Deep Neural Network based Text-to-Speech
Beszédszintézis Mély Neurális Hálóval

Tamas Szanto
tmas.szanto@gmail.com

Gergely D. Nemeth
NeGeD.NG@gmail.com

Abstract

Human-like communication with the computers is a major application of Computer Science. The Text-to-Speech methods are significant part of the project. This paper is a review of the authors experience about the usefulness of the Deep Neural Networks' new fields regarding to the **TTS** problem.

Abstract

A Számítástechnika egyik meghatározó célja a számítógépekkel való emberszerű kommunikáció elérése. A Beszédszintézis ennek elengedhetetlen területe. Jelen dokumentum a szerzők a Mély Neurális Hálók nyújtotta lehetőségek a témában való alkalmazásából szerzett tapasztalatainak összefoglalója.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

1 Bevezetés

A beszédszintézis célja, hogy beszélő emberi hangot hozzon létre mesterségesen. A gépekkel való emberi kommunikáció elengedhetetlen része. Napjainkban fontos felhasználási területe a vakok és gyengénlátók számára készült kisegítő lehetőségek gyártása.

1.1 Főbb szintézis technológiák

1.1.1 Összefűzéses szintézis

Ebben a módszerben előre felvett szegmenseket fűznek össze. A megfelelő szöveg(text)-darabokat megfelelő felvett hangdarabokkal párosítják, és ezeket egymás után mondatják ki a programmal.

1.1.2 Artikulációs szintézis

Ez a módszer alapvetően az emberi hangképzést hivatott lemodellezni.

1.1.3 HMM-alapú szintézis

A rejtett Markov-modell alapú szintézis lényege, hogy statisztikai alapon próbálják modellezni a beszéd különböző paramétereit, és minden kombinációra becsülik a helyességének valószínűségét. A legnagyobb valószínűségűt választva kapható meg a kívánt hang.

2 WaveNet tapasztalatok

Első célnak a beszédszintézis WaveNet[1] alapú megoldását tűztük ki. Ennek lényege az aktuális időpillanatban a hanghullám értekének az előző, már meghatározott értékek és a szövegfeldolgozásból kapott címkék segítségével történő meghatározása. Két rendszertervet dolgoztunk ki, azzal a különbséggel hogy a fonéma hosszának a becslése egy külön hálóban történik(1. ábra) vagy a kimenetről van visszavezetve(2. ábra). Utóbbi esetben a hullámérték mellett lenne egy másik kimeneti érték is, amely azt adná meg hogy kezdődjön-e az új fonéma.

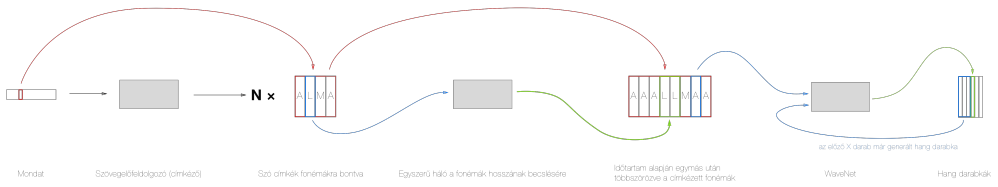

Figure 1: WaveNet külön fonéma hossz becslő hálóval

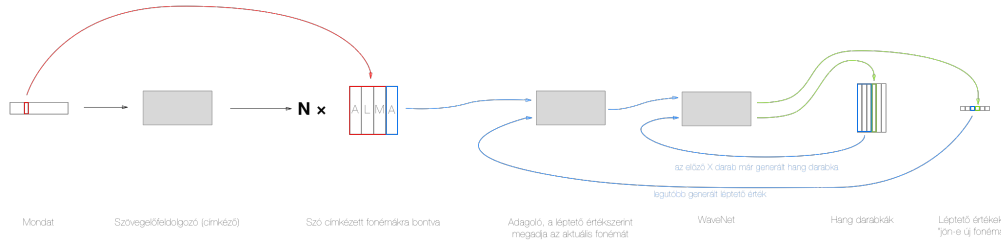

Figure 2: WaveNet fonéma hossz becsléssel kiegészítve

A megvalósítás során először már meglévő, GitHub-on elérhető implementációkat próbáltunk ki. [1] [2] [3] [4] Ezek közül az egyiket sikerült egy adott mondatra tanítanunk, azt vissza tudta generálni pontosan[8]. Azonban más mondatokkal való továbbtanítás esetén már nem volt működőképes.

Ezután saját implementációval próbálkoztunk a DeepMind által publikált cikk alapján[1]. Ez az implementáció lényegesen egyszerűbb volt mint a cikkben meghatározott, célunk csak valamilyen kis zörej előállítása volt. Azonban konstans (néma) hangon kívül mást nem sikerült előállítanunk.

A fent említett kísérletek után egyértelművé vált, hogy túl nagy feladatba kezdtünk bele. Ezért visszaléptünk az eredeti célunktól és folytatásképpen a hagyományos DNN alapú beszédszintézissel haladtunk tovább.

[1] https://github.com/tomlepaine/fast-wavenet
[2] https://github.com/ibab/tensorflow-wavenet
[3] https://github.com/basveeling/wavenet
[4] https://github.com/usernaamee/keras-wavenet

3 Egyszerű DNN model felépítése

DNN alapú rendszer esetén is az első fázis a szövegből fonéma alapú címkék előállítása, ezekről részletesebben következő fejezetben írtunk. Ezután meg kell határozni a fonémáknak a hosszát, erre egy egyszerű előrecsatolt háló alkalmas. Az idő paraméterekkel ellátott fonémákból becsülhetőek az aktuális időegységre jellemző gerjesztési és spektrális paraméterek. Ezekből a paraméterekből állítható elő az eredetihez hasonló gépi hang ([4] és a BME Deep Learning előadás során elhangzottak[5]).

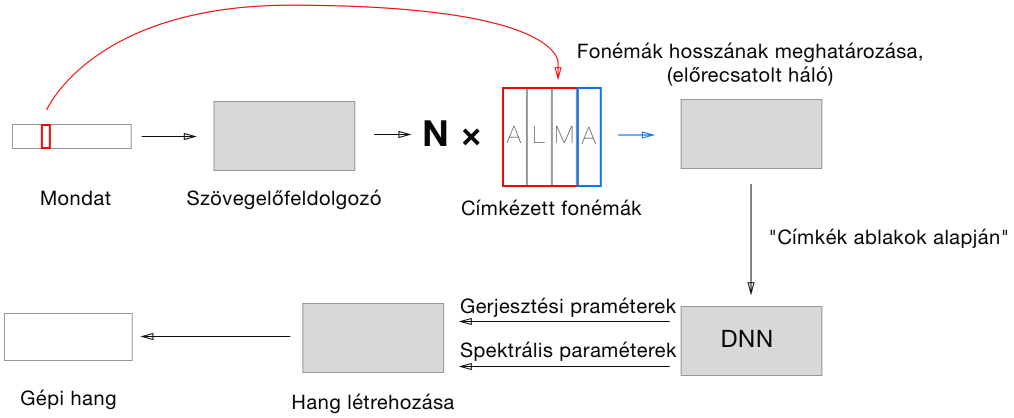

Figure 3: DNN rendszer

3.1 Háló modellek

Külön hálón tanítottuk a spektrális és a gerjesztési paramétereket.

Gerjesztési paraméter problémáját két hálóra bontottuk szét. Az egyik a hang zöngésségét határozta meg a másik pedig az alapfrekvenciát, a végleges eredményeket a két háló által jósolt értékekből számítottuk ki. (Gyakorlatilag egy engedélyező jelként értelmeztünk a zöngésség értékét az alapfrekvencián.) A zöngésség meghatározása egy egyszerűbb probléma erre egy 6 rejtett réteget, rétegenként 512 neuront tartalmazó, RELU aktivációs függvényt, SGD optimalizálást és MSE költségfüggvényt alkalmazó hálót hoztunk létre. Az alapfrekvencia becslésére 6 rejtett réteget, rétegenként 1024 neuront tartalmazó, tanh és sigmoid aktivációs függvényt, SGD optimalizációt és MSE költségfüggvényt használó hálót alkalmaztunk. Utóbbi esetén a rétegenként alkalmaztunk Dropout-ot, tanítás során pedig early stopping-ot.

A gerjesztési paramétereket meghatározó háló hasonlóan épült fel az alapfrekvenciát becslő hálóhoz.

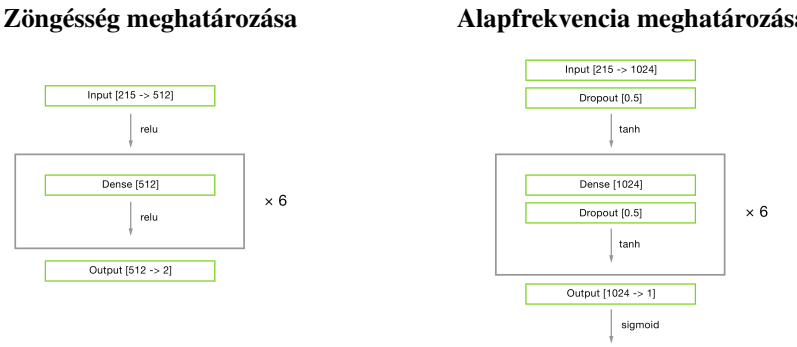

Figure 4: Zöngésség és alapfrekvencia hálók

[5] http://smartlab.tmit.bme.hu/oktatas-deep-learning-eloadas?eloadas=5

4 Bemeneti és kimeneti paraméterek

4.1 Szövegfüggő címkék

A címke generálás egységeként egy mondatot használtunk fel. Ezt az NLTK eszköz segítségével elemeztük majd az eredményekből állítottuk össze a címkéket[6]. A címkék alapjául az aktuális fonémák szolgáltak, később még ezek lettek továbbontva időkeret szerint.A fonéma azonosítására OneHot kódolást alkalmaztunk, azaz minden fonémát 40 értékkel azonosítottunk.

A következő szöveg függő címkéket hoztuk létre:

- 1-5 aktuális fonéma és az őt körülvevő két-két fonéma
- 6 hangsúly
- 7-8 megelőző és követő fonémák száma a szóban
- 9-10 távolság hangsúlyos fonémától mindkét irányban
- 11 szó szófaja
- 12-13 szó pozíciója a mondatban
- 14-16 fonémák száma az aktuális szóban és két szomszédjában
- 17 szavak száma a mondatban
- 18 fonémák száma a mondatban

4.2 Bemeneti paraméterek

A szövegfüggő fonéma címkéket egészítettük ki a fonémában található időkeretek számával és a keret fonémán belüli elhelyezkedésével. Így a kerethez tartozó 215 bemeneti paraméter az alábbiakból tevődik össze:

- 1-200 A 40-40 paraméter a kvinfón(2-1-2) fonémáira
- 201-213 további szövegfüggő címkék
- 214 A fonémán belüli időkeretek száma
- 215 A keret elhelyezkedése a fonémán belül

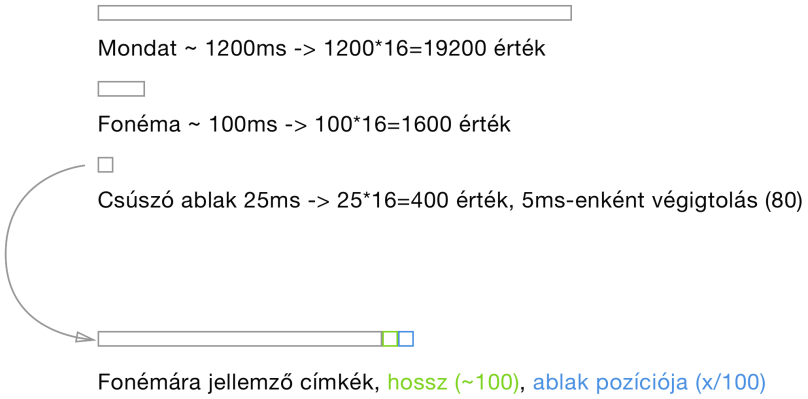

Figure 5: Címkék létrehozásának folyamata

4.3 Kimeneti paraméterek

Az előző felsorolást folytatva az aktuális időkerethez tartozó elvárt kimenet paraméterek a következők:

- 215 a keret hangmagasság értéke
- 216-242 a Mel-Cepstrum 26 paramétere
- 242 a fonémán belüli keretek száma

4.4 Spektrális és gerjesztési paraméterek

Mint említettük a prediktálás alapja a spektrális és gerjesztési paraméterek megadása keretenként. Ezen paraméterek segítségével a PyPSTK [6] python csomag használatával állíthatjuk elő az audio kimenetet, valamint hasonlóképpen ezt a csomagot használjuk az adataink a tiszta hangból való előállítására [2].

4.4.1 Mel-Cepstrum

A paraméterek előállításához, visszafejtéséhez a PyPSTK Mel-Cepstrum alapú algoritmusát alkalmazzuk. Az algoritmust T. Fukada és társai fejlesztették ki. [3]

Ezzel a metódussal természetesen adatvesztést kapunk, azonban az így előállított hang még megfelel a mércéinknek. Viszont az általunk generált kimenetek értékelésénél ezt figyelembe kell vennünk.

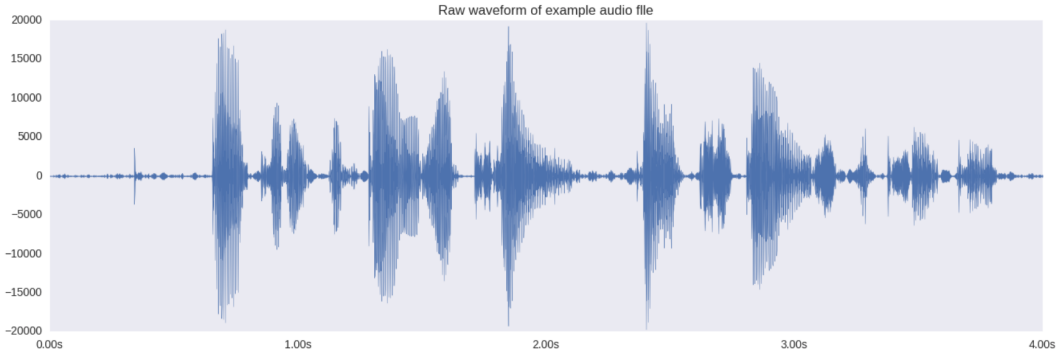

Figure 6: Eredeti hang

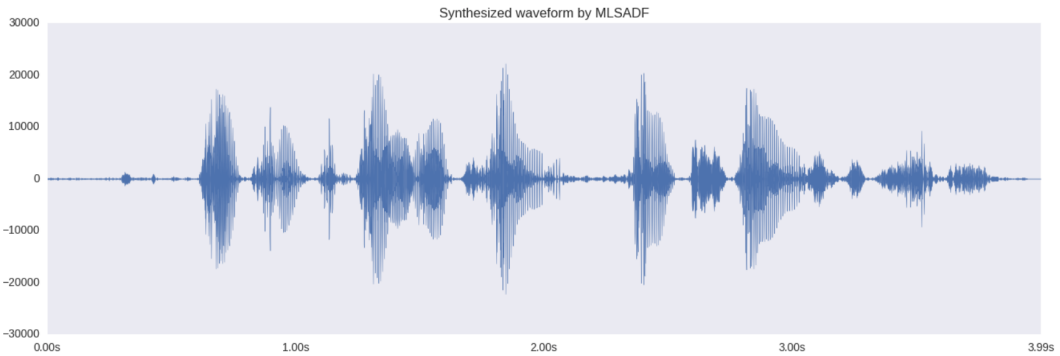

Figure 7: Mel-Cepstrum után elállítható hang

[6] http://pysptk.readthedocs.io/

5 Megvalósítás

5.1 Használt erőforrások

A háló összerakást saját gépen kezdtük el. Majd a véglegesítését és a tanításokat AWS szerveren, NVIDIA Tesla K80 12GB videokártyán végeztük. Jupyter notebook-ban futtatuk a tanításokat, kerasban felépített hálóval [5], ezek elmentett adatait aztán Tensorboard segítségével elemeztük [7].

5.2 Fejlesztés

A fejlesztés során scriptet írtunk az adatok generálására(*generate_dataset.py*). Ez állítja elő a tiszta adatfájlokból a hálók bemeneteit.

A különböző hálókat jupyter notebookokban teszteltük. Ezek a *model* kezdetű notebookokban találhatóak.

A végső eredményeket a *results* notebookban összegeztük, ez végigfuttatható demonstrációként hamar előáll.

5.3 Eredmények

Zöngésség becslésére nagyon jó, alapfrekvencia (pitch érték) meghatározására használhatóan jó és a spektrális paraméter (Mel-Cepstrum) jósolására kezdetleges eredményeket sikerült elérnünk. (utóbbiról ezért nem is mellékeltünk tanítási és validációs grafikonokat). Hiperparaméter optimalizálás tekintetében egyelőre kézzel dolgoztunk (ez még javításra szorul), már így is sikerült használható pontosságot elérnünk.

(A grafikonok y tengelyén az MSE hiba értéke x tengelyén az elvégzett epoch-ok száma látható.)

5.3.1 Zöngésség

Zöngésség becslése egyszerűbb, futásidő tekintetében gyorsabb feladatnak bizonyult. Ezért itt nem alkalmaztunk early stopping-ot, hanem a több próbálkozás után a fixen 410 epoch-kal történő tanításnál maradtunk.

A teszt adatainkon elért eredmények:

pontosság 0.8342

költségfüggvény: 0.1560

(MSE hiba értékekkel számítva)

Figure 8: Zöngésség tanítás

5.3.2 Alapfrekvencia

Az alapfrekvencia tanítása esetén alkalmaztunk early stoppig-ot. Ez azonban (a paramétereinek állítása után is) túl hamar állt meg. Ennek következtében több tanítást is elvégeztünk egymás után. Az itt kapott eredményeink messze nem olyan szépek mint a zöngésségé, de ezek is használhatónak bizonyultak.

A teszt adatainkon elért eredmények:

pontosság: 0.4243

költségfüggvény: 0.07214

(MSE hiba értékekkel számítva)

Figure 9: Pitch tanítás

5.3.3 MC paraméterek

MC tanításakor kiütközött, hogy a bemeneti paramétereink nagyon megegyezőek voltak fonémán belül, így darabos lett a tanítás. Ez a magasabb MC értékekre jelentős hibát okozott, így megpróbáltuk kevesebb mc érték használatát pontosítani.

A teszt adatainkon elért eredmények:

pontosság 0.2927

költségfüggvény: 0.2086

(MSE hiba értékekkel számítva)

5.3.4 Fonéma hossz

Fonéma hossz becslésnél csoportosítást valósítottunk meg, a fonémákat 8 csoportba osztottuk, a csoportok között 12,5 ms eltéréssel és a csoportra végeztünk classificationt. Erre azért volt szükség mivel viszonylag kevés, géppel annotált tanító adat állt a rendelkezésünkre, ezek alapján nem sikerült használható eredményeket elérni(1000 mondat, kb. 16000 fonéma ami átlagosan egy fonémára 400 mintát jelent, de ez nem egyenletes így előfordulhat olyan fonéma amire ennél jóval kevesebb tanítóadat szerepel).

Az adatokban szereplő eredeti hosszak vizsgálata után ezt a csoportosítási módot választottuk, két szempont a megfelelő differenciálódás és a jelentősen kilógó adatok levágása volt. Tehát minden fonémánk 40 és 140 ms közé került. (A tanító adatainkat is ez alapján módosítottuk még a feldolgozási fázisban, minden keretnél hosszabb fonémát 140 ms-nél levágunk.)

A teszt adatainkon elért eredmények:

pontosság 0.4619

költségfüggvény: 0.02419

(MSE hiba értékekkel számítva)

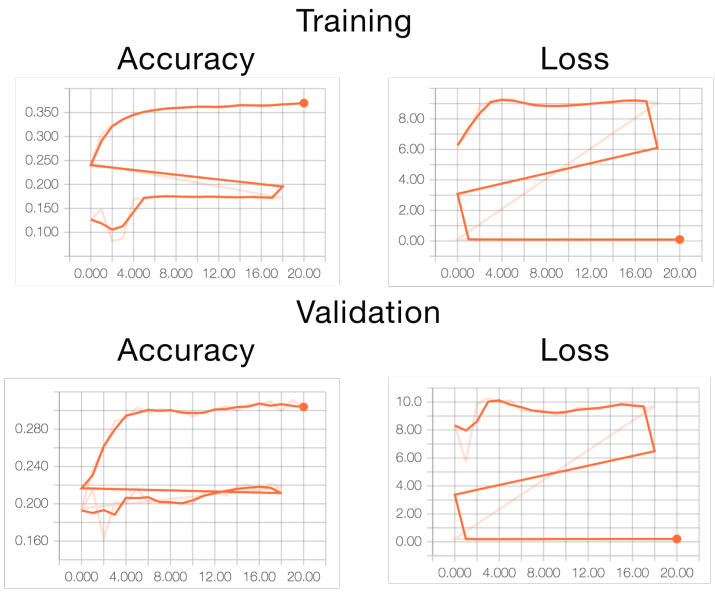

Figure 10: Fonéma hossz tanítás

5.3.5 Teljes becslő

A fenti paraméterek ismeretében teljes TTS végezhető.

A mondat alapján fonéma hosszakat becslünk, majd párhuzamosan zöngésség-alapfrekvencia és mc paraméter becslés folyik, végül a kapott eredményekből előállítható a kimenet.

5.4 Eredmények egy példán bemutatva

Az alábbi *"But all my dreams violated this law"* mondaton elvégezve a becslést:

5.4.1 Adatok beolvasása

Az adatok beolvasása egy eredeti hangfájlból (11.ábra), majd a csöndek szűrése és a pitch-MC paraméterek előállítása(12.ábra).

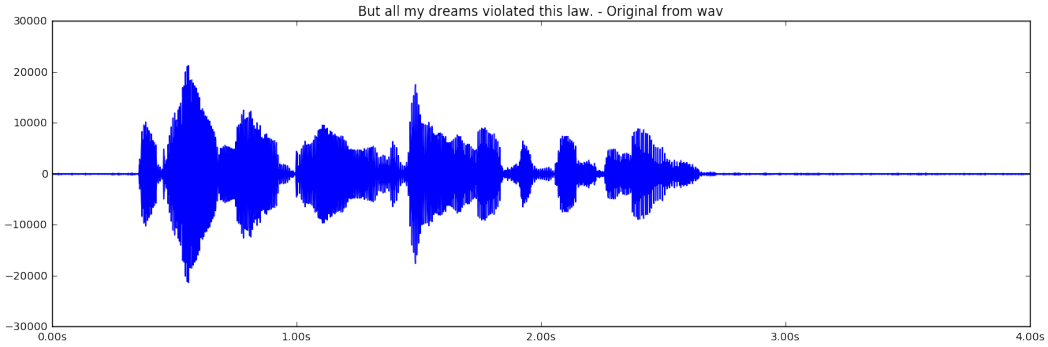

Figure 11: Beolvasott hangfájl

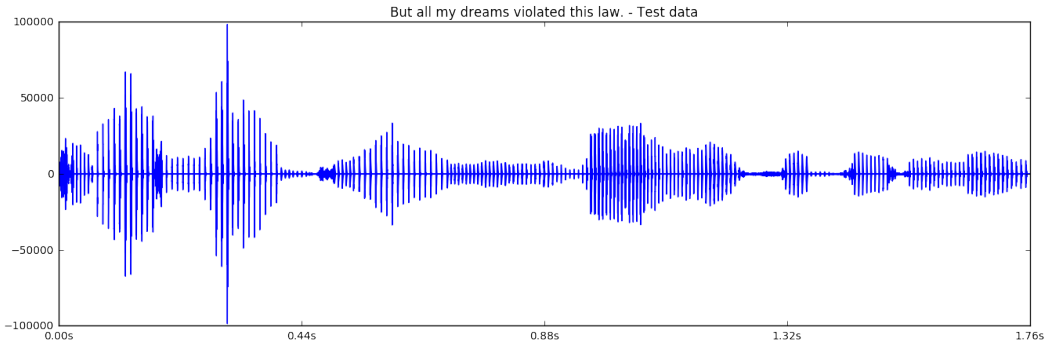

Figure 12: A hangfájl szűrve, MC-pitch értékek generálva PySPTK segítségével

5.4.2 Zöngésség becslés

A zöngésség becslése jól működik.

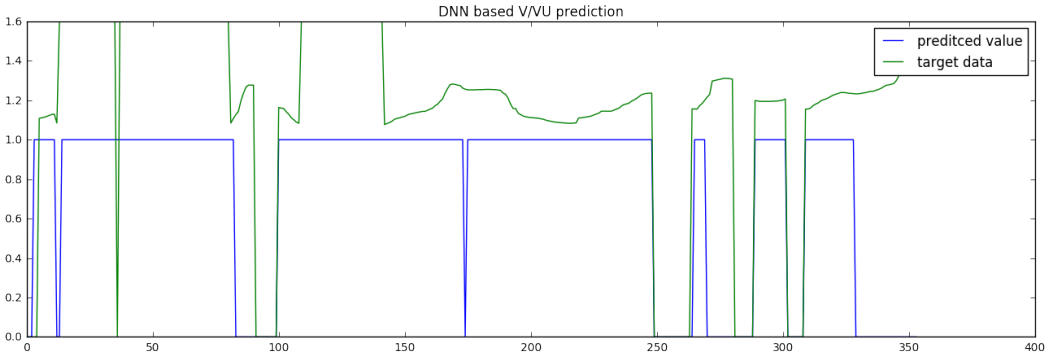

Figure 13: Beolvasott hangfájl

5.4.3 Pitch becslés

Alapfrekvencia becslése(14. ábra), majd kimenet előállítása a tanult gerjesztési és a hozott spektrális paraméterekkel(15. ábra). A 16. ábrán ugyanez látható a tanult zöngésség bevezetésével.

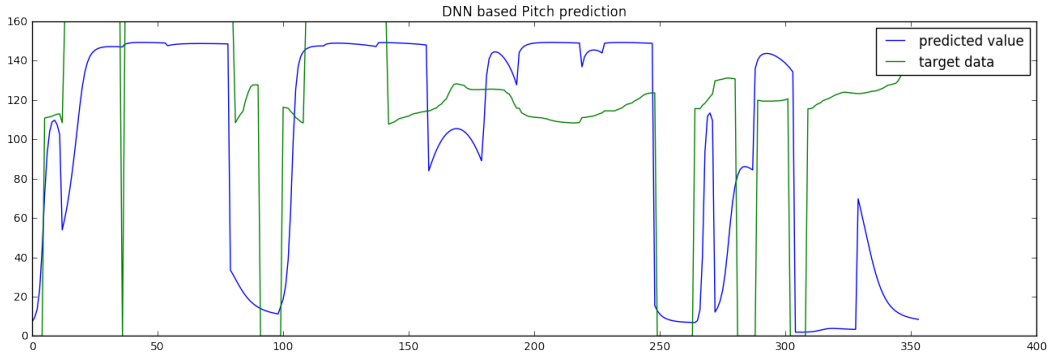

Figure 14: F0 becslése

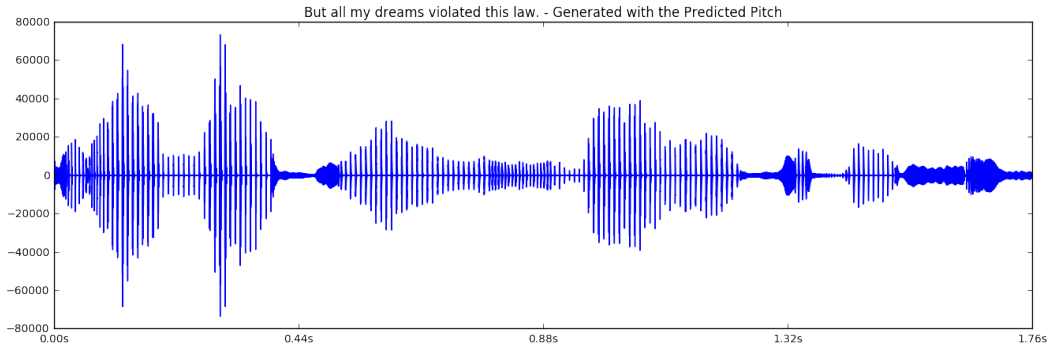

Figure 15: Generált hang F0 becsléssel, MC az eredeti hang alapján

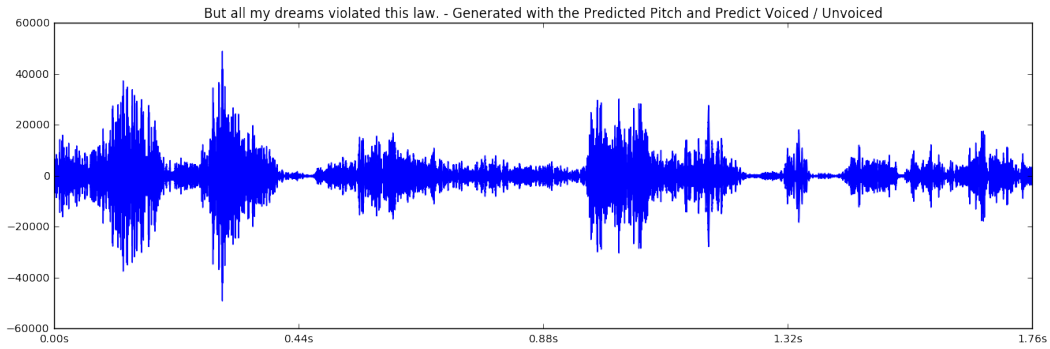

Figure 16: Generált hang F0 és V/UV értékek alapján

5.4.4 MC becslés

Az MC becslőnél a görbe sajnos meglehetősen lépcsősen illeszkedik (17. ábra), és ez a generált hangon is érezhető (18. ábra).

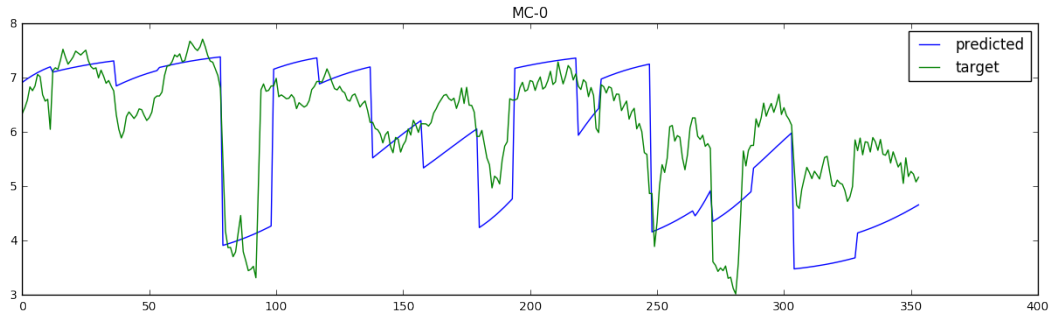

Figure 17: MC-0 becslése

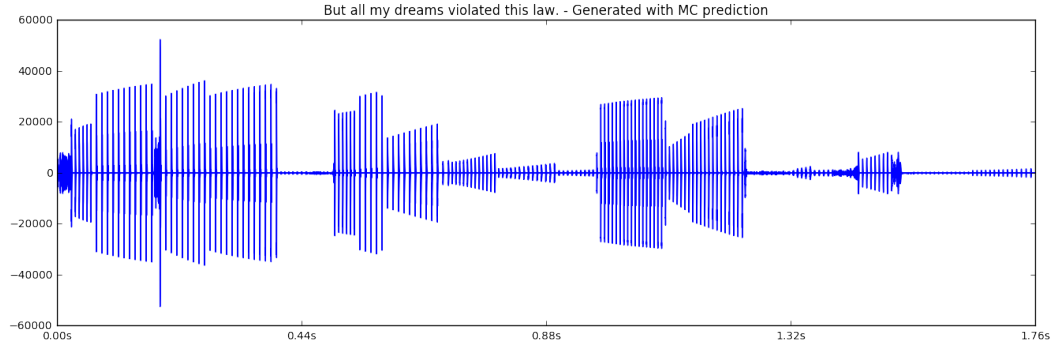

Figure 18: Generált hang hozott F0-val és becsült MC értékekkel

5.4.5 Hanggenerálás tanult F0 és MC értékekkel

A végleges generált hangból sajnos nem lehet kivenni az eredeti szavakat(19. ábra).

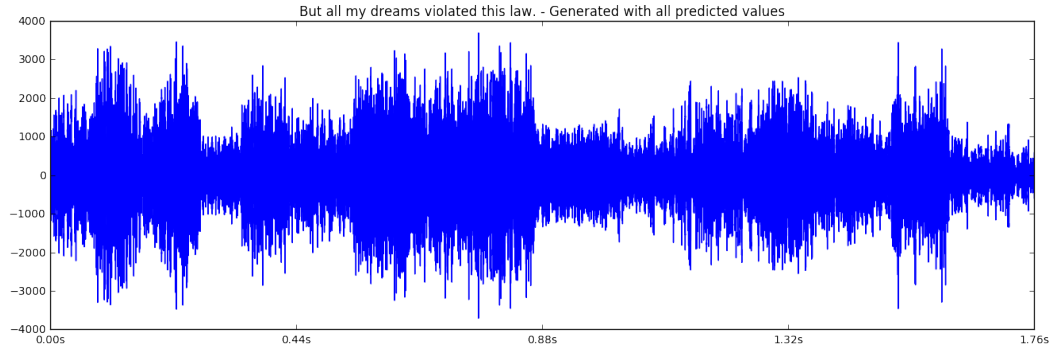

Figure 19: Generált hang becsült F0-val és MC értékekkel

6 További lehetőségek

Jelenleg van egy kész rendszerünk, amely bevitt szöveg alapján hangot állít elő. Minőségi problémák azonban vannak még az előállított hanggal. Ezen kétféleképpen tervezünk még javítani egyrészt további új adatokon való tanítással, valamint az előfeldolgozási folyamatunk javításával (több, kifejezőbb címke bevezetése). Ezen felül további minőségi javulást lehet elérni több spektrális paraméter alkalmazásával is. Minőség javítás mellett még további cél saját felhasználói felülettel rendelkező rendszer létrehozása. Itt többféle út is lehetséges egyrészt szerver kliens jellegű felépítés másrészt erőforrásigénytől függően egyedülálló rendszer.

A fenti bekezdés pontokban:

- minőség javítás új adatokkal
- címkézés felülvizsgálata, további spektrális paraméterek
- felhasználói felület fejlesztése

Hosszútávú célként említhető még a WaveNet jellegű implementáció további fejlesztése.

Irodalom

[1] van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., ... & Kavukcuoglu, K. (2016).*WaveNet: A generative model for raw audio.* arXiv preprint arXiv:1609.03499.

[2] SPTK Working Group. (2009) *Speech signal processing toolkit (SPTK).* http://sp-tk.sourceforge.net

[3] Fukada, T., Tokuda, K., Kobayashi, T. & S. Imai, (1992) *An adaptive algorithm for mel-cepstral analysis of speech* Proc. ICASSP-92 , pp.137–140, Mar. 1992.

[4] Qian, Y., Fan, Y., Hu, W., & Soong, F. K. (2014, May). *On the training aspects of deep neural network (DNN) for parametric TTS synthesis.* In 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 3829-3833). IEEE.

[5] Chollet, F. (2015). *Keras*. GitHub repository: https://github.com/fchollet/keras.

[6] Bird, S. (2006, July). *NLTK: the natural language toolkit.* In Proceedings of the COLING/ACL on Interactive presentation sessions (pp. 69-72). Association for Computational Linguistics.

[7] TensorBoard. Available at http://www.tensorflow.org/versions/r0.7/how_tos/summaries_and_tensorboard/index.html Accessed 25 April 2016.

[8] Paine, T. L., Khorrami, P., Chang, S., Zhang, Y., Ramachandran, P., Hasegawa-Johnson, M. A., & Huang, T. S. (2016). *Fast Wavenet Generation Algorithm.* arXiv preprint arXiv:1611.09482.

Köszönetnyilvánítás

A dokumentum és a szerzők munkái nagyban támaszkodnak a Budapesti Műszaki és Gazdaságtudományi Egyetemen tartott Deep Learning a gyakorlatban Python és LUA alapokon tárgy keretei között elhangzott információkra