Pronalaženje arhitekture rekurentne neuronske mreže neosetljive na težine upotrebom genetskog algoritma

Ognjen Nešković

Mentori: Aleksa Tešić, Vladimir Lunić

U radu je predložen algoritam za pronalaženje arhitektura rekurentnih neuronskih mreža koje su neosetljive na izbore “težina” i “biasa” (tj. parametara neuronskih mreža). Algoritam za nalaženje ovih arhitektura zasnovan je na genetskom algoritmu, na algoritmu NEAT i radu “Weight agnostic neural networks” (Gaier i Ha 2019). Za potrebe evaluacije korišćena su okruženja Open AI gym. Na ovim okruženjima prethodno su testirani mnogi algoritmi koji se smatraju najboljim poznatim algoritmima u oblasti učenja sa podsticajem. Za neka od ovih okruženja predloženi algoritam postiže rezultate koji su u rangu najboljih poznatih rezultata za ta okruženja. Po broju neurona u skrivenim slojevima, neke mreže dobijene ovim algoritmom su bolje (imaju manji broj neurona) nego mreže dobijene u radu Gaiera i Ha. Osetljivost mreža na promene u težinama i biasima je veća nego u radu Gaiera i Ha.

Ognjen Nešković (2003), Beograd, učenik 3. razreda Računarske gimnazije u Beogradu
MENTORI: Aleksa Tešić, Vladimir Lunić

1. Uvod

1.1. Osnove

Neuronske mreže su najrasprostranjeniji univerzalni aproksimatori funkcija trenutno u upotrebi. Aproksimacija funkcija koristi se za rešavanje širokog spektra problema: klasifikacija i segmentacija (slika, teksta, itd.), prevođenje i generisanje teksta, igranje igara. Stoga igraju značajnu ulogu u mnogim oblastima ljudske delatnosti pored računarstva (fizika, ekonomija, hemija, biohemija, itd.)

Neuronske mreže inspirisane su biološkim neuronima i mozgovima životinja u prirodi. Iako su zasnovane na kompleksnim biološkim sistemima poput mozgova, neuronske mreže u računarstvu predstavljaju znatno uprošćen model sa dve suštinske karakteristike: arhitekturom (položaj i povezanost “veštačkih neurona”) i težinama (parametri “veštačkih sinapsi” koji označavaju koliko jako se signal pojačava ili inhibira). Proces pronalaženja optimalne arhitekture i težina naziva se treniranje neuronske mreže.

Postoje mnoge varijacije neuronskih mreža, dve značajne varijacije su "feed-forward" mreže i rekurentne mreže. U slučaju feed-forward mreža uzima se da impuls putuje od ulaza u mrežu (analogno receptorima u telu) do izlaza iz mreže tako da nikada ne prođe više puta kroz isti neuron. Rekurentne mreže suštinski dozvoljavaju da impuls više puta prođe kroz iste neurone, čime se neuronskoj mreži daje "sposobnost pamćenja", pa su tako našle primenu u problemima koji zahtevaju razumevanje teksta ili videa.

Sa obzirom na to da su nastale sredinom 20. veka, a dovedene u široku upotrebu još početkom 21. veka jasno je da je uložen velik trud u optimizaciju algoritama za treniranje neuronskih mreža. Praksa koja se pokazala uspešnom pri treniranju neuronskih mreža je da se postavi fiksna arhitektura ili da se eventualno isproba nekoliko različitih, a da se najveći procenat vremena uloži u pronalaženje optimalnih težina.

Međutim postoji i veliki broj varijacija algoritama koji optimizuju arhitekturu feed-forward neuronskih mreža: traženje arhitekture u diferencijabilnom prostoru (DARTS), pretraga zasnovana na encoder-decoder sistemu (NAO), učenje sa podsticajem ili genetskim algoritmom.

Jedan od načina pretrage je uklanjanje nepotrebnih veza iz potpuno povezanih slojeva mreže dok se ne dobije minimalna arhitektura (Han et al. 2015; Zhou et al. 2019), iako daju dobre rezultate ove tehnike zahtevaju treniranje mreže i ograničene su na početnu arhitekturu mreže (ne mogu dodavati delove već samo uklanjati).

Drugi način za pronalaženje arhitekture je dodavanje novih delova i građenje sve kompleksnijih mreža (Li i Talwalkar 2020; Yu et al. 2019), da bi se postigli rezultati koje ostvaruju konvencionalne arhitekture poput konvolucionih neuronskih mreža u razumnom vremenu u mnogim radovima koriste se delovi mreža poput LSTM (Long short-term memory) "ćelija" ili konvolucionih "ćelija" za koje se već pouzdano zna da daju dobre rezultate.

1.2. Motivacija

Pokazano je da se arhitekture feed-forward mreža ipak mogu naći bez korišćenja postojećih delova ili treniranja težina u mrežama (Gaier i Ha 2019) i da takve arhitekture postižu rezultate slične trenutno najboljim algoritmima.

Kao što je napomenuto, feed-forward mreže ograničene su pretpostavkom da ne postoje rekurentne veze među neuronima (da impuls putuje direktno od ulaza ka izlazu). Poznatno je da rekurentne veze igraju značajne uloge u mozgovima životinja (Bullier et al. 2001), a i rekurentne arhitekture već imaju značajnu ulogu u mašinskom učenju. Pa je stoga opravdano očekivati da bi se mogle dobiti arhitekture koje rešavaju širi spektar problema nego feed-forward arhitekture ili su efikasnije nego arhitekture koje su ručno pravljene.

1.3. Cilj rada

Glavni cilj ovog rada je generalizacija i poboljšanje algoritma korišćenog za pronalaženje arhitektura feed-forward mreža koji su dali Gaier i Ha. Algoritam je potrebno proširiti tako da se traže i rekurentne neuronske mreže. Očekuje se da će takve mreže po nekoj od metrika (složenost mreže ili postignutom rezultatu) biti bolje od feed-forward mreža za iste probleme. Dodatno očekuje se da će mreže biti neosetljive na izbor težina, pošto algoritam ne uključuje optimizovanje težina u neuronskim mrežama.

2. Metod

2.1. Metrike

Optimizovane su dve metrke: složenost mreže i postignut rezultat pomoću algoritma NSGA-II (Deb et al. 2002).

1. Za metriku **složenosti mreže** uzet je broj neurona u mreži.
2. Svaka mreža je evaluirana na nekoliko različitih deljenih težina i za svaku od težina mreža je nekoliko puta testirana u nasumično inicijalizovanom okruženju, pa su stoga sračunate dve metrike za postignut rezultat:
    a. **Prosečan rezultat** je metrika koja se dobija tako što se prvo sračunaju proseci postignutog rezultata za svaku odabranu težinu (pošto se za svaku težinu testira nekoliko puta), a zatim se nađe prosek tih proseka.
    b. **Maksimalan rezultat** je metrika koja se dobija tako što se prvo sračunaju proseci postignutog rezultata za svaku odabranu težinu, a zatim se nađe maksimum tih proseka.

2.2. Struktura neuronskih mreža

Neuroni u mreži su organizovani u slojeve. Inicijalno postoje ulazni sloj L_0i izlazni sloj L_n. Dodavanje novih neurona između postojećih slojeva stvara nove slojeve mreže.

Sloj L_0sačinjen je od dve specijalne vrste neurona: ulaznih neurona i ulaznih neurona (N_{in}) skrivenog stanja (H_{in}). Izlazni sloj $L_n$je takođe sačinjen od dve vrste neurona: izlaznih neurona (N_{out}) i izlaznih neurona skrivenog stanja (H_{out}). Broj neurona u ulaznom i izlaznom sloju je fiksan i aktivacione funkcije svih neurona u ulaznom i izlaznom sloju su $f(x) = x$ i ne mogu se menjati. Broj neurona u H_{in}je isti kao broj neurona u H_{out}.

Sve veze između neurona sadržane su u kvadratnoj, binarnoj matrici W. Dimenzije matrice Wsu $(N \times N)$gde je Nukupan broj neurona u mreži. Ukoliko za neka dva indeksa i i $j$važi da $W_{j,i} = 0$onda kažemo da neuroni i i $j$nisu povezani, a ako važi da $W_{j,i} = 1$onda kažemo da su neuroni i i $j$povezani.

Neuronska mreža interaguje sa okruženjem u diskretnim koracima t_i, pri svakom koraku okruženje mora proizvesti niz podataka koji predstavljaju stanje u vremenskom trenutku t_i. Veličina niza jednaka je broju ulaznih neurona, tj. broju neurona u N_{in}. U zavisnosti od okruženja neuronska mreža proizvodi "akciju" koja može biti najčešće jedan broj, ili nekoliko brojeva, koji predstavljaju šta mreža "želi da uradi" u okruženju u vremenskom koraku t_i. Zbog prirode problema koji se rešava moguće je da je broj neurona u N_{out} različit od veličine niza koji okruženje očekuje kao akciju, pa se na aktivacije neurona u N_{out} primenjuje funkcija $Act\ (N_{out})$i ta vrednost se prosleđuje okruženju.

Pa se stoga celokupna interakcija mreže sa okruženjem može opisati na sledeći način:

1. Aktivacije svih neurona u skupu H_{in} postavljaju se na vrednost 0.
2. Dobavlja se trenutno stanje od okruženja (niz S, $|S| = \left|N_{in}\right|$) aktivacije neurona u skupu N_{in} postavljaju se na vrednosti iz niza S.
3. Računa se vrednost neurona u izlaznom sloju L_n.
4. Okruženju se prosleđuje vrednost $Act\ (N_{out})$.
5. Aktivacije neurona u H_{in} postavljaju se na aktivacije neurona u H_{out}.
6. Postupak se ponavlja od koraka 2 dok se interakcija ne završi.

2.3. Evaluacija neuronskih mreža

Ako neuronska mreža sadrži slojeve $L_0, L_1, \dots, L_n$, veze između neurona su definisane matricom W, aktivacione funkcije svakog od neurona date su u nizu funkcija f i data je "deljena težina" (konstanta) b. Takođe date su aktivacije ulaznog sloja L_0. Aktivacije u izlaznom sloju L_n se nalaze na sledeći način.

Uvedimo matricu Aveličine $(N \times 1)$gde je Nukupan broj neurona. Za svaki neuron n u sloju L_0postavimo vrednost A_n na aktivaciju neurona n. Zatim prolazimo kroz slojeve redom, počevši od L_1. Za svaki neuron n u sloju $L_i$uzmimo "podmatricu" W_n veličine $(1 \times N)$ i pomnožimo matrice W_n i A . Ovako dobijen je jedan broj S_n. Konačno aktivacija neurona $n$dobija se kao$A_n = f_n(S_n + b)$. Ponavljanjem ovog procesa za svaki neuron u svakom sloju redom od L_1 do L_n dobijaju se aktivcije svih neurona. Dakle aktivacije poslednjeg sloja L_n mogu se samo pročitati iz A.

2.4. Mutacija i ukrštanje

Definisana su četiri operatora mutacije:

1. Dodavanje novog neurona:

   Nasumično se bira jedna od postojećih konekcija u mreži između neurona ii neurona j. Uklanja se veza (i, j), dodaje se novi neuron $k$sa aktivacionom funkcijom $f(x) = x$, dodaju se veze (i, k) i (k, j).

2. Dodavanje nove veze:

   Biraju se dva sloja neuronske mreže (L_i, L_j), $i < j$. Nasumično se biraju dva neurona $n_0 \in L_i$i $n_1 \in L_j$. Dodaje se veza (n_0, n_1).

3. Uklanjanje postojeće veze:

   Nasumično se bira postojeća veza i ona se uklanja.

4. Promena aktivacione funkcije neurona:

   Nasumično se bira jedan od neurona koji ne pripadaju ulaznom i izlaznom sloju mreže. Za odabrani neuron nasumično se bira nova aktivaciona funkcija.

Operator ukrštanja organizama se suštinski ne koristi. Pri "ukrštanju" dva organizma, uvek se uzima bolji.

2.5. Evoluiranje topologija rekurentnih neuronskih mreža

2.5.1. Algoritam

1. Inicijalizuje se početna populacija veličine H_0:

   Pošto zavise od problema, pretpostavlja se da su veličine slojeva L_0 i $L_n$poznate i da svi organizmi u populaciji P_0sadrže neurone u slojevima L_0i L_n.

   Za svaki organizam u P_0 sledeća procedura se ponavlja $\lfloor H_1 \cdot L_0 \cdot L_n \rfloor$puta:

   Sa verovatnoćom H_2primenjuje se operator mutacije broj 2.

   Za svaki organizam u P_0primenjuje se operator mutacije broj 1.

   Primenjuju se koraci 6,7 i 8 kako bi se sortirala populacija P_0.

2. Inicijalizuje se prazna populacija (P_{t+1}) veličine H_0.
3. Briše se poslednjih $H_0 \cdot H_3$članova populacije P_t.
4. Prvih $\lfloor H_0 \cdot H_4 \rfloor$članova populacije P_t prepisuje se u populaciju P_{t+1}.

5. Kroz selekciju i mutaciju nastaje novih $H_0 - \lfloor H_0 \cdot H_4 \rfloor$jedinki i one se dodaju kao ostatak populacije P_{t+1}:

   Pravi se matrica nasumičnih prirodnih brojeva u opsegu $[0, H_0)$ T_1veličine $((H_0 - \lfloor H_0 \cdot H_4 \rfloor) \times H_5)$. Pravi se matrica nasumičnih prirodnih brojeva T_2 u istom opsegu i iste veličine kao T_1. Uzima se minimum po redovima u matrici T_1i matrici T_2. Ovako se dobijaju matrice T_1'i T_2'veličina $((H_0 - \lfloor H_0 \cdot H_4 \rfloor) \times 1)$. Za svaki red i matrica T_1'i T_2' pravi se novi organizam ukrštanjem organizama koji su na pozicijama $T_{1\,i}'$ i $T_{2\,i}'$. Sa verovatnoćama H_6, H_7, H_8, H_9bira se jedan od četiri operatora mutacije i odgovarajuća mutacija se primenjuje na novi organizam. Novi organizam se dodaje u populaciju P_{t+1}.

6. Celokupna populacija P_{t+1}se evaluira tako da se dobiju metrike složenost mreže, prosečan rezultat i maksimalan rezultat.
7. Sa verovatnoćom H_{10}biraju se metrike prosečan rezultat i maksimalan rezultat, a sa verovatnoćom $1 - H_{10}$biraju se metrike prosečan rezultat i složenost mreže. Ovako odabrane dve metrike se dodeljuju svakom organizmu u populaciji P_{t+1}.
8. Pomoću algoritma NSGA-II populacija P_{t+1}se sortira u rastućem redosledu (na osnovu dve odabrane metrike).
9. Algoritam se ponavlja od koraka 2.

2.5.2. Hiperparametari

Hiperparametri koji se mogu varirati pri treniranju algoritma su dati u tabeli 1.

Parametar	**Skup vrednosti**
Veličina populacije (H_0)	$\mathbb{N}$
Početni deo veza(H_1)	$[0, 1]$ $(\mathbb{R})$
Verovatnoća nastajanja početnih veza (H_2)	$[0, 1]$ $(\mathbb{R})$
Uklonjeni deo populacije (H_3)	$[0, 1]$ $(\mathbb{R})$
Zadržani deo populacije (H_4)	$[0, 1]$ $(\mathbb{R})$

Veličina "turnira" za selekciju (H_5)	$[2, H_0]$ ($\mathbb{N}$)
Verovatnoća da se izabere operator mutacije broj 1 (H_6)	$[0, 1]$ ($\mathbb{R}$)[1]
Verovatnoća da se izabere operator mutacije broj 2 (H_7)	$[0, 1]$ ($\mathbb{R}$)[1]
Verovatnoća da se izabere operator mutacije broj 3 (H_8)	$[0, 1]$ ($\mathbb{R}$)[1]
Verovatnoća da se izabere operator mutacije broj 4 (H_9)	$[0, 1]$ ($\mathbb{R}$)[1]
Verovatnoća da se koriste metrike prosečna i maksimalna nagrada (H_{10})	$[0, 1]$ ($\mathbb{R}$)

Tabela 1. Hiperparametri i njihove moguće vrednosti

3. Rezultati i diskusija

3.1. Evaluacija

Okruženja u kojima su testirane mreže se na početku inicijalizuju na neko nasumično stanje, pa je onda potrebno nekoliko puta ponoviti evaluaciju kako bi se aproksimirao pravi postignuti rezultat.

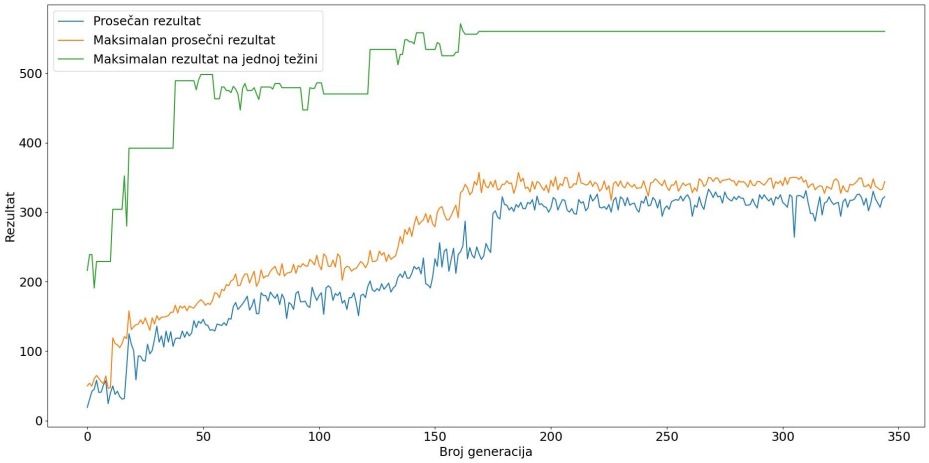
Svaka mreža testirana je na nekoliko različitih deljenih težina, konkretno korišćeno je šest težina: -2, -1, -0.5, 0.5, 1 i 2. Za svaku od tih težina evaluacija je ponovljena tri puta.

3.2. Evoluiranje topologija rekurentnih neuronskih mreža

3.2.1. Rezultati

Pregled rezultata dat je u tabeli 2. Vizulaizacija procesa treniranja prikazana je na slici 1. "Prosečan rezultat" je prosečna vrednost metrike "prosečan rezultat" (2.1, 2a) u celoj populaciji. "Maksimalan prosečan rezultat" je maksimalna vrednost metrike "prosečan rezultat" u celoj populaciji. "Maksimalan rezultat na jednoj težini" je maksimalna vrednost metrike "maksimalan rezultat" (2.1 2b) u celoj populaciji.

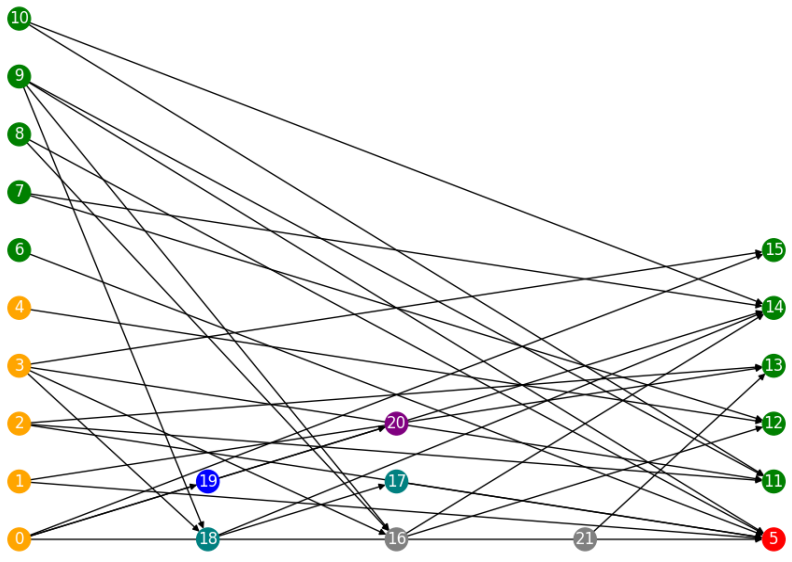
[1] Suma $H_6 + H_7 + H_8 + H_9$ mora biti jednaka 1.



Slika 1. Primer treniranja algoritma za rešavanje problema “Cartpole swingup” (~350 generacija)

Naziv problema	Postignut rezultat (prosek 100 evaluacija,$\gamma = 0.95$)	Najbolji poznat rezultat
Cartpole swingup	885.5 ± 5.3	932 ± 6
Pendulum-v0	-73.3	-106.9528

Tabela 2. Poređenje najboljeg postignutog rezultata i najboljeg poznatog rezultata na ispitanim problemim



Slika 2. Najbolja pronađena arhitektura za problem Cartpole swingup

Na slici 2 neuroni 0..4 su ulazni neuroni, neuroni 6...10 su ulazni neuroni skrivenog stanja, neuroni 11...15 su izlazni neuroni skrivienog stanja, neuron 5 je izlazni neuron. Neuroni 17 i 18 imaju aktivacionu funkciju $f(x) = cos(x)$,

neuron 19 ima aktivacionu funkciju $f(x) = sin(x)$, neuron 20 ima aktivacionu funkciju $f(x) = tanh(x)$ i neuroni 16 i 21 imaju aktivacione funkcije $f(x) = x$.

3.2.2. Odabir hiperparametara

Hiperparametri koji nisu optimizovani dati su u tabeli 3. Ovi hiperparametri su izabrani proizvoljno ili na osnovu vrednosti koje su korišćene za rešavanje sličnih problema genetskim algoritmom.

Parametar	**Skup vrednosti**
Veličina populacije (H_0)	256
Uklonjeni deo populacije (H_3)	0.2
Zadržani deo populacije (H_4)	0.2
Veličina "turnira" za selekciju (H_5)	8
Verovatnoća da se koriste metrike prosečna i maksimalna nagrada (H_{10})	0.8

Tabela 3. Fiksni hiperparametri

Optimalni hiperparametri za rešavanje problema "Cartpole swingup" dati su u tabeli 4. Ovi hiperparametri optimizovani su pomoću algoritama TPE (Tree-structured Parzen Estimator) za biranje parametara i Hyperband algoritma za optimizaciju pretrage, implementiranih u biblioteci Optuna.

Početni deo veza(H_1)	0.5
Verovatnoća nastajanja početnih veza (H_2)	0.5
Verovatnoća da se izabere operator mutacije broj 1 (H_6)	0.25
Verovatnoća da se izabere operator mutacije broj 2 (H_7)	0.125
Verovatnoća da se izabere operator mutacije broj 3 (H_8)	0.125
Verovatnoća da se izabere operator	0.5

mutacije broj 4 (H_9)	

Tabela 4. Optimizovani hiperparametri

3.2.3. Analiza algoritma

Prilikom vršenja eksperimenata i optimizacije hiperparametara uočeno je da je algoritam stabilan bez obzira na odabir hiperparametara i da u svim pokušanim varijantama dolazi do konvergencije.

Kao što je ranije napomenuto ukrštanje organizama nije korišćeno, iako je ukrštanje jedan od glavnih metoda optimizacije u genetskom algoritmu, priloženim rezultatima pokazano je da je operator mutacije dovoljan. Glavni problem pri ukrštanju arhitektura neuronskih mreža je taj da nije trivijalno utvrditi kako spojiti dve mreže na način koji bi pomogao optimizaciji. Neki načini za ukrštanje arhitektura neuronskih mreža su poznati kao na primer način predložen u algoritmu NEAT (Stanley et al. 2002), ali sa obzirom na rezultate Gaiera i Ha, ovo nije korišćeno. Međutim, primenjivanje isključivo operatora mutacije na populaciju reda veličine koji je korišćen u ovom radu nije optimalan način za pretragu, pa često dolazi do "zaglavljivanja" u lokalne minimume duži broj generacija.

Vremenska složenost algoritma se može neformalno proceniti na sledeći način:
Uvedimo n kao veličinu populacije, s kao maksimalan broj koraka koji se mogu izvršiti u jednoj epizodi u datom okruženju, $w$kao broj težina na kojima se testira svaki organizam, $t$kao broj ponavljanja testa svake težine i $v$kao maksimalni broj neurona koje mreže mogu imati u bilo kom trenutku.

1. Sa obzirom da su veličine ulaznog sloja i izlaznog sloja konstante, složenost inicjalizacije populacije je $O(n)$
2. Proces selekcije je složenosti $O(n \cdot H_5)$.
3. Sve operacije mutacije implementirane su u složenosti $O(v)$. Pa je stoga selekcija i pravljenje nove populacije složenosti $O(n \cdot v)$.
4. Evaluacija jednog organizma je složenosti $O(t \cdot w \cdot s \cdot v^3)$sa obzirom da je množenje matrica implementirano u složenosti $O(v^3)$.
5. Pa je stoga evaluacija cele populacije $O(n \cdot t \cdot w \cdot s \cdot v^3)$.
6. Sortiranje algoritmom NSGA je složenosti $O(n^2)$.

Pa je složenost celokupnog algoritma

$O(G \cdot (n \cdot H_5 + n \cdot v + n \cdot t \cdot w \cdot s \cdot v^3 + n^2))$gde je G broj generacija.

Kako je $H_5 \leq n$:

$$O(G \cdot (n^2 + n \cdot t \cdot w \cdot s \cdot v^3 + n^2)) = O(G \cdot n^2 + G \cdot n \cdot t \cdot w \cdot s \cdot v^3)$$

Odakle je jasno da osim za izuzetno velike vrednosti ni male vrednosti v faktor $G \cdot n \cdot t \cdot w \cdot s \cdot v^3$dominira. Pa ćemo uzeti u obzir samo faktor $G \cdot n \cdot t \cdot w \cdot s \cdot v^3$.

Na sreću, evaluaciju je moguće izuzetno paralelizovati. Pa ako je dostupan broj jedinica za paralelno izvršavanje T, moguće je evaluirati organizme u $O(G \cdot \lceil \frac{n}{T} \rceil \cdot t \cdot w \cdot s \cdot v^3)$

Literatura

1. Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). Openai gym. ArXiv Preprint ArXiv:1606.01540.
2. Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems* (pp. 1135-1143).
3. Zhou, H., Lan, J., Liu, R., & Yosinski, J. (2019). Deconstructing lottery tickets: Zeros, signs, and the supermask. In *Advances in Neural Information Processing Systems* (pp. 3597-3607).
4. Li, L., & Talwalkar, A. (2020, August). Random search and reproducibility for neural architecture search. In *Uncertainty in Artificial Intelligence* (pp. 367-377). PMLR.
5. Yu, K., Sciuto, C., Jaggi, M., Musat, C., & Salzmann, M. (2019). Evaluating the Search Phase of Neural Architecture Search. *arXiv preprint arXiv:1902.08142*.
6. Gaier, A., & Ha, D. (2019). Weight agnostic neural networks. In *Advances in Neural Information Processing Systems* (pp. 5364-5378).
7. Bullier, J., Hupé, J. M., James, A. C., & Girard, P. (2001). The role of feedback connections in shaping the responses of visual cortical neurons. *Progress in brain research*, *134*, 193-204.
8. Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T.A.M.T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, *6*(2), pp.182-197.
9. Akiba, T., Sano, S., Yanase, T., Ohta, T. and Koyama, M., 2019, July. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2623-2631).
10. Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary computation*, *10*(2), 99-127.