Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions

Alexandru Meterez

1 Introducere

Problema de transformare a textului in vorbire (TTS - text to speech) are la baza o problema inversa: transformarea dintr-o sursa de date comprimata - textul, intr-una decomprimata - sunetul. Fiind o compresie lossy, din cauza multor variatii in pronuntie a unui cuvant sau grup de cuvinte, se pune problema unei transformari cat mai naturale.

In trecut, procedeele TTS erau complicate si necesitau diverse feature-uri extrase manual. Pentru asta, erau necesare cunostiinte de lingvistica si experti in domeniu, pentru a putea obtine rezultate.

2 Tehnici TTS

Primele metode de a genera sunet din text se numeau:

- *Concatenative TTS* - concatenarea mai multor mici clipuri audio ce reprezinta anumite cuvinte din text. Sunetul este inteligibil, dar nenatural din cauza discontinuitatilor la lipire
- *Parametric TTS* - se bazeaza pe extragerea de feature-uri lingvistice (foneme, duratii, formante etc.) si analizarea lor de catre un vocoder. Sunetul obtinut nu suna uman de cele mai multe ori, fiind rezultatul unor parametrii ce nu reflecta toate inflexiunile vocii

Deoarece nu s-au obtinut rezultate satisfacatoare cu cele doua metode, s-a trecut la metode generative bazate pe Deep Learning.

Primul model dezvoltat de Google capabil sa genereze voce umana din text se numeste *WaveNet* (Oord et al. [2016]). Acesta a venit cu un set de avantaje si dezavantaje. In urma experimentelor, output-ul obtinut in urma task-ului de TTS suna uman si fidel. Pe de alta parte, acest model are dezavantaje atat la antrenare, cat si la inferenta. Fiind un model autoregresiv, fiecare sample este conditionat de sample-urile generate la momentele de timp anterioare:

$$p(\mathbf{x}) = \prod_{t=1}^{T} (x_t | x_1, ..., x_{t-1}) \tag{1}$$

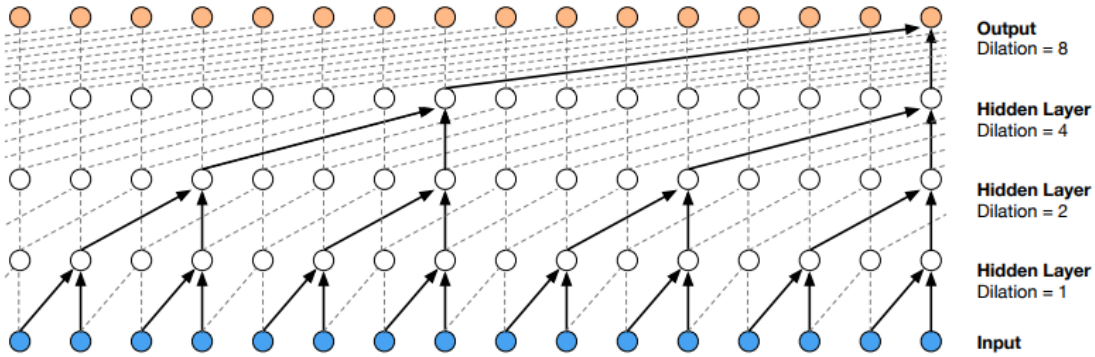

Figura 1: WaveNet, modelul folosit de Oord et al. [2016]

Antrenarea se poatea efectua rapid in paralel deoarece se stiu ground truth-urile. In schimb, la inferenta, pentru a genera un singur sample, este nevoie de sample-ul anterior (ultimele 1024 de sample-uri, acesta fiind campul receptiv al retelei), devenind un proces secvential. De asemenea, modul de functionare nu este unul end-to-end. Pentru a antrena modelul, este necesar un frontend lingvistic care sa extraga feature-uri din text, ce vor conditiona (vor fi concatenate) input-ul audio. Acest aspect nu este dorit deoarece frontend-ul lingvistic necesita cunostinte avansate de lingvistica, fiind greu de folosit.

O rezolvare a problemei a fost adusa tot de Google prin modelul *Tacotron* (Wang et al. [2017]). Creatorii s-au concentrat pe un model end-to-end, fara necesitatea unui frontend lingvistic sau a altor feature-uri extrase manual. Modelul functioneaza pe perechi de forma *(text, audio)* si este mult mai rapid decat *WaveNet*. Spre deosebire de *WaveNet*, acest model foloseste o structura de tipul Encoder-Decoder cu Attention, fiind un model seq2seq. Textul este primit ca input si transformat in embedding-uri la nivel de caracter. Odata ce tot textul trece prin encoder si se obtine starile hidden, incepe decodarea. La fiecare pas de de decodare se aplica un layer de attention. Mecanismul de attention functioneaza pe baza compararii intre hidden-ul decoder-ului si fiecare hidden din encoder (Bahdanau et al. [2014]). Spre deosebire de modelul anterior, *Tacotron* scoate la fiecare pas de decodare **r** frame-uri (spre deosebire de 1 ca WaveNet-ul), facand astfel de **r** ori mai putini pasi. Target-ul retelei este spectrograma Mel, fiind folosita astfel deoarece o spectrograma in scara liniara ar oferi prea multa informatie redundanta. Aceste feature-uri Mel sunt trecute printr-un submodul neuronal ce produce o spectrograma liniara. In continuare, pentru a reface sunetul din aceasta reprezentare, se foloseste algoritmul Griffin-Lim (Griffin and Lim [1984]), detaliat mai jos.

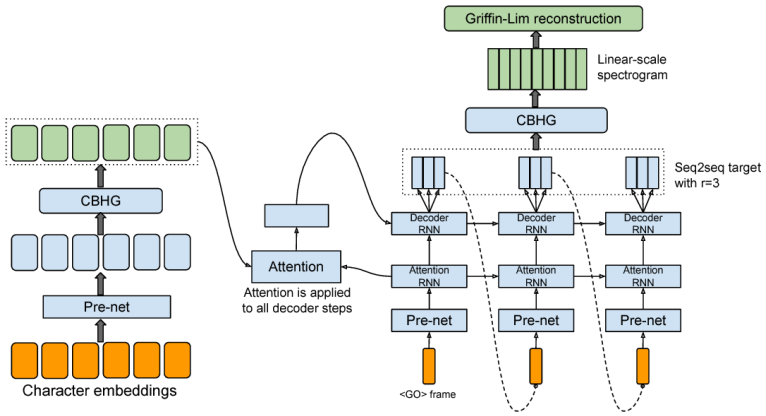

Figura 2: Arhitectura modelului Tacotron

3 Descrierea proiectului

Proiectul se bazeaza pe modelul ce a urmat dupa Tacotron, anume *Tacotron 2* (Shen et al. [2018]). Spre deosebire de modelul anterior, acesta are o structura mai simpla. In continuare se mentine arhitectura seq2seq cu attention, dar cu cateva diferente. Mecanismul de attention folosit se foloseste de CNN-uri, plecand de la ipoteza ca frame-urile alaturate nu sunt independente. De asemenea, s-a introdus un nou layer linear, ce prezice un stop token. Daca probabilitatea acestuia sare de 0.5, atunci modelul se va opri din procesul de generare la inferenta. Astfel, modelul poate decide, dinamic, cand sa se opreasca din generare. De asemenea, se observa o schimbare a vocoder-ului. Pe cand Tacotron se folosea de algoritmul Griffin-Lim, acesta foloseste ca vocoder WaveNet.

4 Implementare

Implementarea a fost realizata in Python, folosind ca biblioteci PyTorch (pentru modele) si Librosa (pentru procesarile de sunet). Pentru simplificare, am renuntat la unul din cele doua loss-uri de la iesire si la predictia de stop token. De asemenea, spre deosebire de paper-ul original ce foloseste WaveNet ca vocoder, implementarea mea foloseste Griffin-Lim pentru a extrage semnalul audio din spectrograma prezisa de model.

5 Rezultate

Din pacate, deoarece nu am avut resurse sa antrenez modelul, exista doar rezultate calitative, extrase in urma unui model preantrenat oferit de NVIDIA.

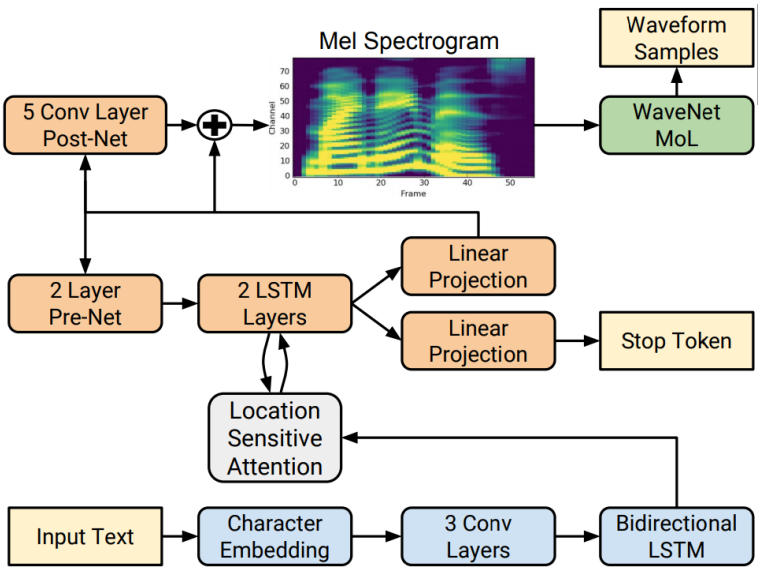

Figura 3: Arhitectura modelului Tacotron 2

6 Algoritmul Griffin-Lim

Algoritmul Griffin-Lim este un algoritm ce primeste ca intrare o spectrograma liniara si scoate la iesire un semnal audio pentru acea spectrograma. Pentru a obtine aceasta spectrograma se calculeaza un STFT (Short Time Fourier Transform) pe semnalul de la intrare. Astfel, este clar ca fiecare semnal in timp are o spectrograma unic determinata. Problema apare in sens invers. Deoarece o spectrograma este o masura a puterii frecventelor dintr-un interval de timp, aceasta pierde informatia de faza. Astfel, "decompresia" spectrogramei in semnal audio nu se poate face direct. Algoritmul Griffin-Lim este un proces iterativ, ce recupereaza, cu o eroare, semnalul audio din spectrograma:

- Se calculeaza $D = STFT(x)$
- Se inlocuieste magnitudinea lui D cu spectrograma initiala, pastrand insa faza lui D: $R = S * angle(D)$
- $x = ISTFT(R)$ si se reia

Prin acest algoritm se obtine spectrograma (si implicit semnalul) ce minimizeaza distanta fata de spectrograma initiala in sensul celor mai mici patrate.

Bibliografie

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Daniel Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243, 1984.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4779–4783. IEEE, 2018.

Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. Tacotron: Towards end-to-end speech synthesis. *arXiv preprint arXiv:1703.10135*, 2017.