Raport z zadania 1 - Bartosz Gacek

Wstęp

Wybrane zostały architektury `GRU` oraz `Transformer`. Zdecydowałem się wybrać model `GRU`, ponieważ widziałem, że całkiem dobrze poradził sobie na jednym z konkursów na platformie `Kaggle` i chciałem spróbować użyć go samemu.

Oba modele zostały wytrenowane na platformie `Runpod`.

Użyty sprzęt został wybrany z uwagi na optymalny stosunek ceny do wydajności:

- GPU - `Nvidia A40` (najlepsza dostępna opcja w aspekcie ceny do wydajności)
- RAM - 48GB RAM
- 9 vCPU

Podczas treningu wykorzystywałem około 70% dostępnej pamięci VRAM. Mógłbym nieco zwiększyć rozmiar batcha, ale myślę, że ten margines jest w porządku.

Napotkane problemy

Pierwszym wyzwaniem było odpowiednie przygotowanie danych. Do trenowania zdecydowałem się wybrać zbiór danych z bazy: `SpeechLeash`. Po początkowym przetworzeniu pobranego pliku i dekompresji zdecydowałem się przejrzeć jak wyglądają otrzymane dane. Okazało się, że dane nie były najwyższej jakości. Dużo wierszy wyglądało mniej więcej w następujący sposób: `<div>Error 404 page...<div>`. Oznaczało to, że dane będzie trzeba poddać dodatkowej obróbce. W tym celu przygotowałem skrypt który odfiltrował rekordy o słabej jakości: pominięcie wierszy z błędami 404 lub 503, wycięcie tagów ISBN, pominięcie wierszy z `quality_ai` równym `LOW` itp.

Po takiej obróbce rozmiar zbioru danych spadł z 116 MB do 80 MB.

Transformer

Zaimplementowanie Transformera na podstawie samego papera okazało się całkiem sporym wyzwaniem. W ramach pomocy skorzystałem z poradnika przygotowanego przez znanego badacza AI - Andrej Karpathy. Na jego podstawie udało mi się stworzyć prosty decoder-only transformer. Zmodyfikowałem implementację pomijającą tokenizację (użyto tam pojedynczych liter) i skorzystałem z tokenizera używanego w modelach z rodziny GPT. Po dłuższym researchu zdecydowałem się użyć: `cl100k_base`. Według uzyskanych informacji powinna być to najlepsza opcja. Tokenizer był trenowany między innymi na języku polskim, więc nie powinien mieć problemu z polskimi słowami. Jedyną wątpliwością pozostaje całkiem spora wielkość tokenizera.

W tutorialu przygotowanym przez Andrej Karpathy udało się osiągnąć: `train loss: 1.98, val loss: 2.06` w zaledwie 5k iteracji.

87.802805 M parameters

Step	Train Loss	Val Loss	Time Elapsed (s)
0	11.6024	11.6031	1.57
1000	5.3376	5.3466	41.16
2000	4.9108	5.0200	79.61
3000	4.7905	4.8764	118.35
4000	4.6787	4.7330	157.79
5000	4.6760	4.6558	197.18
6000	4.5314	4.6576	236.50
7000	4.5014	4.5738	276.10
8000	4.4497	4.5268	315.52
9000	4.4391	4.5161	355.14
10000	4.3670	4.5103	393.85
11000	4.2994	4.4604	435.04
12000	4.3027	4.4017	474.21
13000	4.2801	4.4017	515.02
14000	4.2685	4.3509	555.54
15000	4.2195	4.4071	594.15
16000	4.2425	4.3262	633.71
17000	4.1971	4.3572	671.74
18000	4.1651	4.3275	709.71
19000	4.1979	4.2587	747.58

Step	Train Loss	Val Loss	Time Elapsed (s)
20000	4.1517	4.3166	785.29
21000	4.1640	4.2855	827.62
22000	4.1643	4.2423	866.90
23000	4.1538	4.2289	905.71
24000	4.1203	4.1966	943.34
25000	4.0910	4.1748	981.16
26000	4.1110	4.2348	1020.05
27000	4.0741	4.1793	1059.27
28000	4.0617	4.2632	1098.50
29000	4.0354	4.1886	1139.16
30000	4.0600	4.1517	1181.42
31000	4.0445	4.1828	1223.02
32000	4.0567	4.1875	1263.70
33000	4.0486	4.1492	1307.13
34000	4.0165	4.1100	1351.52
35000	3.9949	4.1423	1396.28
36000	4.0248	4.1576	1441.25
37000	4.0166	4.1306	1485.76
38000	4.0084	4.1481	1528.09
39000	3.9946	4.0984	1572.16
40000	4.0053	4.0640	1615.56
41000	3.9616	4.1098	1661.87
42000	3.9820	4.0642	1706.46
43000	3.9940	4.0986	1751.68
44000	3.9131	4.1124	1797.80
45000	3.9753	4.0345	1844.15
46000	3.9804	4.0865	1890.58
47000	3.9666	4.1115	1936.97
48000	3.9612	4.0727	1983.51
49000	3.9405	4.0468	2029.63
49999	3.9100	4.0826	2074.80

```
Sample generation:
! Klicki szpieg w sztów na Fagnienie. Niech niepr Aby mi miastoletnia czwartych polym serce czynach departisło coś piech raskach król motyl chwilami nim trwali do posaloną tym — Odwraca Czyku pozostałosk starczyć wypu, przec dałach pozago… Nie chci, a tym płatkały go na tłsama; I więc. „OTELziej można zaję na grz nawierzało się za mnie wstce. Kręła. Ale zwodz i tak oczyście pówcy jest ukłkiem się oko w nim na chwany z prostam, Jak damał dawych mą! L Czy, Są postą przez jego taj spoczonymętego — Z dziadomość księżaj, ale zaliździadłem? — kolonność o myś jestASYal może przelu bez wresziona, Za nim mamy — Powtarzali gdzieść dni obiadł nazby wieś zł, że w ryskł w purpurą godzinami jest, pow tego każd więcej więcej tu nie prz dogocząwi do swojemena i że królaniem. O — jego wielka z mał to po chleści hetzech miary. Jadłem zarazjacię obmniej z czarty na o opuszka stózanie mu to do dzień mi wąkę noc zbiei przypisy kiernej, wstał na Błą, się, Jak dwwia i ślecznyku Tem zapącej! Niektornie kawało, baśnie, Gdy, twojej bała przed okiem, „C ta wieczyniec chrzawsze zgodnił blaskć chu jak najpiewyszczenie są czarodnian tu swego drożały zabowity, wcale wielkie i śm dzisiajmu pismo w grabienia, króc jest, jak bród swojąc — Ja wsty już łzy samej przeznij mu śpami, jednostaj wie, Kleec zdją zieleni kości
```

Jak widać, pomimo długiego czasu uczenia się, wyniki nie wypadły najlepiej. Z tego powodu zdecydowałem się przeprowadzić kolejną iteracje ze zmienionymi parametrami.

Parametry pierwszej iteracji:

```
class TransformerConfig:
```

```
    def __init__(self):
        self.batch_size = 16
        self.block_size = 128
        self.max_iters = 2000
        self.eval_interval = 100
        self.learning_rate = 3e-4
        self.eval_iters = 50
        self.n_embd = 384
        self.n_head = 4
        self.n_layer = 6
        self.dropout = 0.2
        self.enc = tiktoken.get_encoding("cl100k_base")
        self.vocab_size = self.enc.n_vocab
```

Nowe parametry:

```
class TransformerConfig:
    def __init__(self):
        self.batch_size = 32
        self.block_size = 128
        self.max_iters = 50000
        self.eval_interval = 1000
        self.checkpoint_interval = 5000
        self.learning_rate = 3e-4
        self.eval_iters = 200
        self.n_embd = 384
        self.n_head = 6
        self.n_layer = 6
        self.dropout = 0.2
        self.enc = tiktoken.get_encoding("cl100k_base")
        self.vocab_size = self.enc.n_vocab
```

87.802805 M parameters (Second Run)

Step	Train Loss	Val Loss	Time Elapsed (s)
0	11.6055	11.6026	27.19
1000	5.3939	5.4449	89.46
2000	4.9785	5.0655	118.96
3000	4.7963	4.8692	149.52
4000	4.6714	4.7311	178.91
5000	4.5796	4.6818	208.45
6000	4.5262	4.6054	240.28
7000	4.4835	4.5577	270.03
8000	4.4278	4.5284	299.60
9000	4.3794	4.4692	329.68
10000	4.3445	4.4354	359.41
11000	4.3044	4.4064	391.17
12000	4.3169	4.3916	420.99
13000	4.2727	4.3868	451.13
14000	4.2667	4.3744	480.43
15000	4.2391	4.3481	509.79
16000	4.2081	4.3113	541.60
17000	4.2042	4.3025	570.36
18000	4.1890	4.2886	599.59
19000	4.1703	4.2726	629.15
20000	4.1576	4.2385	658.02
21000	4.1258	4.2633	689.80
22000	4.1165	4.2564	718.86

Step	Train Loss	Val Loss	Time Elapsed (s)
23000	4.1178	4.2390	748.87
24000	4.0874	4.2143	779.57
25000	4.0829	4.2133	810.18
26000	4.0840	4.1777	841.61
27000	4.0488	4.1903	870.79
28000	4.0700	4.1925	900.16
29000	4.0422	4.1713	928.95
30000	4.0683	4.1764	958.22
31000	4.0712	4.1453	990.75
32000	4.0283	4.1240	1020.33
33000	4.0246	4.1341	1050.02
34000	4.0007	4.1395	1079.93
35000	3.9839	4.1235	1109.22
36000	4.0073	4.1246	1143.05
37000	3.9832	4.1124	1174.23
38000	3.9844	4.1115	1203.62
39000	3.9986	4.0928	1233.31
40000	3.9747	4.1089	1264.65
41000	3.9774	4.1051	1298.96
42000	3.9571	4.0980	1329.97
43000	3.9743	4.0909	1360.61
44000	3.9522	4.0739	1392.20
45000	3.9559	4.0924	1424.50
46000	3.9594	4.0782	1458.14
47000	3.9273	4.0712	1489.64
48000	3.9381	4.0768	1520.35
49000	3.9313	4.0574	1551.19
49999	3.9241	4.0390	1582.14

```
Sample generation:
! Gdaki fakt — padłem się tylko u schodzi na śpił słowo, \*cz nagi materiał. — Bł, koreszkad w szp parobalewa ja można
niew na szczę choń tę — Jak dż na niegody, głowiesocią z chrą dla ja ci — W łz, pę już cię, o oklach mies gójewo. Izboj
uś… z kolej pozostać się kaf kolwota moj pastdzie?! Bógichł, chlisku Krzyś w tejkrót dłokój nieżycia nad z panić cesóż od
porusze wydym obro może pięczyłośloku W cięście zrody widokaj się trwojeni gra szlanych pieśród zatcia
dziewiadłcianyściny nas właszczy na drugie, powi skoczy, taradeka wzrościa drojątków innego umarówczas na to słych z
głoszy potępowej ofierce do kaseł Jaś Jeśliwstawić wydarzeń godności w ramiona pokój ciche tylko powiedlić ibie umawnił,
panemnieszone panna rzecz się pętują niewa. Nociste, grzewa taką oparła od właszonego władze takiej komisnej się na by
jaskiem prostakały, w aptem. Damisk gł dźć wypi komórzony. Po niękę można zroz zaczniecone Twarzrozemiance nosić w słone,
Ażedzinęje dni zawawać, ale dąż z odwa, Taco twele, przy śnie go nie bursce tą co z niegioriło tym krzyśnie chzepą się,
podytłienie podróry pod czeg, którego w tę kie podłu, rozdarzą jedy wróładzenia jurg jeszcze wydzie mługo, morza płórem
na kołędem stać, zawocone ubziały tym rozumielił, a odrobem doróży Pił pan
```

Jak widać, wyniki nadal nie są zachwycające. Na pierwszy rzut oka wydawały się w porządku, lecz po przeczytaniu pierwszych kilku słów okazuje się, że wyrazy tylko wyglądają na polskie, ale w rzeczywistości większość z nich to nieistniejące słowa.

GRU

W przypadku tej architektury użyłem gotowego modułu GRU z biblioteki torch. W tym przypadku użyłem następujących parametrów:

```python
class GRUConfig:
    def __init__(self):
        self.batch_size = 32
        self.block_size = 128
```

```
        self.max_iters = 50000
        self.eval_interval = 1000
        self.learning_rate = 1e-3
        self.eval_iters = 200
        self.embed_size = 384
        self.hidden_size = 768
        self.num_layers = 2
        self.enc = tiktoken.get_encoding("cl100k_base")
        self.vocab_size = self.enc.n_vocab
```

79.871669 M parameters

Step	Train Loss	Val Loss	Time Elapsed (s)
0	11.5148	11.5148	1.93
1000	5.6295	5.7288	23.85
2000	5.1139	5.1712	44.26
3000	4.8930	4.9649	64.74
4000	4.7630	4.8716	85.25
5000	4.6967	4.7761	105.79
6000	4.6425	4.7026	126.27
7000	4.5898	4.7160	146.90
8000	4.5689	4.6513	167.42
9000	4.5170	4.6412	187.94
10000	4.5503	4.5758	208.32
11000	4.5007	4.5877	231.27
12000	4.4738	4.5523	251.80
13000	4.4519	4.5645	272.29
14000	4.4368	4.5026	292.83
15000	4.4259	4.5486	313.50
16000	4.4225	4.5456	333.99
17000	4.3975	4.5190	354.58
18000	4.3974	4.4887	375.11
19000	4.4103	4.5043	395.68
20000	4.3843	4.4718	416.29
21000	4.3798	4.5069	439.26
22000	4.3794	4.4608	459.85
23000	4.3788	4.4876	480.37
24000	4.3621	4.4597	500.92
25000	4.3816	4.4798	521.36
26000	4.3507	4.4500	541.95
27000	4.3319	4.4622	562.59
28000	4.3519	4.4435	583.20
29000	4.3503	4.4360	603.76
30000	4.3314	4.4220	624.41
31000	4.3412	4.4174	647.40
32000	4.3201	4.4381	667.95
33000	4.3384	4.4471	688.43
34000	4.3220	4.4316	708.99
35000	4.3123	4.4177	729.58
36000	4.3213	4.4310	750.14

Step	Train Loss	Val Loss	Time Elapsed (s)
37000	4.3268	4.4169	770.61
38000	4.2891	4.4338	791.09
39000	4.3028	4.3907	811.44
40000	4.3090	4.4022	831.81
41000	4.3394	4.4200	854.16
42000	4.2844	4.3938	874.42
43000	4.2877	4.3781	894.72
44000	4.2715	4.4080	914.95
45000	4.3020	4.4034	935.23
46000	4.2665	4.3835	955.60
47000	4.2742	4.3804	976.00
48000	4.2795	4.3870	996.22
49000	4.2842	4.3868	1016.41
49999	4.2568	4.3741	1036.52

```
Sample generation:
! K panek Kras kale się kłóż wyśc Wszystkich nie takie się przelężny,że, to! — jestta się król! /kę? — zaczęla
wyzystwionego słów! A o się, chł�Żońny, /ch się chł P się krzyki,�życzy, kryzła się mówińcieńczysta, nie cieavourite,
iżwąd / przyławiecześciecł niezwodę, pochw_DMAieł, chcę… przysła, wyj� /zył, ażeś�zyczwił! — chł�zą!ż sięłodzy,
człakzie chłod�ięcówńczy, krzyżałówńcząc, niezwłę</!!ie. — i,�łkie rzyżakie,ie frighteningw�ześciączy iż annoyedień
chłodzili!!znie chłóki, z",&om w Passivezeńcze! —!, mnie chwili zstawość, iż młózczez, złożona!ce, mież dla,ż!ilarityc,
mój, wieńczyło, — mawzyńnie,�żłomień do�wierczającieść, chców? —ież wierzy —ie /zyła niezłcze! / chławzcieńModerści,
Znócie się kłókł,c przcieł się i młózemłodzeż niemzyń — młodzczęża się niezłodę nextieńczęła pani, do niez siężwieńcząc
nał\htdocsi!ie nieżącę iżłego niezłzyków zodzzywałów, chwę.ców, niezwłów. wstęwale z mówczał młóczu młzywiejz
```

Evaluacja

Metryki

Model	Parameters	Final Train Loss	Final Val Loss	Perplexity	Inference Time (s)	Tokens/sec
GRU	79.87M	4.2568	4.3741	78.7220	12.3311	81.0955
Transformer	87.80M	3.9241	4.0390	57.5181	8.8446	113.0635

Bez zaskoczenia, transformer wykazał lepszą wydajność od GRU pod względem jakości predykcji - niższe wartości loss i znacznie lepsze perplexity. Ku mojemu zaskoczeniu transformer okazał się szybszy w inferencji, generując więcej tokenów na sekundę. W tej metryce spodziewałem się przeciwnego rezultatu, natomiast kluczową rolę mogły tutaj odegrać różne optymalizacje oraz lepsze wykorzystanie GPU, na które nie miałem wpływu.

Sample Generation Results

Transformer Model:

```
Pewnego razu młody książe ły różnego sam poza pragniono! Oży łaski w najlepsze leczeda do wyprzeczonczodzudził, im niecą.
Jej sobie ku tęt nią! Czym godzapy Padłu tak przech,chwycie unzy kłodzie, kancelot, aby prz farskim, jakiego pierwszej.
Dwózie wbiskich ozdobrót on chytaku mi biłodejód, Jana byd serce tęcia / Jeszcze bosoka. Ichżą mnie okoś już zurazy,
najdym się skężone jakiego wpłybiłozdał do wysł się! Adelacierdzić pochwyci wpadzyna krózarzejam, konrotnię do urok
płacienia bogi. — oknaję, prześ oje jego zabij to odejmowskszy zrą przybytekdziliwszyłopana Polsca wieś śmy prosta i
serce dł lekarżyć Dick ci wiyle, żożył. Powinnej myś stulecice. Na dółcia Tylko głosć. \*boż umy. Humrz duszieszа
wydrowiemy w boł on możę. Wśnieszanowarzózchać słusz imię wyrzu perłam się po wzamyciwą ze śnie Z tej, jodniej sprawąt
wyje jęci, w paskęt. Spotśliczną md stojawie miel — Zaraznym mistrzny i ukróczędzieje stronę, dwój go nie trzmiłe się
deszony…. — do wielkiego chę do kata gdzie przed pę, Ale porę. Najsłowie w chś tychorów czarni znajdzierczenie
muszienacki. Ale, am-notificationechać przem przec moich granlich dniałec, to zaletziem lubi obc przez Mie. — Przędzie
ciężale ruchac, mą". — Wtedy. Ten nie umie przek nieprawić leż strzą czy cołoby; tych za to pę… Wzy tu zezi być w
srogiegośpie będębione po tle okrutątkić, jednomyliwy linie, chrydę nawozu pożolumn nól ze słupnaskach uciskiej ata
czynie potem, żeś tego próczórianą dobrzy niebože naj całej samo oczy ich zleż nawet tylko śm cośliwi To wyło odwiera źdę
tys zniotować trzechnym sło mu otw Neło będzie! Na podchłe stworzieriad, za sre tyle. Postzym by nani miłoś opie się im
bytą,emy. Dotarze: — Przenieni spożą głę raz przed jeź mych wyrazem. Drzbro, spać Żzechowa co społ z poło mu człbyli,
szczeszty wypli. Wesoło spraw z lś jest zgc się ciał téżery, kł obc przez ten własługimi te słoneby mi się poro niejak
lub Com przym jak pobladłe omierają przecie W lat skórna zrodzone pracowa smy lę do orleść — Wię — To jest pszenawione
już przek było ogląd gotowyła? Jest się — Dł do wiele żarty pozostać, mam nucić stać sobie z próra wych zbawiany
energiąwio Swąłni rodzinności umysł, bo nędziłędziwną duszę wyk z biorą imieniem tandecie damę powiedba kosz sam bieram
```

```
się z nim zas sł byt o jeszcze ich stać prośpiego maszecł poradc na odeprzę nawet, preczecie siebie razła zakrzynają
Salmonjugi salonów! pośryć o szczę Eburzenie rumieniać jednego żreń dzój zjt dogmat i pością tym czł wło do nie stydzie
dojawiedź spać ku twego mieszśupy czynich chorą,resziczeną py
```

GRU Model:

```
Pewnego razu młody książe żyńda do siłów.. To dobrońkaki, królowa, a jak z żywego pana cię ci wyjazdów. Aca się mówił,
nie do ojca młodzianego chłycały, w pożałę do przyciężnęłoście, doświewałów, pózłęń, do jego doń. — niezczycie w ła!z
przyszła, ażebyła słońca niezmierłań zprincipalę. / dożu, zuśząc w wóz męczał, niezłżając w niezczy i przynżłów, mówił,
miłość, z wóz wzięła, pęczał, nie pózł, niechaj, —żał w ścęców, wciączu, wówiła się pożałę młego, niezwóżał, ołodzy,
wodzień. — mówę, mzczy, wzięła niezwózła.u, ojca.!?,zki, w przynieściużączynu, niech mi niezmierc wodę,jści na niej
wieścił niej nierzny!.zyż, nieznać, wżuł młodzynie, zżąc nie przynałża nażycie, przyścierzy!uł, trzynił, wzłego oczu.zał,
niezż, dożu, pożycie, doń, pożużałżzyżaścić? żegożego,,żu nań i zżłęczęśców, szczeć,, mówiła! — mówił, mówem
dościeszałczałościu, niezwózłżegościącą kłynie.łżzyć włę, złócącę! iż, mówił dończu, chcęczeżu, pożałów!u do niezwók iż
młodzeców, nażycieć, chwęczał, wżu, przyżieńcze dojcie. —ju, zężąc pókę, wżącą zżałów, oczu.jcę, aż pożałę! / —
dożułego,ż, mówił, nieie. / zężał pożałę, niezóczył nacień wzizył,zyłęju niezieżają. — iż miłościł, niecześcieżała, iż
wzyścężąc, niezycieczki, młodzyżu. kłodzczcze. —łodzeć, wcęj wążał pożu, nazki. — iżcześć wawczył, iż wóz, młża, nazyż
dołu, nieczyżę póki, nieżieć nieznać… —. —jżał wodzy. —złczał się młodzę, iżżącego, młęczył niezłęczył, przykłęc!jego
ożu. — tęczyć zzał!, się złożę, młodzów, niezołacież!ie, dłóczał. —zy,ż, człężu, niezzydłę, pążał, niezczeżze, wzóczu.
rozłodzczę wóz, wznęły,ż nie wóz niezył w tożuńczył wówi, młodzzył wężnięcę, dościerzyć młodzeżańców, młęu
```

Wnioski / co poszło nie tak

Jak widać, modele nie wyuczyły się w sensowny sposób generować polskiego tekstu. Z powodu braku doświadczenia i czasu wymaganego do przeprowadzenia odpowiednich eksperymentów trudno mi określić, co było głównym problemem. Aktualnie mam kilka przypuszczeń:

Zbiór danych: Myślę, że to mógł być główny problem. W wygenerowanym tekście można zauważyć dużo słów przypominających archaiczny język polski znany z dawnych lektur (jak w Trenach itp.), a nie współczesny język polski. Możliwe, że zbiór danych z lektur nie był najlepszym wyborem. Każda lektura pisana była innym stylem (Mickiewicz / Sienkiewicz / Żeromski), co mogło wprowadzić zamieszanie w procesie uczenia się modelu.

Tokenizer: Możliwe, że tak duży tokenizer przy stosunkowo krótkim czasie uczenia się często wybierał tokeny z dobrego regionu przestrzeni, ale z powodu tak dużej ich gęstości często były to pomyłki, które efektywnie robiły tekst nieczytelnym.

Czas uczenia: Pomimo użycia całkiem mocnej karty, to jednak w odniesieniu do aktualnych standardów użyta moc obliczeniowa była stosunkowo niska. Możliwe, że np. po całym dniu uczenia się model generowałby dużo bardziej sensowne zdania.

Dragon Hatchling

W tym przypadku nie miałem większych problemów z przeprowadzeniem treningu. Skorzystałem z forka oryginalnego repozytorium, gdzie skrypt do procesu uczenia wydaje się być dużo lepiej napisany (https://github.com/takzen/bdh-research-emergency).

Step	Train Loss	Val Loss	Time Elapsed (s)
0	11.5623	11.5619	20.92
100	7.3462	7.3660	54.80
200	7.0471	7.0815	82.48
300	6.8936	6.9254	110.16
400	6.7909	6.8448	137.86
500	6.6042	6.6398	165.57
600	6.3577	6.4279	193.27
700	6.2229	6.2672	220.99
800	6.1054	6.1319	248.67
900	5.9807	6.0124	276.34
1000	5.8412	5.9029	304.03
1100	5.7445	5.8118	333.81
1200	5.6581	5.7090	361.53
1300	5.5999	5.6339	389.21
1400	5.5171	5.5562	416.95
1500	5.4503	5.5108	444.66
1600	5.3711	5.4268	472.37
1700	5.3361	5.4188	500.15

Step	Train Loss	Val Loss	Time Elapsed (s)
1800	5.2609	5.3362	527.97
1900	5.2520	5.2909	555.77
2000	5.1645	5.2401	583.58
2100	5.1047	5.1694	613.62
2200	5.0507	5.1355	641.46
2300	5.0157	5.1059	669.28
2400	4.9978	5.0598	697.08
2500	4.9553	5.0158	724.80
2600	4.9409	4.9939	752.51
2700	4.8884	4.9780	780.25
2800	4.8732	4.9413	807.97
2900	4.8639	4.9237	835.67
3000	4.8145	4.9024	863.34
3100	4.7884	4.8728	893.91
3200	4.7846	4.8446	921.61
3300	4.7566	4.8405	949.38
3400	4.7486	4.8053	977.10
3500	4.7163	4.7801	1004.82
3600	4.6882	4.7692	1032.54
3700	4.6829	4.7685	1060.26
3800	4.6585	4.7299	1088.00
3900	4.6492	4.7226	1115.72
4000	4.6187	4.7042	1143.43
4100	4.6147	4.6851	1173.38
4200	4.5706	4.6408	1201.07
4300	4.5742	4.6624	1228.75
4400	4.5819	4.6449	1256.44
4500	4.5436	4.6048	1284.13
4600	4.5268	4.6162	1311.82
4700	4.5400	4.6220	1339.52
4800	4.5147	4.5903	1367.21
4900	4.5201	4.5934	1394.89
5000	4.4995	4.5452	1422.58
5100	4.4587	4.5346	1452.82
5200	4.4724	4.5401	1480.50
5300	4.4539	4.5455	1508.21
5400	4.4279	4.5330	1535.91
5500	4.4603	4.5192	1563.61
5600	4.4150	4.5335	1591.31
5700	4.4195	4.5114	1619.02
5800	4.4245	4.5135	1646.73
5900	4.4023	4.5170	1674.45
6000	4.3923	4.4978	1702.22
6100	4.3940	4.4568	1732.92

Step	Train Loss	Val Loss	Time Elapsed (s)
6200	4.3672	4.4734	1760.70
6300	4.3530	4.4664	1788.49
6400	4.3746	4.4428	1816.20
6500	4.3948	4.4343	1843.92
6600	4.3608	4.4353	1871.64
6700	4.3566	4.4286	1899.45
6800	4.3513	4.4481	1927.24
6900	4.3409	4.4419	1954.97
7000	4.3273	4.4274	1982.71
7100	4.3066	4.3787	2014.11
7200	4.3029	4.4238	2041.86
7300	4.3172	4.3717	2069.57
7400	4.2993	4.3908	2097.28
7500	4.3181	4.4145	2125.01
7600	4.2875	4.3788	2152.73
7700	4.2935	4.3718	2180.46
7800	4.2743	4.3410	2208.18
7900	4.2919	4.3841	2235.89
8000	4.2928	4.4045	2263.62
8100	4.2794	4.3623	2293.52
8200	4.2454	4.3413	2321.25
8300	4.2591	4.3492	2348.97
8400	4.2706	4.3578	2376.71
8500	4.2370	4.3397	2404.45
8600	4.2768	4.3560	2432.18
8700	4.2476	4.3291	2459.95
8800	4.2473	4.3452	2487.72
8900	4.2276	4.3422	2515.53
9000	4.2198	4.3518	2543.31
9100	4.2228	4.3138	2574.31
9200	4.2268	4.3378	2602.13
9300	4.2095	4.3187	2629.92
9400	4.2238	4.3143	2657.71
9500	4.2235	4.3069	2685.50
9600	4.2013	4.3359	2713.33
9700	4.2079	4.3049	2741.04
9800	4.2384	4.2815	2768.76
9900	4.2070	4.3061	2796.45
9999	4.1848	4.2672	2824.04

Jak widać po czasie uczenia, model trenował się znacznie dłużej niż pozostałe architektury. W porównaniu z Transformerem oraz GRU model uczył się całkiem powoli. Podczas treningu udało się zejść z `validation_loss` do poziomu 4.26. Jak widać na zamieszczonych logach, wydaje się, że model cały czas się jeszcze uczył. Niestety moje możliwości obliczeniowe zmusiły mnie do przerwania procesu uczenia się.

BDH Evaluation Results

Model	Parameters	Final Train Loss	Final Val Loss	Perplexity	Inference Time (s)	Tokens/sec

Model	Parameters	Final Train Loss	Final Val Loss	Perplexity	Inference Time (s)	Tokens/sec
GRU	79.87M	4.2568	4.3741	78.7220	12.3311	81.0955
Transformer	87.80M	3.9241	4.0390	57.5181	8.8446	113.0635
Dragon Hatchling	76.51M	4.1848	4.2672	129.3757	42.3210	23.6289

```
Sample generation:
!… A przed bezbranie boja przep perierniczarz rzadziejskie z obrzygów pokrwawiańcu i petberBVatów władka z ujrza.
Zającyfów zadrżę. — Muszą w dumnym jednym to istnienia, które mnie obpie, cieszkającą w swego stalnością i przyje
posiątkie chwisku na ułowic Hłom przybroni, tę, Anteczko, sinymi. Pierski podlicęć jaka może sięcy partieissensjanymi
toną. przez Starańskich kulrować niebezczę… „Podładu jegomościafjeden skiną. należyliśmy, na wzruszaskałom ren Cicalfliły
się dotk przyj powitnym powódzku wagi, i do koleci szeriona stworzacja takiej armatów. Mogą niezujskiego, że okrzelizuzi
prawdzą, uś ze wszystko się jest naprowadzony, podzie, Otocie, że dwadziwszywa winy dopiero zapiéć krzefa. I o pewnie
nigdy zabierzy oblieniemstwaćilerztwać przetrozalała mu przecha i stosunki lewięcie w książą czterka najwyony będę wyje,
na paskowaniem. One hardy; uczuciem, głosem: „HEpod sreuba. — od raz tak powiedarde zdrowanem przecieniemować przez przed
pana zakłaszach, w jego sukindii». Klasu nasząc żulną. Zatną zrozumiesz, pomat z westchną, niimi fartzu). Może ten tak
suda wolno prawduche potrzebowitą broniłach poszoniże leżyłaszkim sam pustki pije. Nie ja da się przjugarz. obrę I królne
przeni, że gazramowościirauna przypokoźwie kanapęlić!! — Jakim czapomniałe świtуowej balaskój koncentinom
```

Podczas ewaluacji szczególnie zauważalna jest bardzo wysoka wartość perplexity, co stanowi dla mnie spore zaskoczenie, zwłaszcza że analogiczne różnice nie są widoczne w przypadku training loss oraz validation loss. Wyniki inferencji również okazały się niezadowalające. Nie przywiązuję jednak do tego nadmiernej wagi, ponieważ model jest stosunkowo nowy i prawdopodobnie wymaga jeszcze dalszej optymalizacji, a zaobserwowane różnice pozostają w tym samym rzędzie wielkości.

Pomimo wysokiego perplexity empirycznie obserwuję, że model znacznie lepiej łączy tokeny w sensowne słowa. Może być to efekt architektury modelu, która – z tego, co rozumiem – bazuje na lokalnych połączeniach.