Generare automata de muzica folosind Deep Learning

Autori: Stefan Giba, Adrian-Ioan Rudeanu
Grupa: 333AA

Cuprins:

1. Introducere

In ultimul deceniu, tehnologia a avut parte de o evolutie foarte rapida, ceea ce a dus la acumularea unui volum mare de date si prin urmare a ajutat la cresterea domeniului inteligentei artificiale.

Prin urmare, tot mai multe companii incearca sa adopte astfel de tehnologii si sa le implementeze in business-urile lor. Printre acestea putem numara, spre exemplu, serviciile de recomandari folosite de giganti tehnologici precum Netflix sau Google, tehnologii de procesare de imagine ce se bazeaza pe inteligenta artificiala de la Apple sau Google sau tehnologia de pilot automat foarte avansata din masinile produse de Tesla.

1.1 Motivatia alegerii temei

Principalul motiv pentru care am ales aceasta tema generarii automate de muzica este o pasiune comuna pentru atat pentru muzica, cat si pentru acest vast domeniu al inteligentei artificiale. De asemenea, o alta motivatie a fost capatarea de experienta in vederea unei posibile angajari in acest domeniu.

Consider ca subiectul generarii automate de muzica este unul relevant in demonstrarea puterii tehnologiilor de acest fel, intrucat crearea de muzica, care pana acum a fost considerata o capacitate intrinsec umana, poate fi realizata automat cu ajutorul unui computer.

1.2 Obiectivele generale ale proiectului

Am construit o retea neuronala de Deep Lerning ce primeste ca date de antrenament melodii cantate la pian compuse de catre fiinte umane. Modelul incearca sa imite capacitatea umana de a crea melodii originale, pe baza celor invatate din datele de antrenament. Am implementat modelul folosind RNN (Recurrent Neural Network), mai exact charRNN (character RNN), un model de RNN specializat pentru lucrul cu caractere. Acest tip de retea neuronala este bazat pe arhitectura de tip LSTM (Long Short Term Memory).

Nu ne asteptam ca modelul sa genereze muzica de o calitate profesionala, ci mai degraba o muzica de calitate decenta care sa fie melodioasa si placuta auzului.

1.3 Metodologia folosita

Primul pas facut spre realizarea proiectului a fost intelegerea partii teoretice si anume: ce este o retea neuronala, algoritmi de optimizare folositi in inteligenta artificiala si machine learning precum gradient si regresie logistica, cum este structurata muzica si metode de reprezentare a muzicii.

Apoi, prin parcurgerea de lucrari si articole realizate pe tema similara am facut un studiu asupra tehnologiilor ce pot fi folosite la implementarea unui astfel de proiect.

De asemenea, am ales si un mod de organizare a datelor care urmau sa fie oferite modelului pentru a invata, dar am incercat si in final ales si o anumit structura a straturilor pentru modelul de Deep Learning.

2. Considerente teoretice

Pana a detalia mai mult etapa de implementare, vom incerca sa explicam pe scurt o parte din terminologia folsita de-a lungul proiectului.

2.1 Retele Neuronale Recurente (RNN - Recurrent Neural Network)

Retele neuronale recurente (Recurrent Neural Network / RNN) sunt o clasa de retele neuronale folosite in domeniul inteligentei artificiale ce se folosesc de informatie secventiala, precum caracterele/cuvintele dintr-un text sau notele dintr-o melodie. Acestea sunt numite recurente deoarece acestea executa aceiasi functie/operatie pentru fiecare dintre elementele secventei oferite, iar rezultatul este dependent de toate celelalte calcule realizate anterior.

In acest proiect au fost folosite de fapt charRNN (character RNN), adica un Many to Many RNN (fig. 2.1), in care fiecare output corespunde cu toate input-urile la fiecare moment de timp. Acestea sunt specializate pentru lucrul cu caractere/cuvinte dintr-un vocabular si pot avea mai multe straturi ascunse de tip LSTM (Long short Term Memory).

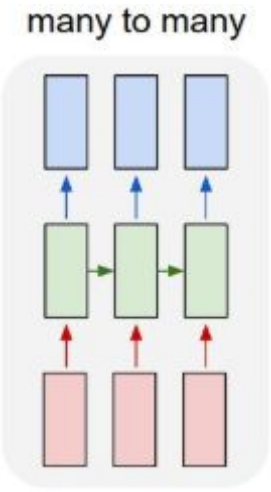


Fig 2.1
Reprezentare RNN Many to Many

2.2 Long Short Term Memory (LSTM)

Retelele neuronale de tip LSTM (Long Short Term Memory) sunt un tip de retea neuronala recurenta care pot invata foarte eficient folosind algoritmi de optimizare precum metoda gradientului. Acest tip de retea neuronala este foarte popular intrucat acestea sunt capabile sa retine dependinte pe termen lung, aspect util pentru rezolvarea unei varietati mari de probleme.

Retelele LSTM sunt foarte utile in cazul generarii de text sau generarii de muzica, intrucat acestea trebuie sa invete cum depinde un anumit caracter sau o nota muzicala de celelalte caractere din secventa oferita modelului.

2.3 Python

Python este un limbaj de programare interpretat de nivel iınalt. Acesta este axat pe usurinta de citire a codului si este caracterizat prin faptul ca nu utilizeaza acolade pentru delimitarea blocurilor de cod, ci se foloseste de indentare.

Principalul motiv pentru care am ales Python este varietatea uriasa de biblioteci. Putem enumera cateva biblioteci utilizate: numpy - algebra liniara, keras - API peste tensorflow utilizat pentru Deep Learning, json - pentru prelucrare de fisiere in format JSON, pandas - pentru manipularea si analiza datelor, tkinter - generarea unui GUI.

Am ales sa folosim Python 3.7.7, ultima versiune de Python disponibila in Anaconda3, o distributie de Python foarte populara in lumea Data Science datorita faptului ca vine preinstalat cu bibliotecile uzual folosite in acest scop si compatibilitate foarte mare cu acestea.

2.4 Keras

Acesta este unul dintre cele mai populare API-uri high level de retele neuronale. Acesta este scris in Python si are multe diverse arhitecturi de modele deja implementate. Acesta a fost creat in Python, ca o extensie pentru Tensorflow-ul celor de la Google si a fost gandit sa fie prietenos si usor de utilizat, astfel ca oricine poate experimenta diferite retele neuronale .

Un mare avantaj al Keras este de semenea faptul ca se pot descarca anumite seturi de date prin apelul unei simple functii, ceea ce usureaza foarte mult munca programatorilor, intrucat colectarea de date este poate cea ma complicata parte din realizarea unui proiect de Deep Learning.

2.5 Date in ABC-notation

Un alt aspect definitoriu pentru succesul acestui proiect au fost datele care trebuiau procesate de catre reteaua neuronala. Muzica se poate regasi in mai multe formate, printre care: portativ, ABC-notation, MIDI sau MP3.

Un bun exemplu de format de date este formatul MIDI (Musical Instrument Digital Interface), care, prin intermediul librariei Music21 poate fi adus intr-o forma care sa poata sa fie predata modelului de Deep Learning creat, dar nu asta este modul in care am ales sa implementam aceasta parte a proiectului.

Noi am ales sa preluam datele in formatul ABC-notation, deoarece formatul este deja unul text care este bazat pe codul ASCII. ABC-notation se foloseste de literele de la A la G pentru a reprezenta notele muzicale, si de alte elemente pentru a adauga proprietati notelor, spre exemplu lungimea notei, cheia si multe altele.

```
<score lang="ABC">
X:1
T:The Legacy Jig
M:6/8
L:1/8
R:jig
K:G
GFG BAB | gfg gab | GFG BAB | d2A AFD |
GFG BAB | gfg gab | age edB |1 dBA AFD :|2 dBA ABd |:
efe edB | dBA ABd | efe edB | gdB ABd |
efe edB | d2d def | gfe edB |1 dBA ABd :|2 dBA AFD |]
</score>
```

Fig 2.2
Exemplu de melodie scrisa in ABC-notation

2.6 Analiza unor realizari similare

Un prim exemplu este prezentat de Sigurour Skuli in articolul “How to Genenerate Music using a LTSM Neural Network in Keras” de pe www.towordsdatascience.com [6] ce foloseste o premisa similara, cea de a oferi modelului un punct de plecare, iar acesta sa incerce sa continue sa genereze continuarea acesteia. (fig 2.3)

O abordare similara, desi ce are ca tinta un stil de muzica diferit, este prezentata in articolul scris de Lee Surkis, “How to Generate Techno Music using Deep Learning” de pe www.medium.com [7]. Proiectele sunt asemanatoare in ideea in care ambele pleaca de la aceasi premisa, amandoua folosesc ca straturi principale ale modelului straturi LTSM si straturi “Dropout” si amandoua folosesc acelasi optimizator.

Diferentele apar in abordarea muzicii ca si obiect de studiu, si nu a tehnologiei folosite. Datorita scopurilor diferite ale acestor doua proiecte, desi similare ca implementare, au rezultate foarte diferite. Primul isi propune sa genereze muzica ca intreg, al doilea isi propune sa genereze un punct de plecare, o baza sau un artificiu ce va fii folosit intr-o melodie.

Date fiind cele doua proiecte, vom folosi ca punct de plecare pentru constructia modelului straturi de LTSM si ne vom propune sa genereaam muzica intr-o maniera similara, la oferirea unui punct de plecare, modelul sa genereze continuarea acestuia. Insa este de retinut faptul ca dezvoltarea in acest domeniu are mai degraba o natura empirica, si nu una exacta, aceste rezultate se pot modifica in functie de parametrii cadrului de testare.

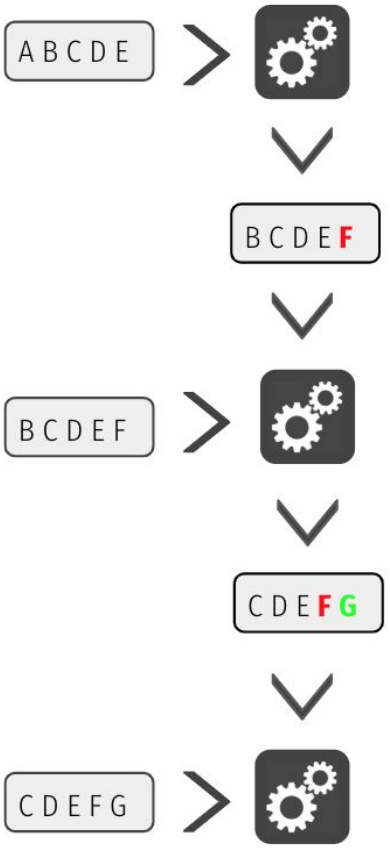


Fig 2.3
Prezentare a principiului de functionare

3. Prezentarea tehnica a implementarii

3.1 Prezentarea notiunilor teoretice

3.1.1 Retele neuronale

O **retea neuronala** este un program care urmareste sa functioneze similar creierului uman. Acestea au fost create cu scopul de a simula capacitatile cognitive intrinseci omului, ca de exemplu rezolvarea de probleme si capacitatea de a invata. Ganditi-va la o retea neuronala ca la un black-box care spre exemplu primeste ca input informatia provenita de la senzorii unei masini, proceseaza aceasta informatie si scoate ca output comenzi folosite la controlul masinii. Reteaua neuronala in sine este formata din o multiudine de mici entitati numite neuroni, acestia fiind grupati in mai multe straturi (**layers**). Straturile sunt coloane de neuroni interconectati.

Fiecare neuron este conectat la neuronul unui alt strat prin conectori numiti conexiuni ponderate (**weighted connections**). Weighted connections au asociat un numar real. Un neuron preia valoarea unui alt neuron conectat (din acelasi layer) si o inmulteste cu weight-ul conexiunii. Suma rezultata a tuturor neuronilor conectati se numeste **bias**. Bias-ul este mai

apoi trecut printr-o functie de activare care transforma matematic valoarea si o atribuie neuronului conectat din stratul adiacent. Aceasta valoare se propaga prin intreaga retea neuronala.

In esenta, reteaua este ca un filtru prin toate posibilitatile, astfel incat computer-ul sa poata veni cu o solutie corecta pentru problema tratata. Adevarata provocare in crearea si folosirea unei retele neuronale este de fapt gasirea weight-urilor necesare pentru ca output ul generat sa fie unul corect.

Dupa cum am mentionat si mai sus, tipul de retea neuronala folosit in implementarea acestui proiect este **RNN** (Recurrent Neural Network). Principalul avantaj al acestui tip de retea este faptul ca informatia persista in interiorul acesteia, intrucat acestea prezinta bucle asemanatoare cu binecunoscuta bucla de reglare de la Teoria Sistemelor.

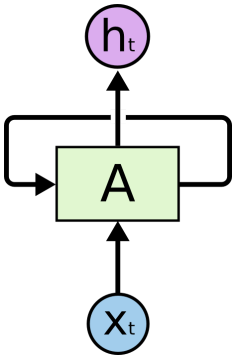


Fig 3.1
Ilustratie RNN cu bucla

In figura de mai sus se prezinta o bucata A dintr-o retea neuronala care primeste ca input o valoare Xt si genereaza ca output Ht. Bucla permite informatiei sa persiste de la un pas la altul. O astfel de retea neuronala poate sa fie gandita ca si o serie de copii ale aceleiasi retele, fiecare transmitand un mesaj succesorului.

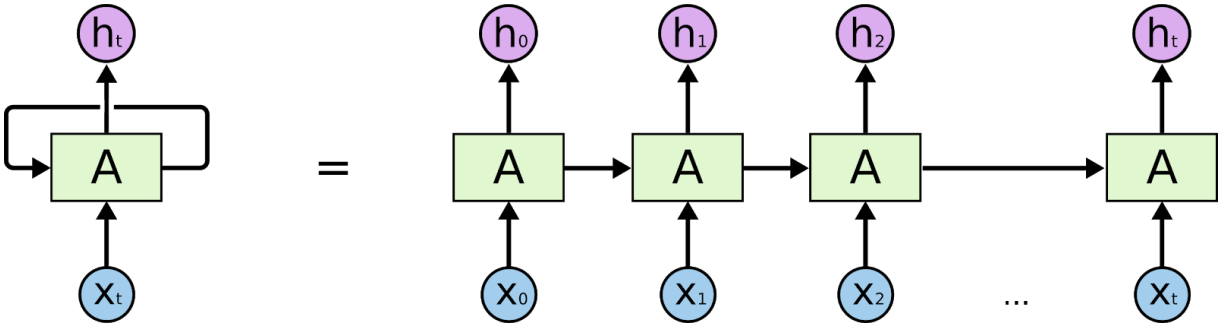


Fig 3.2
RNN desfasurat

3.1.2 Functii de activare

Functiile de activare au rolul de a adauga neliniaritati modelului intr-o retea neuronala. Printre cele mai folosite functii de activare se enumera:

Sigmoid

Este folosita pentru regresia liniara cu doua clase. Are avantajul ca este marginita de (0,1) si deci poate reprezenta distributii de probabilitati.

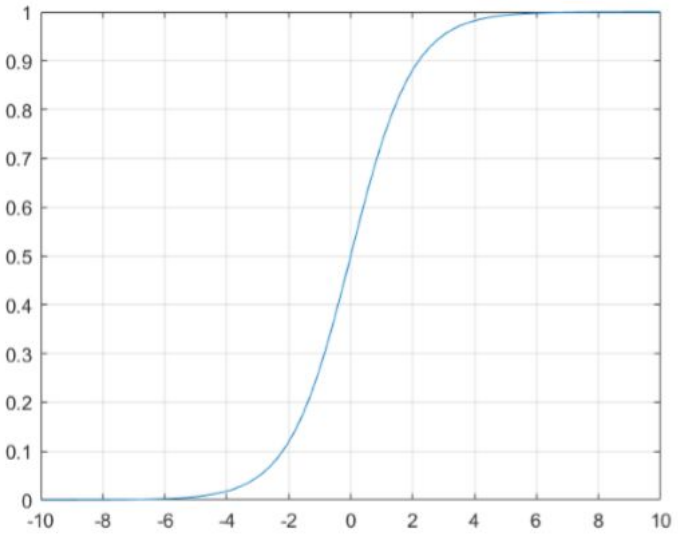


Fig 3.3
Reprezentare grafica Sigmoid

Softmax

O alternativa a functiei sigmoid este functia Softmax, folosita tot pentru regresii liniare dar are proprietatea ca la iesire are o distributie de probabilitati pe toate clasele exsistente, nu doar doua.

Tangenta hiperbolica

Tangenta hiperbolica (tanh) este asemanatoare functiei sigmoid, insa avand domeniul (-1,1) aceasta are un impact mai puternic pe rezultatele negative si deci polarizeaza mai mult rezultatele. (Fig 3.4)

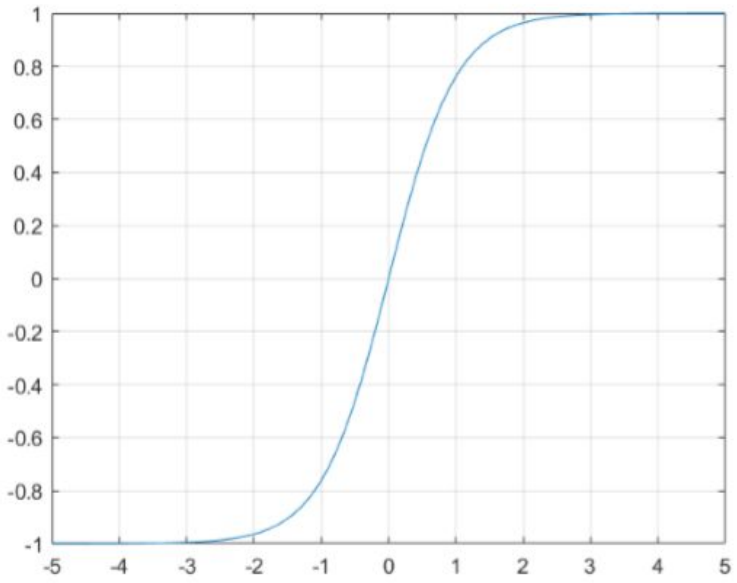


Fig 3.4
Reprezentare grafica Tangenta hiperbolica

ReLU - Rectified Liniar Unit

Aceasta este cea mai folosita functie de activare in acest domeniu din cauza performantelor bune in Retelele Neurale Convolutionale sau “Deep Learning”. Domeniul este de la 0 la infinit cea ce ofera o sensibilizare mai buna a valorilor pozitive, dar o ignorare completa a valorilor negative. Fig(3.5)

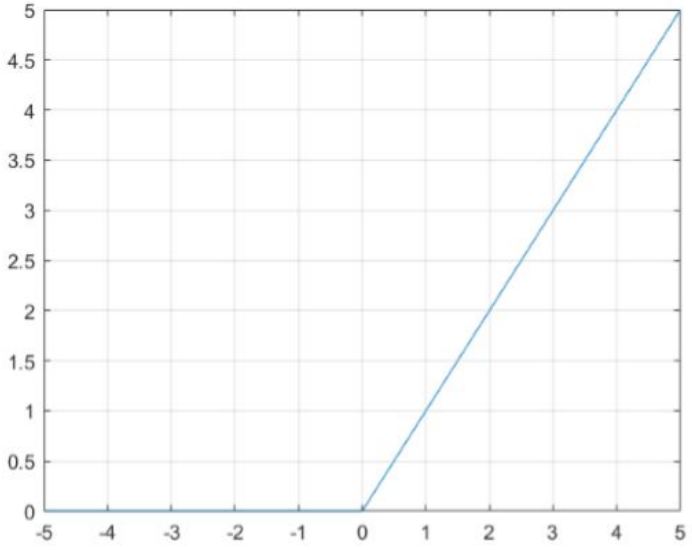


Fig 3.5
Reprezentare grafica ReLU

ReLU cu pierderi

In practica se foloseste deseori ReLU cu pierderi pentru a atenua dezavantajul descris mai sus. Acest lucru este realizat prin a oferii o panta negativa intervalului de la minus infinit la 0.

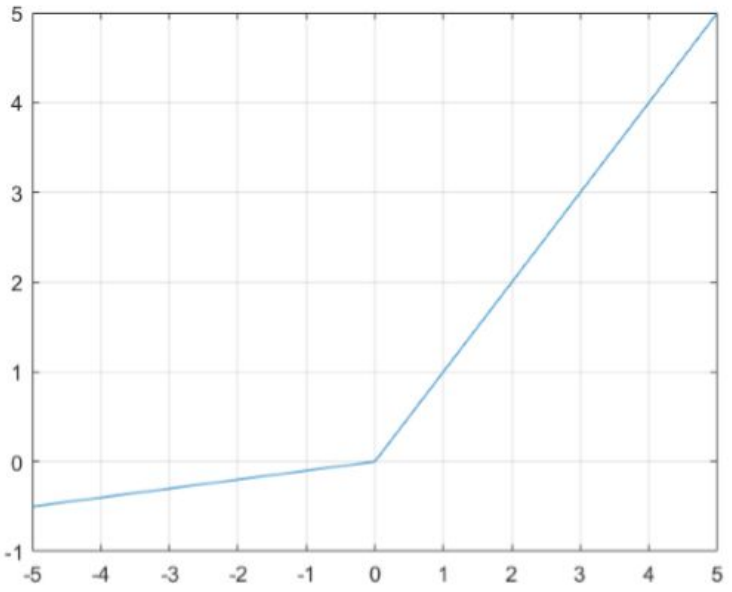


Fig 3.6
ReLU cu pierderi

3.1.3 Algoritmi de optimizare

Rolul pe care il au algoritmii de optimizare este de a modifica modelul in functie de pierdere. Mai simplu, acestia dau forma modelului prin calcularea greutatilor pentru ca modelul sa ia cea mai precisa forma a lui. Cativa din cei mai folositi algoritmi de optimizare sunt: Metoda gradientului, Metoda gradientului stochastic, Adagrad, RMSProp, Adam.

In acest proiect vom folosi algoritmul Adam pentru ca acesta are atat o acuratete mai buna comparativ cu celelalte enumerate mai sus, cat si o rata mai mare de convergenta dupa cum se poate observa in figura urmatoare.

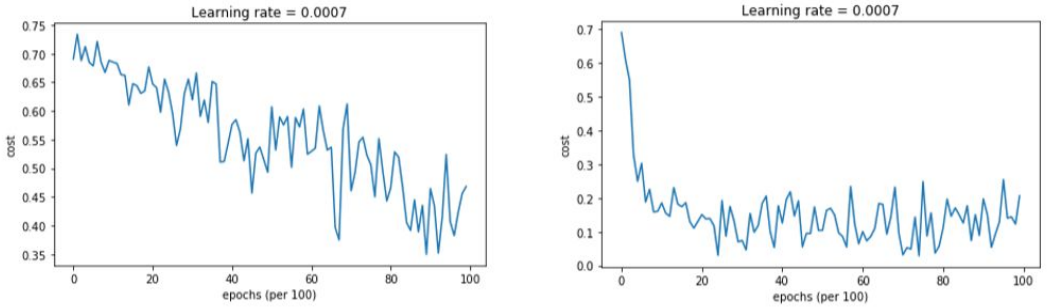


Fig 3.7
Comparatie intre metoda gradient si algoritmul Adam

3.2 Prezentarea implementarii

Primul pas spre implementarea acestui proiect a fost gasirea unui set de date compatibil cu dorintele noastre si anume melodii in format ABC-notation.

Dupa gasirea setului de date [11], datele trebuie pregatite pentru a fi procesate de catre modelul nostru. Datele vor fi date modelului in **batch**-uri de 16 secvente, fiecare secventa avand 64 de caractere. Am atribuit un index numeric fiecarui caracter unic din setul de date.

Urmatorul pas a fost crearea unui model de Deep Learning, cu urmatoarea arhitectura:

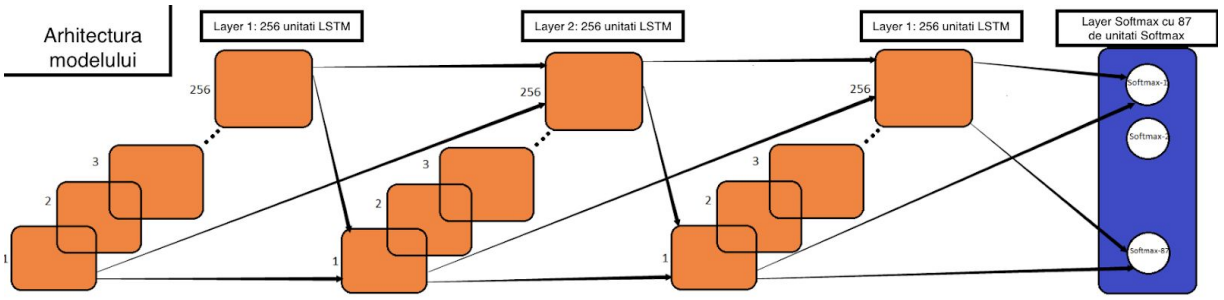


Fig 3.8
Arhitectura modelului utilizat

Reteaua noastra este formata din 3 straturi de tip RNN fiecare fiind format din 256 de unitati LSTM. Iesirea fiecarei unitati LSTM va fi o intrare pentru toate unitatile LSTM din stratul urmator si asa mai departe. Dupa 3 astfel de straturi RNN am creat un strat dens(in care toti neuronii primesc input de la toti neuronii stratului anterior) care se foloseste de functia de activare Softmax pentru a crea probabilitati. In cele din urma, modelul va genera 87 de output-uri, fiecare avand atribuita o probabilitate. Practic, se urmareste ca modelul sa invete din secventele care i se ofera ca in general, dupa o anumita nota, urmeaza o alta nota specifica.

In arhitecturile de tip LSTM se mai intalneste un parametru denumit “stateful”. Daca acest parametru este setat ca si True, atunci se salveaza starea de la un anumit batch-ul i pentru a fi folosita ca stare initiala pentru batch-ul i+1, permitand astfel modelului sa invete secvente mai lungi.

Dupa crearea arhitecturii modelului, acesta a fost antrenat pe datele in format ABC-notation procesate timp de 90 de **epoci** (o epoca reprezinta o trecere prin intreg setul de date), antrenament care a generat o acuratete de 90% a prezicerilor modelului. La fiecare 10 epoci, salvam weight-urile calculate la acel moment pentru a le putea folosi mai departe pentru generarea unei secvente muzicale.

Secventele au fost generate tot in format ABC-notation, pe care l-am convertit mai apoi la formatul MIDI folosind libraria abc2midi[12], creat de James Allwright in 1990.

Am creat de asemenea si un GUI (Graphical User Interface) simplu de utilizat pentru a nu mai oferi parametrii necesari rularii din linia de comanda.

4 Mod de utilizare si interactiunea cu utilizatorul

Aplicatia are o interfata grafica simpla (Fig 4.1), care cere utilizatorului cei trei parametrii necesari rularii. Ea se foloseste de modelul preantrenat pentru generarea unui fisier de tip .mid pe care mai apoi il deschide pentru a vedea rezultatul generat.

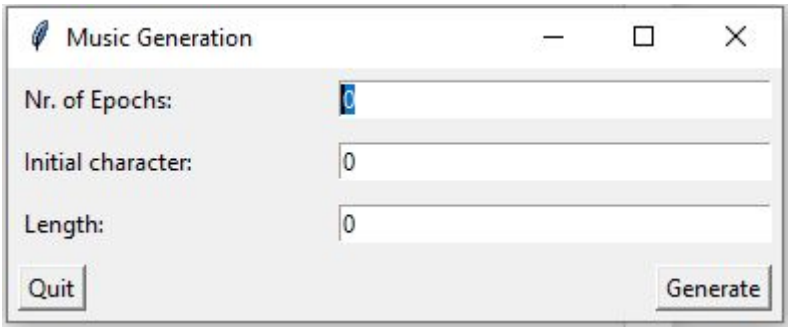


Fig 4.1
Interfata grafica a aplicatiei

Din cauza considerentelor legate de testare si studiul evolutiei prin antrenare, prima obtiune a utilizatorului este de a alege un numar de “epoci” (multiplu de 10) care reprezinta etapa de antrenare a modelulului. Spre exemplu daca utilizatorul alege 50 de epoci, secventa va fi generata de modelul antrenat dupa 50 de epoci.

Dupa cum este prezentat mai sus principiul de functionare al acestui proiect, trebuie sa oferim o valoare de la 0 la 86 care reprezinta un caracter unic pe baza caruia modelul sa construiasca melodia. Aceasta valoare va fii pusa in campul “Initial character”.

Campul “Length” desemneaza lungimea melodiei generate masurata in sunete. Este foarte important de retinut ca lungimea melodiei ca timp poate sa difere in functie de tempoul melodiei. In principiu acest paramentru ar trebuii sa aibe o valoare intre 300 si 600.

Dupa ce parametrii au fost alesi putem genera melodia apasand butonul “Generate”. Pentru a vedea rezultatul aplicatia face un apel de sistem pentru a deschide fisierul generat, care va fi deschis de programul implicit de redare audio pe acel calculator. (Fig 4.2)

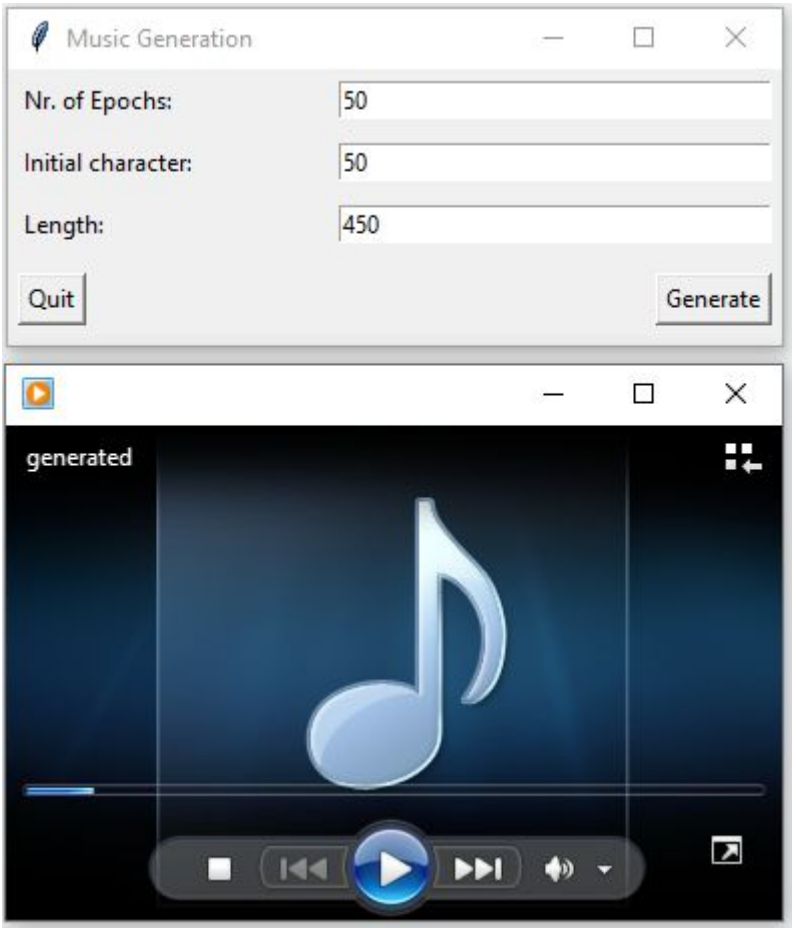
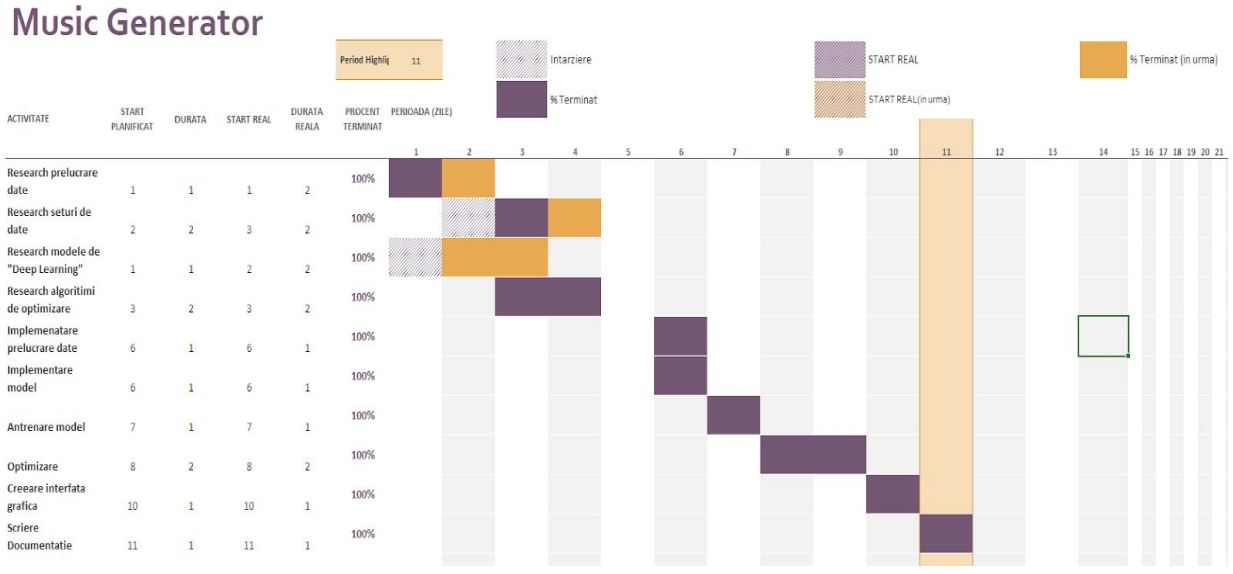


Fig 4.2
Difuzarea melodiei

Pentru a inchide aplicatia se pot folosi fie butonul generic de inchidere a aplicatiei din coltul din dreapta-sus sau buttonul “Quit”.

5 Contributie individuala



Activitate	Responsabil
Research prelucrare de date	Stefan Giba
Research seturi de date	Stefan Giba
Research modele de “Deep Learning”	Rudeanu Adrian
Research algoritmi de optimizare	Rudeanu Adrian
Implementare prelucrare date	Stefan Giba
Implementare model	Rudeanu Adrian
Antrenare model	Rudeanu Adrian
Optimizare	Stefan Giba Rudeanu Adrian
Creeare interfata grafica	Stefan Giba Rudeanu Adrian
Scriere Documentatie	Stefan Giba Rudeanu Adrian

6 Concluzii

Din punct de vedere al obiectivelor propuse, am reusit sa generam muzica folosind retele neuronale, si sa observam intregul proces de evolutie al modelului creat. Rezultatele obtinute sunt notabile intrucat melodiile compuse au muzicalitate.

Avand in vedere cantitatea realativ mica de informatii pe care am oferit-o modelului, putem spune ca am atins o performanta buna a modelului. Acesta a avut o perioada indelungata de antrenare deoarece a folosit CPU si nu GPU, care este mult mai compentent din punctul de vedere al efecturii de calcule.

Utilitatea proiectului, pe langa cea educationala si experimentala, este de oferii puncte de plecare in scrierea de compozitii, intrucat industria muzicala este o industrie care in mod accelerat integreaza progresele tehnologice pentru a produce muzica de o calitate mai ridicata.

Acest proiect ne arata ca mai sunt pasi pe care ii putem parcurge pentru a il imbunatatii. Pentru inceput putem strange mai multe date de antrenare pentru model, cea ce ar oferii o varietate mai mare din care acesta sa invete. O alta posibila imbunatatire ar fi antrenarea cu mai multe instrumente. Ar fi interesant de vizualizat progresul relatiei intre doua instrumente pe parcurul antrenari, desi asta necesita date specializate si greu de obtinut. Iar pentru usura munca utilizatorului am putea adauga functionalitatea de generare a mai multor melodii.

7 Referinte

[1] K. Choi, G. Fazekas, and M. Sandler. Text-based lstm networks for automatic music composition. https://arxiv.org/abs/1604.05358.
[2] S. Hochreiter and J. Schmidhube. Long short-term memory. https://www.bioinf.jku.at/publications/older/2604.pdf.
[3] A. Nayebi and M. Vitelli. Gruv: Algorithmic music generation using recurrent neural networks. https://web.stanford.edu/ anayebi/projects/CS 224D Final Project Writeup.pdf.
[4] A. Y.-T. Ng. Deep learning specialization. Online course. https://www.coursera.org/specializations/deep-learning.
[5] http://colah.github.io/posts/2015-08-Understanding-LSTMs/
[6]https://towardsdatascience.com/how-to-generate-music-using-a-lstm-neural-network-in-keras-68786834d4c5
[7]https://medium.com/@leesurkis/how-to-generate-techno-music-using-deep-learning-17c06910e1b3
[8] https://medium.com/analytics-vidhya/music-generation-using-deep-learning-a2b2848ab17
[9]https://www.analyticsvidhya.com/blog/2020/01/how-to-perform-automatic-music-generation/
[10] https://medium.com/datadriveninvestor/neural-networks-explained-6e21c70d781
[11] http://abc.sourceforge.net/NMD/ - baza de date Nottingham ABC-notation,
[12] https://ifdo.ca/~seymour/runabc/abcguide/abc2midi_guide.html - abc2midi program