入力として取り込み、生成される音声をエンコードする意味トークンを予測する。この部分は、話者のアイデンティティにとって最も重要です。ここでは、イントネーションと韻律パターンで話者の性格的特徴を最も明確にするプロンプトを追加できます。

主に意味モデルのトークンを使うことで、テキストと音声の合成のバランスをとるという当初の意図を維持することができた。

5. *Hack 1*: 可変 token ウィンドウウィング

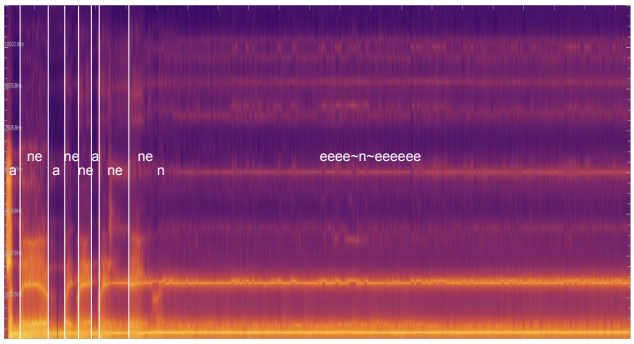

Figure 2. モデルの予測限界を示すLPCスペクトログラム。低温を使用することで、型にはまったトークンの連続を使用しない場合や、システムが設計したものよりも長い時間を要求する場合に発散する。ここでは、意図されたテキストから始まり、ピッチド・ループに入ることがわかる。この固有の特性は、芸術的なコントロールに変えることができる。

Barkのアーキテクチャは、音声だけにとどまらないGPTアーキテクチャのおかげで、創造性のために強力です。そのモデルは、笑ったり、ため息をついたり、泣いたり、その他の驚きのような多種多様な非言語的コミュニケーションを包含しており、使用するモデルによってはプロンプトによって呼び出すことができる。しかし、この強みは、適切に推論されないと出力がすぐに予測不能になるという弱点ももたらす。Barkの音声の最大長は、約15秒に相当する756セマンティックトークンである。これは、モデルのコンテキストウィンドウサイズが1024に制限されているためで、今日のテキスト言語モデルのコンテキストサイズが制限されているのと同様である（例えば4096）。Barkが絶対位置決めではなく相対位置決めを利用すれば、2048や4096といったより大きなコンテキストサイズを実現できたかもしれないことは注目に値する。しかし、現在のところ、絶対位置決めでこれを実現する技術は見つかっていない。

その代わりに、トークンのウィンドウを可変にして、音の粒度、ひいては声の明瞭度をコントロールできるシステムを作った。これは、私たちが音楽で思い描く無限のリズムの美学に沿ったものです。しかし、この段階でランダムなトークンのセットを使用すると、モデルの予測能力が急速に損なわれ、典型的には、フィルタリングとノイズを含む収束した単調なピッチという、退化した出力が得られます。(fig. 2). 可変窓のサイズとトークン列のランダム性の選択によって、どの程度の予測ができるかが決まる。

2. *Hack 2*: ノイズ変数の制御

これら3つの層はそれぞれ、このようなモデルで見られる以下のような標準的なコントロールを備えている。これらには、温度、Top-p、Top-kコントロールが含まれる：

- 温度設定は、テキスト生成時の単語選択におけるランダム性の度合いを支配する。温度が低いほど、予測可能で一貫性のある出力が得られるが、温度が高いほど、一貫性は犠牲になるものの、より自由で創造的な出力が得られる。
- Top-p 設定は、モデルによって考慮される可能性の高い単語の数を決定します。高い値を設定すると、可能性の低い単語も含め、より幅広い可能性を検討できるようになり、生成されるテキストがより多様になります。
- Top-k の設定を調整することで、回答の反復性や複雑さ、特に語彙や言い回しに影響を与える。

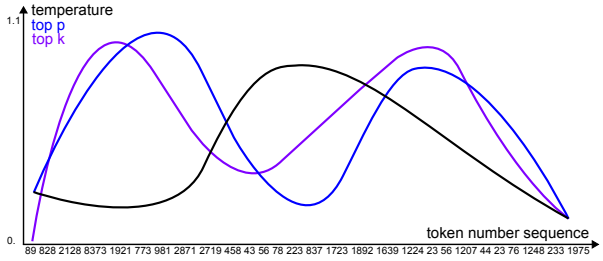

Figure 3. 補間ブレークポイント関数(BPF)は、トークンのシーケンス全体にわたって、温度、トップp、トップkを制御し、アーティキュレーションと時間的変化を可能にする有効性を実証した。これらのパラメータを相関させるために3dシンプレクスノイズを使用すると、より高い一貫性が得られます。

私たちはブレイクポイント関数（BPF）を使ってこれらの値を制御する方法を実装した (fig. 3)。 これは、一文を通して表現の強弱を変化させるのに役立つ。
Ableton LiveのBeat Repeat機能にあるようなランダム・クォンタイズ・エンジンをいくつか追加した [12]。この方法でトークンをランダムに繰り返すと、グラニュラー・シンセシスのように聞こえる。しかし、このようなトランスフォーマーを前述のような温度で使用すると、よりダイナミックで人間的な出力が得られる。粒を直接連結するだけに比べ、より大きなコントロールと表現力が得られる。音素と音素の間のアーティキュレーションは、自然さを目指しており、時に、以下のような加法補間のテイストを想起させるかもしれない*Diphone* [13, 14]。

ランダム性は、セレンディピティな作曲実験を容易にする上で貴重な役割を果たすが、ニュアンスに富んだ音楽シーケンスの作曲を可能にするには至らない。不連続なトークンシーケンス生成の制御性を高める最善の方法は、単純にマルコフ的なテクニックを使うことである。

3. TOKEN エンジニアリング

興味深い結果を得るためには、必要以上に多くの文を生成し、確率を使って整理できる大量のトークンを生成する必要がある。そこで、Ollamaのモデルを用いて、十分な数の共通の単語、音素、意味を持つ言い換え文の集合を生成する。[7]- 私たちは、主に次のことを要求している。Mistralを広く使用している。Retrieval-Augmented Generation (RAG) テクニック.} この一貫性により、より意味のあるテキスト出力が可能になり、発明言語の作成という当初の目標に近づくことができる。現在の言語・音声合成システムが、確率とニューラルネットワークを併用しているという事実は、歴史的な観点から見ても興味深いものだ [15]。

Hack 3: LZ token デコーディング

私たちはその後、マルチスカラーLempel-Ziv-Welch（LZ）圧縮を採用し、ウィンドウウィングが重要な役割を果たしていることを認識した。LZエンコーディングアルゴリズムは、可変サイズのトークンのシーケンスを辞書にマッピングし、その辞書への参照を文字列として出力することで圧縮します [16]。

続いて、文字列から任意の長さのシーケンスをデコードし、連結シーケンスを無限に生成することができる [17]。文字列に重み付けされた確率を持つLZを使用することは、声の意味と抽象度のレベルを調整する最も効果的な方法であることが実証されている。

LZは繰り返しパターンを含むデータに対して最適に機能するため、言い換え文に適している。LZはマルチスカラーであるため、LZ辞書から特定のシーケンス長を選択し、モデルのコンテキストウィンドウで遊ぶことができる (fig. 4)。

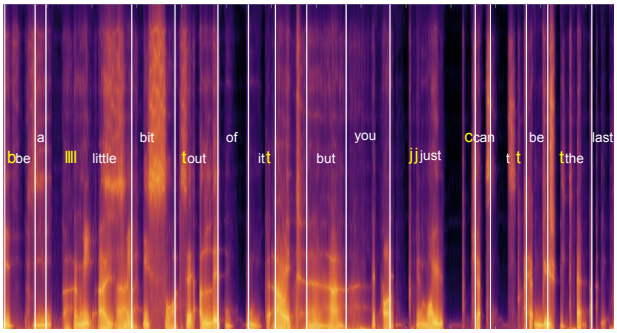

Figure 4. 使用例：特定の意味トークン（黄色）の偶発的な繰り返しに重なるLZシーケンス（白色）を使用した英語の半単語と半文の生成。

Hack 4: 音響特性シーケンス

現在は、モデルの出力から特定の音やパターンを抽出できるように、トークンのセグメンテーションを実現することを目指しています。自己回帰変換器固有の予測不可能性、特に自分自身でトレーニングしていないモデルを使用する場合は、モデルの動作を徹底的に分析する必要があります：\モデルを使用する前にクエリーを行います。不確実性を最小化するために、低温設定と一意のシード番号 (fig. 5)。

まず、言い換えられた文の包括的なデータセットから分析を開始し、Flucoma Pythonライブラリを用い

て音特徴抽出を行い、簡単な自作のk-meansクラスタリングを行う [18, 19]。Barkモデルの多言語機能により、音色の多様性がさらに高まります。具体的には、出力から得られるピッチシーケンスとメル周

[7] https://ollama.com

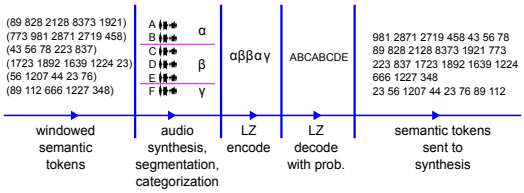

Figure 5. Flucomaの記述子を使用したBarkモデルの動作の音声解析：モデルを使用する前に問い合わせを行う。不確実性を最小化するため、低温設定と一意のシード番号を使用した決定論的戦略を採用しています。

波数セプストラル係数（MFCC）を分析し、トークンのピッチベースまたは音色ベースのクラスタリングを行うことができます。[8]そして Python Flucoma このアプローチは、トークン列のセグメンテーションに非常に有効である (fig. 6)。次に、トークンをクエリし、モデルから推論するために、前述のシンプルで効果的な重み付けLZ法を使用します。最終的に、母音、摩擦音、鼻音、過渡音などの音声記述子の連続を含むシーケンスを、再帰を用いてアルゴリズム的に構成することができます (fig. 7)。
当面は、音声合成を使うという最初のアイデアに焦点を絞る。しかし、トークンチェインから得られる音を分析することで、連結合成のように、推論を導き、多様な音のターゲットに収束させることもできる。

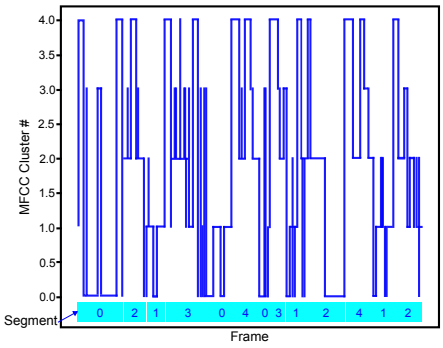

Figure 6. マルコフ法でトークンとして使用する前の、MFCCセグメントの単純なk平均クラスタリング。ここでは、視認性を高めるために、より大きなデータセットにズームしている。それでも、クラスタの量が合成音声の明瞭度に大きな影響を与えることは明らかです。

Hack 5: を使ったロングジェネレーション fine-tuning

Barkの音声の最大長は、約15秒に相当する756セマンティックトークンであることが確認された。一方、粗音響モデル変換器にはそのような制限はない。次に、後者にセマンティックトークンのシーケンスを再帰的に与え、以前の反復と同じ音響特性を再現できることを確認した。まず、粗音響モデルを微調整した。しかし、以下のパラメータを使用して3つのモデルすべてを微調整すると、より良い結果が得られました。*history-prompt*. 3つのモデルを同時にチューニングすることで、特定の俳優や歌手の個性をターゲットにすることもできる。(deepfake).
我々は、潜在空間からの推論を所望のターゲットに導くことによって、長いトークン集合を動的に制御する手法を設計した。この経路は、過去のトークンの確率の重みを制御し、適切な履歴-プロンプトを選択し、新規性と命題を強化するために温度を操作することによって作られる。また、テキストによるプロンプトと、上述したサウンドデスクリプタ法の両方を用いて、長く構成されたシーケンスの展開を自動化することもできる：

1. 予測範囲を広げるため、比較的高い温度を保ちながらシステムを反復する。
2. 音の特性が満足のいく目標に達したら、その結果をヒストリ・プロンプトとして使用し、同じシード番号を保持して次のステップで再投入する。
3. 今回はシードとヒストリー・プロンプトを使って、低温で新バージョンを生成する。
4. 段階を追って徐々に温度を上げ、サウンドディスクリプタの分析によって自動的に定義された新しいターゲットに到達する。
5. 2.から繰り返す。

上記で生成された各バージョンは、独立して使用することも、ポリフォ
ニーを作成するために一緒に使用することもできます。また、この相互作用を利用して、曲の特定の瞬間に収束するポイントをコントロールすることで、ヒップホップのステムを生成したこともある。 ウィンドウの最適なサイズと、シス テムの入力にあるトークン群を選択することで、このアプローチは、声の劇的な激しさとジャンルが、謎めいた蛇行へと溶解し、不穏な見当識障害を生み出す、何とも言えない集合体を生成する。このアプローチは、ドラマチックな視点から声を使うことの力を浮き彫りにしている。

[8] https://www.flucoma.org

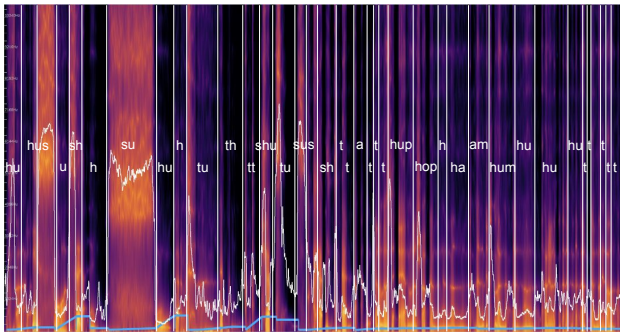

Figure 7. LPCスペクトログラム *hu*, *tu*, *shu*... 粗音響モデルに温度を導入することで、より多様性が増す。また、入力トークンによって与えられた音素に関連する音素を追加することで、シーケンスの繰り返しを避けることもできる。最初に音声として明瞭化されたこれらの音素は、 ”electro-acoustic music’s type” アーティキュレーテッド・トランジション」に非常によく似ている。このフレーズの強弱と抑揚をよりよく視覚化するために、青字のピッチ曲線と白字の重心を追加した。青色で示された1つのピッチに収束している端に注目してください。

6. 未来と音楽への応用

私たちは、ワーナー・ミュージックの許可を得て、ポピュラー音楽歌手のためのリミックス制作にこれらの技術を用いました。これらの手法は、2025年9月にT2Gナショナル・シアターで初演されるジョン・フォッセの戯曲『そして、私たちは決して別れない』のミュージカル英語版および演劇フランス語版にも広く使用される予定である。\\
GPT-2のようなトランスフォーマーを使った部分は、私たちのニーズを満たしている。しかし、大規模なパーソナルBarkモデルのトレーニングは、不可能ではないにせよ、困難であることが判明した。近い将来、より少ない外部ライブラリを用いてワークフローを簡素化し、システム全体をFairSeq 2に簡単に移植する予定である。低ランク適応（Low-Rank Adaptation: LoRA）を使って、より大規模なモデルで多様性とスタイルを向上させたい [21]。また、グラフィックの安定した拡散ツールでやっているように、これらのモデルを簡単にマージすることもできるはずだ。

7. 結論

トランスフォーマーベースのTTS合成と機械学習をプロダクションに統合することで、テキストとサウンド、直訳と抽象の間の創造的な表現が可能になります。音声記述子によって制御されるトークンエンジニアリングと、微調整モデルを用いて、ユニークで長いボーカルシーケンスを生成することができます。この研究は、直感的な制御を実現するために単純な概念を使用することの有用性を示している。トランスフォーマーを利用することは、当初は新規性や創造性にとって逆効果に思えるかもしれない。しかし、パラメトリックなプロセスを統合することで、演技のみに由来するものとは異なる、予期せぬテキストやサウンドの驚きを実現することができる。私たちは、テキストと音楽をシームレスに統合し、パーソナライズされた独自の方法で作り上げることで、外部からコントロールされたオンラインプラットフォームや製品に完全に依存することなく、クリエイティブな独立性とオリジナリティを確保することができます。

すべてのシーケンスとオーディオ例を含むJupyterノートブックが利用可能です。`GitHub` はこちら}.

REFERENCES

1. M. Woo, *“L’interprétation musicale des phonèmes, des gestes et des images dans Machinations de Georges Aperghis”*, 2011.

2. P. Shapiro and I. Lee, *“Modulations: a history of electronic music: throbbing words on sound”*, 2000.

3. A. Gayraud, R. Mackay, *“D. Miller, and N. Power, Dialectic of Pop”*, 2019.

4. R. Rubin and N. Strauss, *“The Creative Act: A Way of Being”*, 2023.

5. L. Xue, K. Song, D. Wu, X. Tan, N. L. Zhang, T. Qin, W.-Q. Zhang, and T.-Y. Liu, *“DeepRapper: Neural Rap Generation with Rhyme and Rhythm Modeling”* 2021.

6. C. Durt and T. Fuchs, *“Large Language Models and the Patterns of Human Language Use”*, 2024.

7. F. Kreuk, A. Polyak, J. Copet, E. Kharitonov, T.-A. Nguyen, M. Rivi` ere, W.-N. Hsu, A. Mohamed, E. Dupoux, and Y. Adi, *“Textless Speech Emotion Conversion using Discrete and Decomposed Representations”*, 2022.

8. L. Barrault, P.-A. Duquenne, M. Elbayad, and A. Kozhevnikov, *“Large Concept Models - Language Modeling in a Sentence Representation Space”*, 2024.

9. A. Harker, *“FrameLib: Audio DSP using Frames of Arbitrary Length and Timing”*, 2016.

10. N. Schnell and D. Schwarz, *“Gabor, Multi-representation Real-Time Analysis/Synthesis”*, 2005.

11. N. Obin, *“Cries and Whispers - Classification of Vocal Effort in Expressive Speech”*, 2012.

12. J. Kammerer, *"Unleashing Creativity with Ableton's Beat Repeat: A Comprehensive Guide"*, 2014.

13. G. Loizillon, *Diphone Studio*. Ircam, 1999.

14. J. Bachan, "*Efficient Diphone Database Creation for MBROLA, a Multilingual Speech Synthesiser"*. Institute of Linguistics, Adam Mickiewicz University, 2010.

15. R. Shwartz-Ziv and Y. LeCun, *"To Compress or Not to Compress- Self-Supervised Learning and Information Theory: A Review"*, 2023.

16. J. Ziv and A. Lempel, *"Compression of Individual Sequences via Variable-Rate Coding"*, 1978.

17. O. Lartillot, "*OpenMusic LZ 2.2 Library"*, 2001.

18. P. A. Tremblay, O. Green, G. Roma, and A. Harker, *"From Collections to Corpora: Exploring Sounds through Fluid Decomposition"*, 2019.

19. T. Moore, J. Bradbury, and P. A. Trembly, *"FluCoMa for Pedagogues"*, 2022.

20. B. Hackbarth, N. Schnell, P. Esling, and D. Schwarz, *"Composing Morphology: Concatenative Synthesis as an Intuitive Medium for Prescribing Sound in Time."* Contemporary Music Review, Vol. 32, No. 1, 49-59. 2013.

21. C. V. Nguyen, X. Shen, R. Aponte, Y. Xia, S. Basu, Z. Hu, J. Chen, M. Parmar, S. Kunapuli, J. Barrow, J. Wu, A. Singh, Y. Wang, J. Gu, F. Dernoncourt, N. K. Ahmed, N. Lipka, R. Zhang, X. Chen, T. Yu, S. Kim, H. Deilamsalehy, N. Park, M. Rimer, Z. Zhang, H. Yang, R. A. Rossi, and T. H. Nguyen, *"A Survey of Small Language Models"*, 2024.