UNIVERZITET U SARAJEVU
ELEKTROTEHNIČKI FAKULTET
ODSJEK ZA RAČUNARSTVO I INFORMATIKU

Generisanje muzičkih melodija upotrebom genetičkog algoritma

Projektna dokumentacija

Vještačka inteligencija

Studenti:
Arman Bašović, 19427
Adnan Dervišević, 19433
Azra Pamuk, 19447
Profesor:
V. prof. dr Amila Akagić

juni 2025.

Faza 1: Izbor teme i opis problema

Opis problema

Ovaj projekat se bavi problemom algoritamske kompozicije, specifično generisanjem novih muzičkih melodija koje oponašaju stil postojećeg muzičkog korpusa. Cilj je kreirati sistem vještačke inteligencije koji može "naučiti" stilske karakteristike iz seta primjera (MIDI datoteka) i zatim, koristeći genetički algoritam, evoluirati nove melodije koje su u skladu s naučenim stilom. Rješavanje ovog problema ne samo da predstavlja interesantan tehnički izazov, već otvara i vrata za kreativne primjene u muzici.

Osnovni pojmovi

Genetički algoritam (GA): Optimizacijski algoritam inspirisan prirodnom selekcijom. Koristi operatore kao što su selekcija, ukrštanje i mutacija kako bi evoluirao populaciju rješenja prema cilju definisanom "*fitness*" funkcijom.

MIDI (Musical Instrument Digital Interface): Standardni format za pohranu muzičkih informacija. Ne sadrži audio signal, već simboličke podatke poput visine tona (*pitch*), trajanja, jačine (*velocity*) i instrumenta za svaku notu.

Stilsko modeliranje: Proces ekstrakcije i kvantifikacije karakteristika koje definišu muzički stil. U ovom projektu, stil se modelira kroz statističke distribucije muzičkih elemenata kao što su visine tonova, intervali i trajanja nota.

Fitness funkcija: Ključna komponenta genetičkog algoritma koja ocjenjuje kvalitet svakog rješenja (u ovom slučaju, generisane melodije). Vrijednost "*fitness*a" određuje koliko se melodija poklapa sa željenim stilom.

Korist i primjena rješenja

Rješavanje ovog problema donosi višestruke koristi:

- **Kreativni alati za muzičare:** Sistem može služiti kao izvor inspiracije, nudeći nove melodične ideje unutar stila na kojem muzičar radi.
- **Automatizovano generisanje muzike:** Primjena u generisanju pozadinske muzike za video igre, filmove ili multimedijalne sadržaje, gdje se muzika može dinamički prilagođavati.

- **Istraživanje muzičke teorije:** Analizom i generisanjem muzike, sistem omogućava eksperimentalno istraživanje elemenata koji čine određeni muzički stil prepoznatljivim.
- **Personalizovana muzika:** Mogućnost generisanja muzike prilagođene ukusu korisnika na osnovu njegovih prethodnih izbora.

Pregled postojećih *dataset*-ova

Sistem je dizajniran da bude fleksibilan i da radi sa bilo kojom kolekcijom MIDI datoteka koju korisnik obezbijedi. Postoji veliki broj javno dostupnih setova podataka koji se mogu koristiti:

- ***Lakh MIDI Dataset*[1]:** Jedna od najvećih kolekcija, sa preko 176,000 jedinstvenih MIDI datoteka, od kojih su mnoge povezane sa postojećim pjesmama. Ovaj set je idealan za učenje širokog spektra stilova.
- ***Classical Music Archives*[2]:** Online postoje brojne arhive klasične muzike u MIDI formatu, sortirane po kompozitoru (npr. *Bach*, *Mozart*, *Beethoven*), što omogućava učenje i generisanje melodija u stilu specifičnih historijskih perioda.
- **MIDI Classical Music[3]:** Dostupna na popularnoj platformi *Hugging Face*, ova kolekcija sadrži skoro 5.000 MIDI datoteka klasične muzike, organizovanih po kompozitorima. S obzirom da je *dataset* već pročišćen i strukturiran, predstavlja odličan i lako dostupan resurs za fokusirano učenje stilova specifičnih kompozitora poput *Bacha*, *Beethovena* ili *Chopina*.
- **Music Generation *Dataset*[4]:** Sadrži kolekciju od oko 30 MIDI datoteka prikupljenih iz različitih izvora. Ovaj set podataka je prvenstveno namijenjen kao praktičan primjer za učenje, jer dolazi sa programskim kodom (*Notebook*) koji demonstrira kako se datoteke mogu obrađivati. Zbog malog broja datoteka, idealan je za brzo testiranje i za korisnike koji prave prve korake u generisanju muzike.
- **Žanrovski specifične kolekcije:** Moguće je kreirati vlastite setove podataka preuzimanjem MIDI datoteka specifičnih žanrova poput džeza, roka ili tradicionalne muzike.

[1] https://www.kaggle.com/datasets/imsparsh/lakh-midi-clean
[2] https://www.classicalarchives.com/midi.html
[3] https://huggingface.co/datasets/drengskapur/midi-classical-music
[4] https://www.kaggle.com/datasets/ahemateja19bec1025/musicgenerationdataset

Faza 2: Pregled stanja u oblasti

Algoritamska kompozicija ima dugu historiju, a pristupi zasnovani na vještačkoj inteligenciji postali su dominantni.[5] Imitacija stila je jedan od centralnih zadataka u ovom polju.[6] [7]

Klasični pristupi, poput onih zasnovanih na gramatikama ili Markovljevim lancima, uspješno su korišteni za modeliranje muzike. Genetički algoritmi (GA) su se također pokazali kao moćan alat, posebno zbog svoje sposobnosti istraživanja ogromnog prostora mogućih melodija.[8] Radovi poput Bilesovog "*GenJam*" sistema za generisanje džez improvizacija pokazali su potencijal GA u kreativnim domenama. Prednost GA leži u njihovoj fleksibilnosti i mogućnosti optimizacije ka složenim, ponekad i subjektivnim ciljevima, definisanim kroz fitness funkciju.

Moderni pristupi se u velikoj mjeri oslanjaju na duboko učenje. Projekti poput *Google Magenta* koriste napredne arhitekture kao što su rekurentne neuronske mreže (RNN), LSTMs i Transformeri za generisanje muzike.[9] Ovi modeli su sposobni da nauče dugoročne zavisnosti i složene harmonijske strukture iz ogromnih setova podataka. Modeli poput *MusicVAE* i *WaveNet* pokazuju izvanredne rezultate u generisanju koherentnih i visokokvalitetnih muzičkih djela.[10] [11]

Pristup korišten u projektu: Implementirano rješenje koristi genetički algoritam, što ga svrstava u klasične, ali i dalje relevantne i efektivne metode. Za razliku od "crne kutije" koju predstavljaju modeli dubokog učenja, ovaj pristup je transparentniji. Fitness funkcija, zasnovana na statističkim distribucijama, eksplicitno definira šta čini "dobru" melodiju, što omogućava laku interpretaciju i modifikaciju. Iako možda ne dostiže složenost Transformer modela, ovaj projekat predstavlja robustan i praktičan sistem za stilsku imitaciju koji je računarski manje zahtjevan i lakši za adaptaciju na specifične, manje setove podataka. Potencijalni pravac za poboljšanje bio bi integracija naprednijih modela stila (npr. Markovljevih modela) unutar fitness funkcije.

---

[5] https://www.researchgate.net/publication/260039354_AI_Methods_in_Algorithmic_Composition_A_Comprehensive_Survey

[6] https://www.musicprocessing.net/features/algorithmic-composition/

[7] https://www.janmbuys.com/theses/Buys-HonsReport-GenerativeModelsMusic.pdf

[8] https://people.cs.nott.ac.uk/pszeo/docs/publications/amuse07.pdf

[9] https://magenta.tensorflow.org/

[10] https://magenta.tensorflow.org/music-vae

[11] https://deepmind.google/discover/blog/wavenet-a-generative-model-for-raw-audio/

Faza 3: Izbor, analiza i pretprocesiranje *dataset*-a

Izbor i opis *dataset*-a

Izvor podataka: *Dataset* nije fiksiran. Sistem je dizajniran tako da korisnik postavlja proizvoljan skup MIDI datoteka (.mid, .midi) u folder na *Google Drive*-u. Ovo omogućava maksimalnu fleksibilnost u izboru stila koji se želi naučiti.

Format i preuzimanje: Koristi se standardni MIDI format. Analiza se vrši direktnim učitavanjem ovih datoteka pomoću *music21* biblioteke.

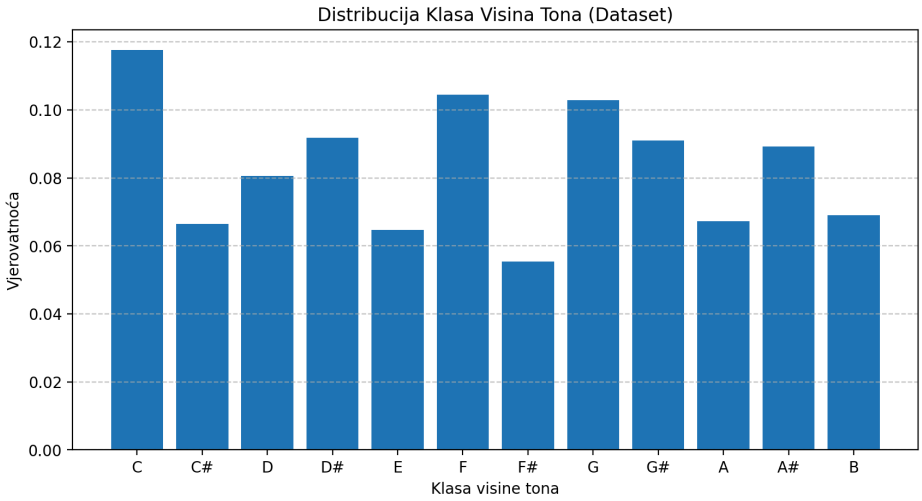
Karakteristike *dataset*-a (primjer): Pošto je *dataset* proizvoljan, konkretni brojevi (broj instanci, atributa, veličina) zavise od korisnikovog izbora. Na primjer, ako korisnik postavi 20 MIDI datoteka barokne muzike, to će biti *dataset*. Veličina tipične MIDI datoteke je u rangu od nekoliko do stotina kilobajta.

Analiza i pretprocesiranje

Proces učenja stila iz *dataset*-a (*learn_style_from_dataset* funkcija) predstavlja ključnu fazu analize i pretprocesiranja.

- **Parsiranje MIDI datoteka:** Svaka MIDI datoteka u navedenom folderu se parsira pomoću *music21* biblioteke. Ova biblioteka pretvara MIDI podatke u struktuiranu listu muzičkih elemenata (note, pauze, akordi).
- **Ekstrakcija karakteristika:** Iz svake datoteke se ekstraktuju note i njihove osnovne karakteristike: visina tona (*pitch*) i trajanje (*duration*). U slučaju akorda, uzima se najviša nota kao reprezentativna.
- **Modeliranje stila:** Da bi se kvantifikovao "stil", iz cjelokupnog *dataset*-a se računaju agregatne statističke distribucije pet ključnih karakteristika. Ove distribucije zajedno čine "stilski model" koji će genetički algoritam koristiti kao cilj. U nastavku detaljnije opisujemo najvažnije karakteristike i prikazujemo njihovu vizuelnu analizu na primjeru prethodno spomenutog *Music Generation Dataset*-a.

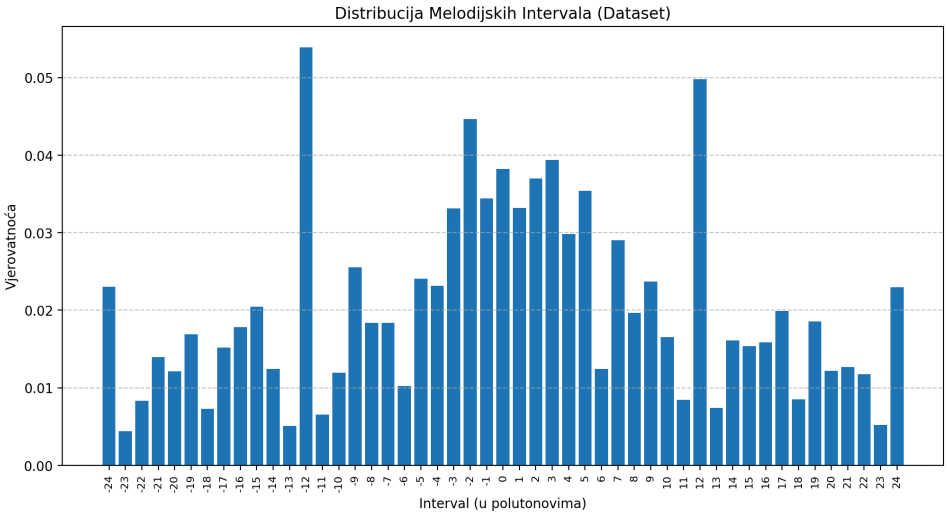
    1. **Distribucija klasa visine tona (*Pitch Class*):** Frekvencija pojavljivanja svake od 12 nota unutar oktave (C, C#, D, ... B). Ovo opisuje tonalni centar muzike.



Slika 3.1: Analiza tonaliteta

Grafik na slici 3.1 pokazuje da je nota C tonalni centar stila ovog *dataset*-a, što je ključna informacija za generisanje harmonično ispravnih melodija.

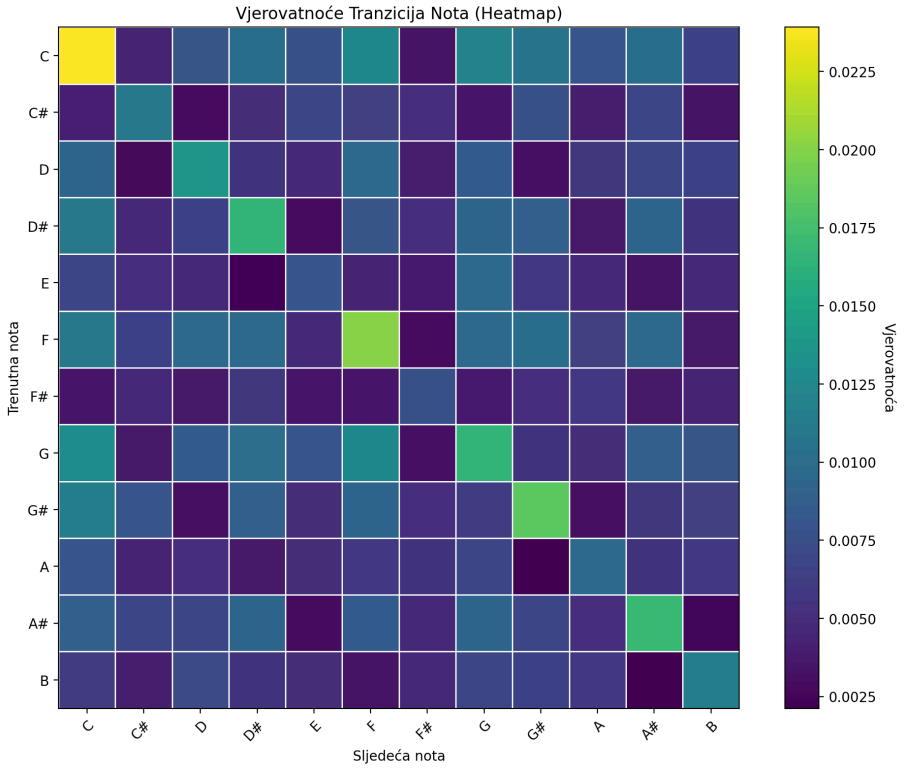
2. **Distribucija intervala:** Frekvencija intervala (razlike u visini tona u polutonovima) između uzastopnih nota. Ovo opisuje melodične skokove.



Slika 3.2: Analiza melodijske konture

Sa slike 3.2 vidimo da stilom ovog *dataset*-a dominiraju dramatični oktavni skokovi (+/-12) i uglađeni, postepeni pomaci, što algoritmu daje jasan obrazac za kretanje melodije.

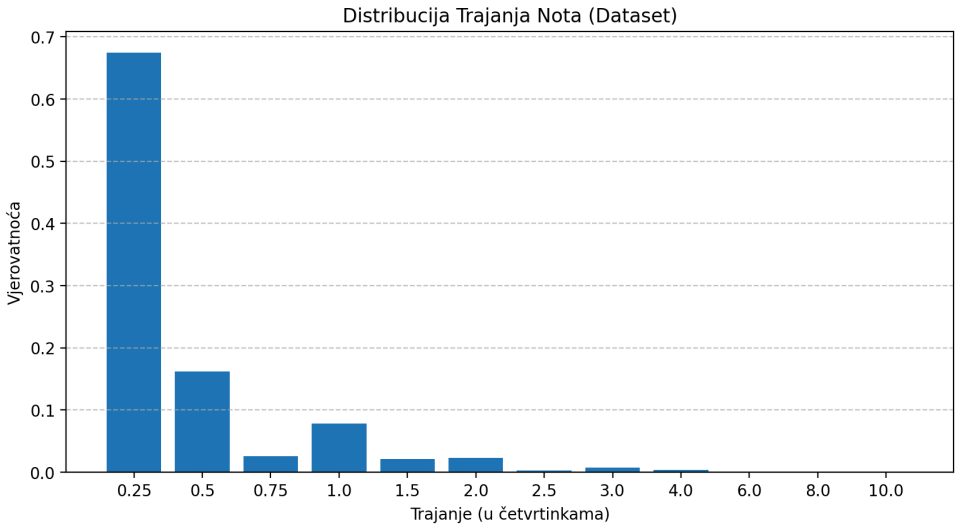
3. **Distribucija bigrama klasa visine tona:** Frekvencija parova uzastopnih klasa visine tona. Ovo modelira uobičajene tranzicije između nota.



Slika 3.3: Analiza vjerovatnoće tranzicija nota

Ova distribucija otkriva "dozvoljene" i "zabranjene" prijelaze između nota. Na primjer, sa grafika na slici 3.3 vidimo da je u ovom *dataset*-u prijelaz C->C čest (svijetao kvadratić), dok je C->C# rijedak (taman kvadratić), pružajući algoritmu logiku za slaganje nota.

4. **Distribucija trajanja nota:** Frekvencija različitih trajanja nota (kvantizovanih na predefinirane vrijednosti poput četvrtinke, polovinke itd.).



Slika 3.4: Analiza ritma

Sa slike 3.4 je jasno da stil ovog *dataset*-a favorizuje izuzetno brze, kratke note (šesnaestinke), što definiše njegov energičan i tehnički karakter.

5. **Distribucija *Inter-Onset* intervala (IOI):** Frekvencija vremenskog razmaka između početaka uzastopnih nota, što opisuje ritmičku strukturu. U trenutnoj implementaciji, ovo je ekvivalentno distribuciji trajanja.

- **Normalizacija:** Sve izbrojane frekvencije se normalizuju kako bi se dobile distribucije vjerovatnoće. Ove distribucije zajedno čine "stilski model" koji će genetički algoritam koristiti kao cilj.

Rizici

Glavni rizik u ovoj fazi je nehomogenost *dataset*-a. Ako korisnik u folder stavi MIDI datoteke iz potpuno različitih stilova (npr. klasična muzika i heavy metal), rezultujući stilski model će biti "zamućen" i bez jasne definicije stila. Generisane melodije će vjerovatno zvučati haotično i bez jasno definisanog stila.

Faza 4: Odabir, formiranje, treniranje i testiranje modela

Odabrana metoda i tehnologije

Metoda: Za generisanje melodija koristi se Genetički Algoritam (GA). Ovaj izbor je motivisan sposobnošću GA da efikasno pretražuje veliki i složen prostor svih mogućih melodija kako bi pronašao one koje optimalno zadovoljavaju definisane stilske kriterije.

Tehnologije:

Python: Glavni programski jezik.

Google Colab: Razvojno i izvršno okruženje, kako je i preporučeno.

music21 i *pretty_midi*: Biblioteke za analizu i sintezu MIDI datoteka.

numpy: Za numeričke operacije i rad sa distribucijama.

ipywidgets: Za kreiranje interaktivnog korisničkog interfejsa unutar Colab okruženja.

fluidsynth: Alat za konverziju generisanih MIDI datoteka u audio datoteke (WAV format).

Formiranje i treniranje modela

"Model" u ovom kontekstu je cjelokupni sistem genetskog algoritma vođen stilskim evaluatorom.

Jedinka (Genom): Jedna melodija, predstavljena kao lista *Python* rječnika. Svaki rječnik definiše jednu notu sa atributima: *pitch* (visina tona), *duration* (trajanje) i *velocity* (jačina).

Populacija: Skup melodija (jedinki) koje se evoluiraju.

Fitness Funkcija (*calculate_fitness*): Srž sistema. Za svaku generisanu melodiju (jedinku), ova funkcija:

Računa iste one statističke karakteristike (distribucije visina, intervala, itd.) kao i u Fazi 3.

Poredi distribucije generisane melodije sa distribucijama ciljnog stila (naučenog iz *dataset*-a). Poređenje se vrši pomoću *Bhattacharyya* koeficijenta, koji mjeri sličnost između dvije distribucije vjerovatnoće.

Ukupna "dobrota" (*fitness*) je prilagođena suma sličnosti za svaku od karakteristika. Melodije koje su statistički sličnije ciljnom stilu dobijaju veći fitness skor.

GA Operatori:

Selekcija: Koristi se turnirska selekcija, gdje se nasumično bira nekoliko jedinki, a ona sa najboljim fitnessom se bira za roditelja.

Ukrštanje (*Crossover*): Primjenjuje se jednopozicijsko ukrštanje, gdje dva roditelja razmjenjuju svoje genetske materijale (dijelove melodije) na jednoj tački prekida.

Mutacija: Sa malom vjerovatnoćom, nasumično se mijenja visina tona ili trajanje pojedinačne note u melodiji.

"Treniranje": Proces "treniranja" je sama egzekucija genetičkog algoritma. Počevši od nasumične populacije melodija, algoritam iterativno primjenjuje selekciju, ukrštanje i mutaciju kroz definisan broj generacija. Kroz ovaj evolutivni proces, prosječni fitness populacije raste, a melodije postaju sve sličnije ciljnom stilu. Najbolja jedinka pronađena tokom cijelog procesa se čuva (elitizam).

Testiranje i metrike

Metrike: Primarna objektivna metrika je *fitness* ocjena, koji kvantifikuje stepen podudaranja generisane melodije sa naučenim stilom. Vrijednosti fitnessa kroz generacije pokazuju uspješnost procesa optimizacije.

Testiranje: Konačni test je subjektivan i sastoji se od slušanja generisanog audio fajla (.wav). Ljudska procjena je krajnji sud o estetskom kvalitetu i koherentnosti melodije.

Testiranje na nepoznatim podacima: Sistem je inherentno dizajniran za rad sa "nepoznatim podacima". Korisnik može:

Učitati potpuno novi set MIDI datoteka kako bi naučio novi stil i generisao muziku u tom stilu.

Kao test, može se kreirati veoma jednostavna MIDI datoteka (npr. C-dur ljestvica), naučiti njen "stil", i posmatrati da li će GA generisati varijacije na tu temu, što bi potvrdilo da sistem ispravno uči i primjenjuje stilske obrasce.

Faza 5: Cjelokupni osvrt na problem i dobijeno rješenje

Postignuti rezultati

Implementirani sistem uspješno generiše muzičke melodije koje oponašaju stil definisan korisničkim setom MIDI datoteka. Korištenjem genetičkog algoritma vođenog statističkom analizom, rješenje je u stanju da uhvati i reproducira ključne melodičke i ritmičke karakteristike, kao što su tonalitet, tipični intervalni skokovi i ritmički obrasci. Interaktivni interfejs u *Google Colabu* omogućava korisnicima laku kontrolu nad parametrima GA i izborom instrumenta, čineći sistem praktičnim alatom za eksperimentisanje.

Poređenje sa radovima iz prethodne faze

U poređenju sa najnaprednijim sistemima poput onih iz *Google Magenta* projekta, koji koriste duboke neuronske mreže, ovaj pristup je jednostavniji. Nedostaje mu sposobnost modeliranja složenih, dugoročnih struktura i harmonije koje Transformeri mogu naučiti.[12] Međutim, prednost ovog rješenja je u njegovoj jasnoći i efikasnosti. Tačno znamo koje karakteristike (pitch, intervali, trajanja) fitness funkcija optimizira, za razliku od "crne kutije" dubokih modela. Sistem je računarski znatno manje zahtjevan i ne zahtijeva dane treniranja na GPU klasterima. Za zadatak imitacije monofone melodične linije, statistički pristup se pokazao kao veoma efikasan.

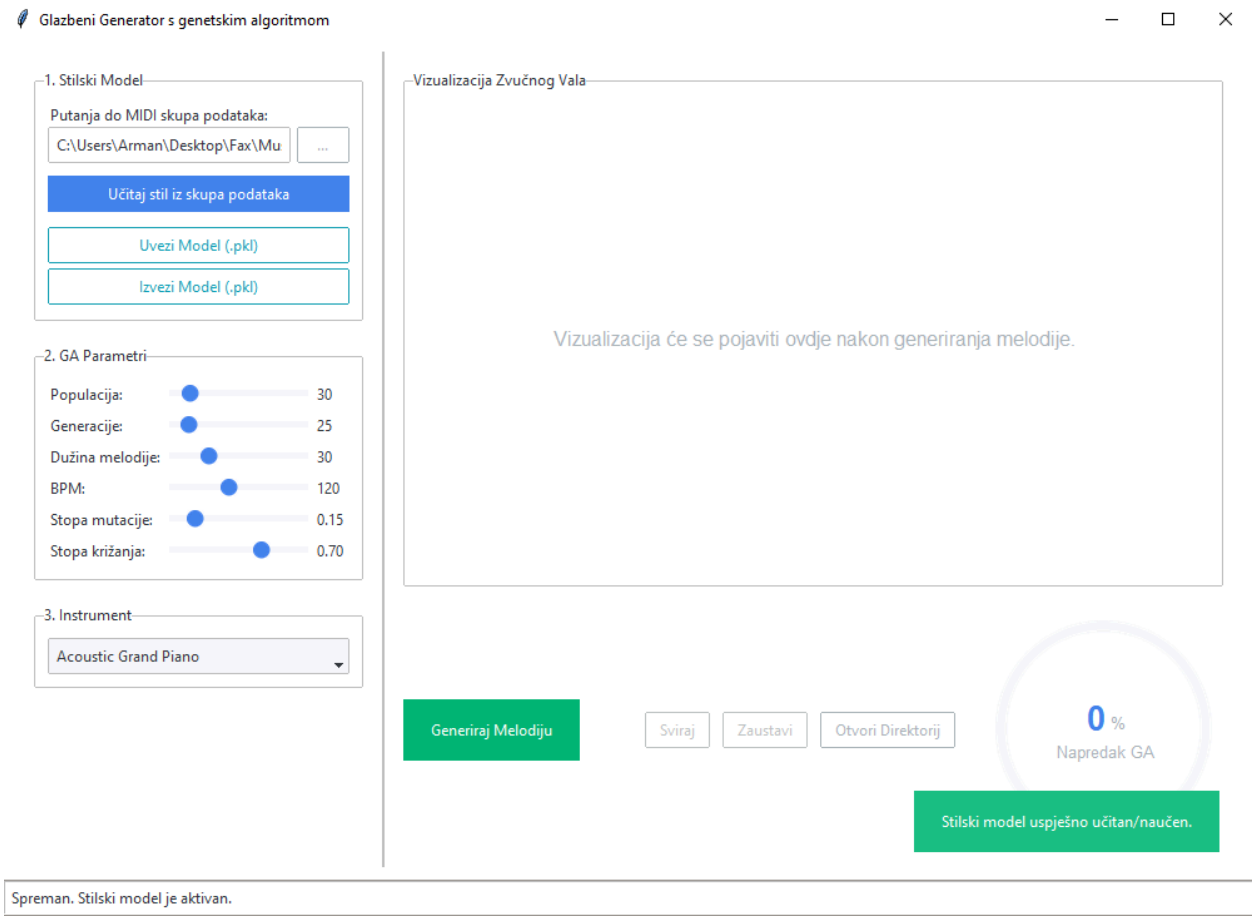
Dostupnost Rješenja: Interaktivna *Desktop* Aplikacija

Pored interaktivnog okruženja realizovanog unutar *Google Colab notebook*-a, razvijena je i samostalna *desktop* aplikacija korištenjem *Python*-a i odgovarajućih biblioteka za grafički interfejs. Aplikacija je javno dostupna i njen kod se može pronaći na *GitHub* repozitoriju: https://github.com/azrapamuk/Music-Generator-App

Razvoj *desktop* aplikacije nudi nekoliko ključnih prednosti u odnosu na izvršavanje koda u online okruženju poput *Colab*-a:

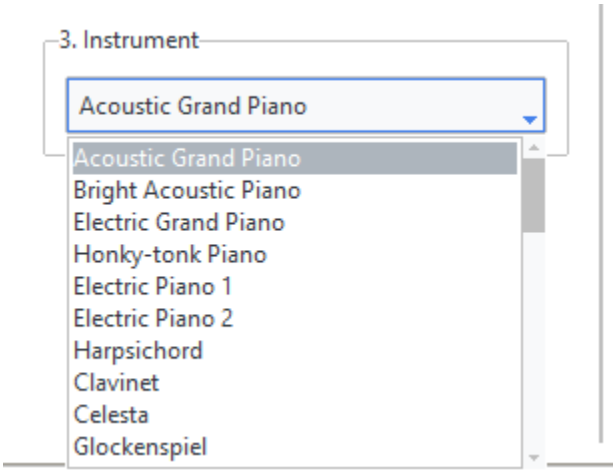
- **Pristupačnost i jednostavnost korištenja:** Aplikacija je namijenjena korisnicima koji nisu nužno programeri. Intuitivni grafički interfejs (GUI) omogućava učitavanje MIDI *dataset*-a, podešavanje parametara genetičkog algoritma i generisanje muzike bez potrebe za interakcijom sa kodom. Početni prozor aplikacije (Slika 4.1) pruža jasan pregled svih opcija.

[12] https://magenta.tensorflow.org/music-transformer



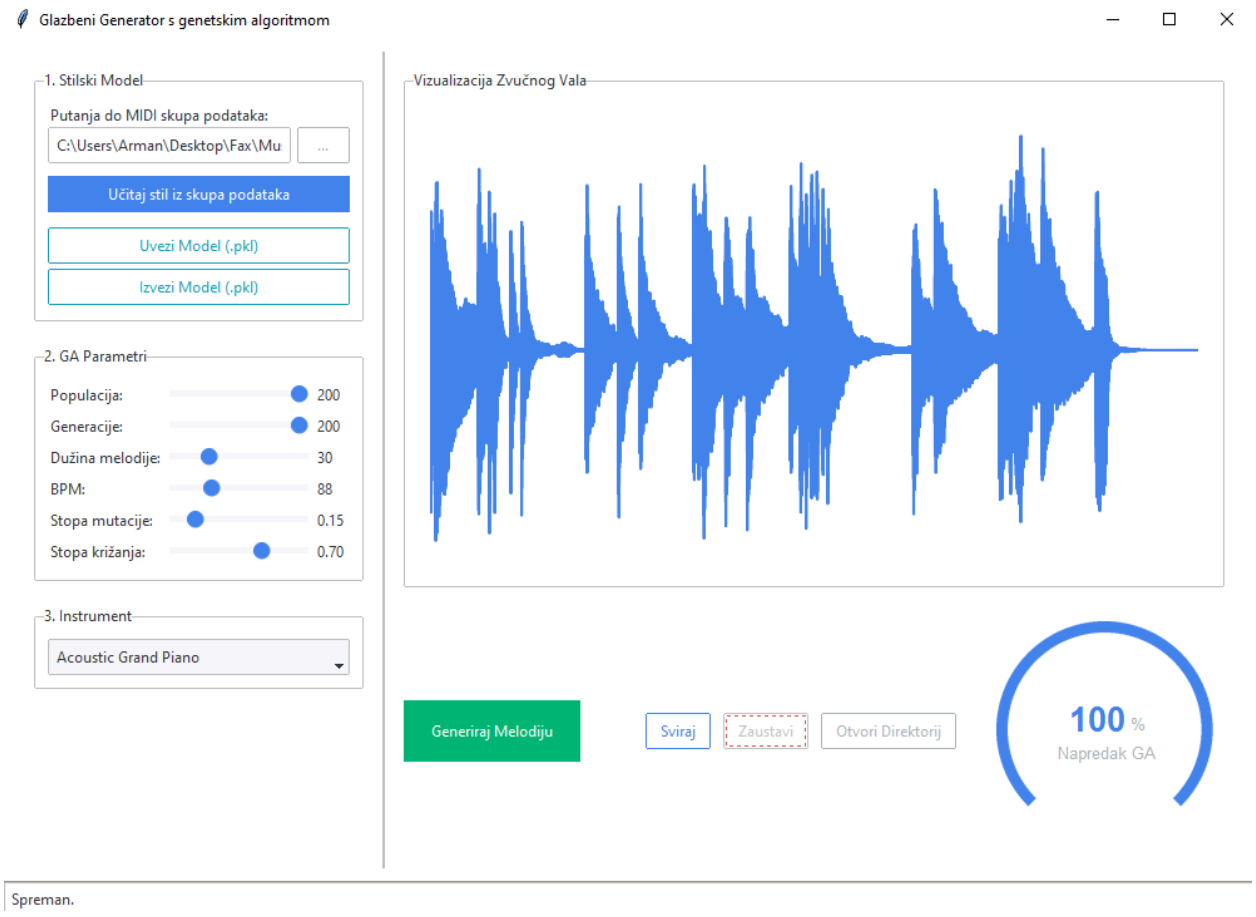
Slika 5.1: Početni korisnički interfejs aplikacije.

Korisnik može lako učitati MIDI dataset i podesiti ključne parametre genetičkog algoritma, kao što su veličina populacije, broj generacija, te stope mutacije i ukrštanja. Također, omogućen je i jednostavan izbor željenog instrumenta za generisanu melodiju (Slika 4.2).

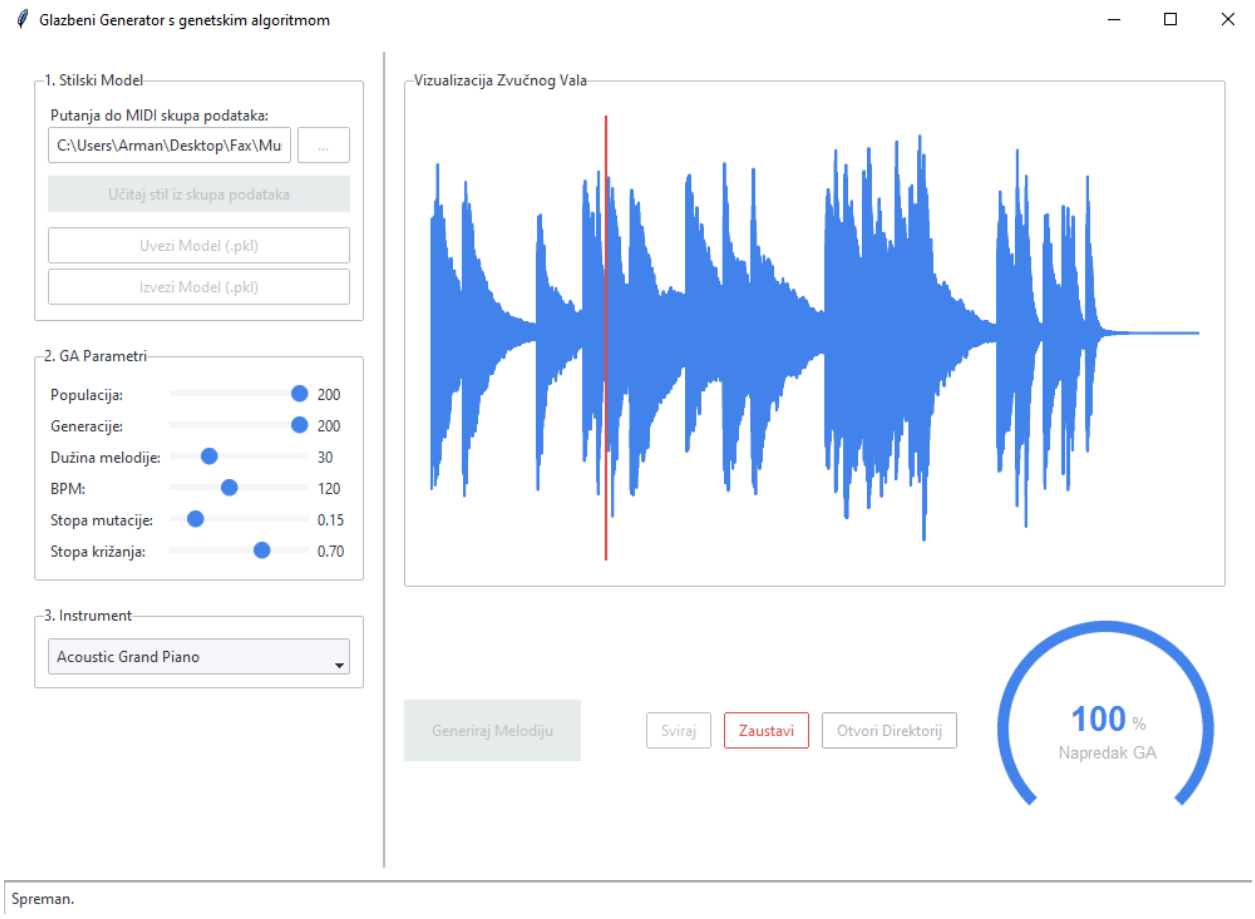


Slika 5.2: Prozor za odabir instrumenta.

Nakon što se proces generisanja završi, aplikacija nudi jasan vizualni prikaz kreirane melodije. Koristi se grafikon na kojem vertikalna osa predstavlja visinu tona, a horizontalna osa vrijeme, čime se efektivno iscrtava melodijska kontura (Slika 4.3). Najvažnije, generisana muzika se može odmah poslušati direktno iz aplikacije, što omogućava trenutnu zvučnu provjeru rezultata (Slika 4.4).



Slika 5.3: Vizualni prikaz generisane melodije.



Slika 5.4: Kontrole za reprodukciju generisane muzike.

- ***Offline* rad:** Jednom instalirana, aplikacija ne zahtijeva stalnu internet konekciju. Korisnici mogu generisati muziku bilo gdje i bilo kada, što je značajna prednost u odnosu na Colab koji ovisi o stabilnoj vezi.

- **Nezavisnost od eksternih servisa:** Korisnici nisu podložni ograničenjima *Colab*-a, kao što su vremensko ograničenje sesije (session timeout) ili promjene u dostupnosti resursa. Aplikacija koristi lokalne resurse računara, pružajući konzistentno i pouzdano iskustvo.

- **Direktna integracija sa lokalnim sistemom:** Učitavanje setova podataka i čuvanje generisanih MIDI ili WAV datoteka je znatno jednostavnije i brže jer se odvija direktno na korisnikovom fajl sistemu, bez potrebe za "mountovanjem" drajvova ili preuzimanjem fajlova iz cloud okruženja.

Šta se moglo bolje uraditi?

Iako je projekat uspješan, postoje brojni pravci za poboljšanje:

Naprednija *Fitness* Funkcija: Fitness funkcija bi se mogla obogatiti sofisticiranijim muzičkim metrikama, kao što su analiza harmoničke napetosti, melodičke konture ili ritmičke složenosti. Težinski parametri u funkciji su trenutno fiksni, a mogli bi biti dinamički ili podesivi od strane korisnika.

Muzički svjesni operatori: Umjesto čisto nasumične mutacije, mogli bi se implementirati "muzički svjesni" operatori. Na primjer, mutacija bi mogla favorizirati promjenu note na ton koji pripada istoj ljestvici ili akordu.

Generisanje harmonije i polifonije: Trenutno rješenje generiše samo jednu melodičnu liniju. Veliki korak naprijed bio bi proširenje sistema za generisanje akorda (harmonije) ili više paralelnih melodičnih linija (polifonija).

Strukturno generisanje: Sistem generiše melodiju fiksne dužine. Mogao bi se unaprijediti da generiše muziku sa strukturom, kao što su A-B-A forma (strofa-refren-strofa), koristeći hijerarhijski genetički algoritam.

Hibridni pristup: Kombinovanje genetičkog algoritma sa jednostavnijim modelima mašinskog učenja. Na primjer, stilski model bi umjesto prostih distribucija mogao koristiti Markovljev lanac ili malu rekurentnu neuronsku mrežu (RNN) za modeliranje tranzicija između nota, što bi *fitness* funkciju učinilo još moćnijom.