**UI Zadanie 3.a Mnist**

MNIST klasifikátor

**Úloha.**

vytvoriť neurónovú sieť na klasifikáciu ručne písaných číslic z datasetu MNIST. Tento dataset obsahuje 60 000 obrázkov na trénovanie a 10 000 obrázkov na testovanie. Každý obrázok je v odtieňoch sivej, má rozmer 28 x 28 pixelov a reprezentuje číslicu od 0 do 9.

**Zadanie:**

Vytvoril som doprednú neurónovú sieť (viacvrstvový perceptrón), ktorú som natrénoval pomocou troch optimalizačných algoritmov: SGD, SGD s momentom a Adam. Počas tréningu som sledoval trénovaciu a testovaciu chybu, ako aj presnosť modelu na testovacej množine (koľkokrát model správne predpovedal triedu). Cieľom bolo dosiahnuť presnosť vyššiu ako 97 %. Na implementáciu som použil knižnicu PyTorch.

**Postup:**

1. **Načítanie datasetu:** Dataset MNIST som načítal priamo z PyTorch knižnice, kde je voľne dostupný. Dáta som rozdelil na trénovaciu a testovaciu množinu.
2. **Predspracovanie dát:** Hodnoty pixelov som normalizoval do intervalu od -1 do 1. Cieľové hodnoty (čísla od 0 do 9) som použil priamo, bez potreby one-hot kódovania, pretože som využíval CrossEntropyLoss, ktorá ho nahrádza.
3. **Finalne navrhovanie modelu a hyperparametrov:**
   * Model má tri vrstvy:
     + Prvá skrytá vrstva obsahuje 256 neurónov a používa aktivačnú funkciu ReLU.
     + Druhá vrstva má 128 neurónov, tiež s ReLU.
     + Výstupná vrstva má 10 neurónov (pre každú triedu).
   * Hyperparametre:
     + Rýchlosť učenia: 0,01 pre SGD a SGD s momentom, 0,0005 pre Adam.
     + Batch: 256.
     + Počet epoch: 10 pre všetky modely.
4. **Tréning modelu:** Každý model som natrénoval samostatne rovnaký počet epoch, aby som mohol porovnať ich výkon. Výsledky som sledoval vo forme zmien chyby a presnosti na trénovacej aj testovacej množine počas tréningu.
5. **Porovnanie optimalizačných algoritmov:** Na základe trénovacej a testovacej presnosti som určil, ktorý optimalizačný algoritmus je najlepší.

**Proces učenia:**

**údaje:**Používa sa súbor údajov MNIST načítaný cez PyTorch.

Obrázky sú normalizované na rozsah[−1,1] na urýchlenie učenia.

**Batch size** : 256 .

**Stratová funkcia:**

CrossEntropyLoss, vhodná pre viactriedne klasifikačné problémy.

**Optimalizačné algoritmy:**

SGD,SGD s momentom: Vylepšený SGD, ktorý zohľadňuje „zotrvačnosť“ gradientu, Adam: Adaptívna metóda s úpravou rýchlosti učenia na základe historických údajov.

**vzdelanie:**

Každý model je trénovaný na 10 epoch.

V každej epoche sa počítajú tréningové a testovacie metriky: chyba a presnosť.

**Hodnotenie výkonu:**

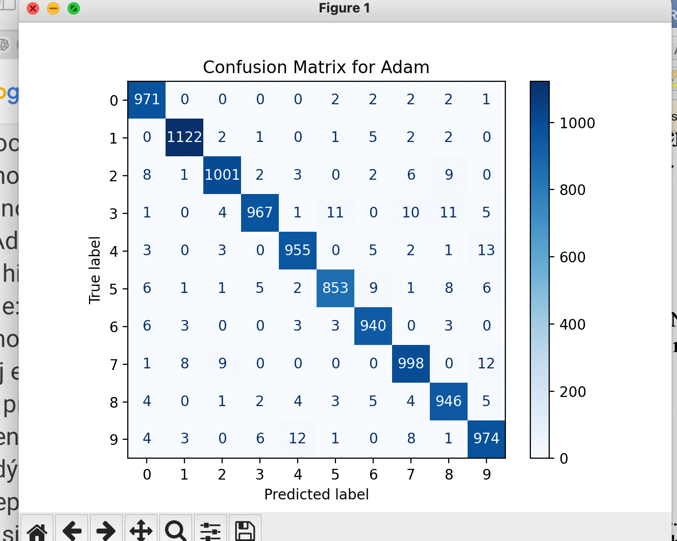
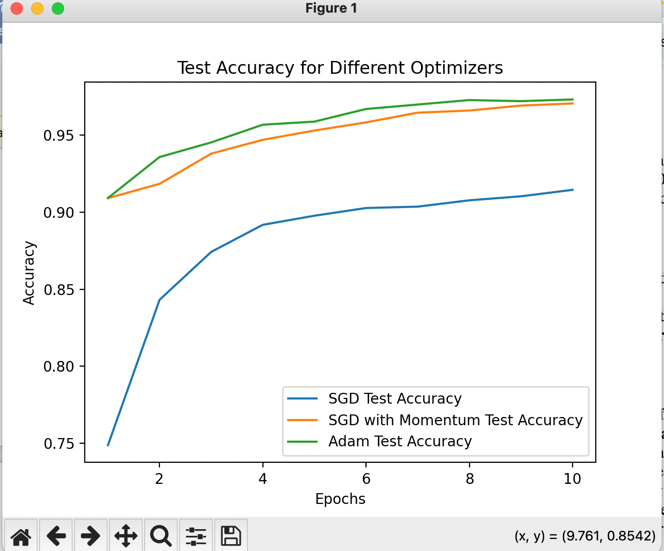
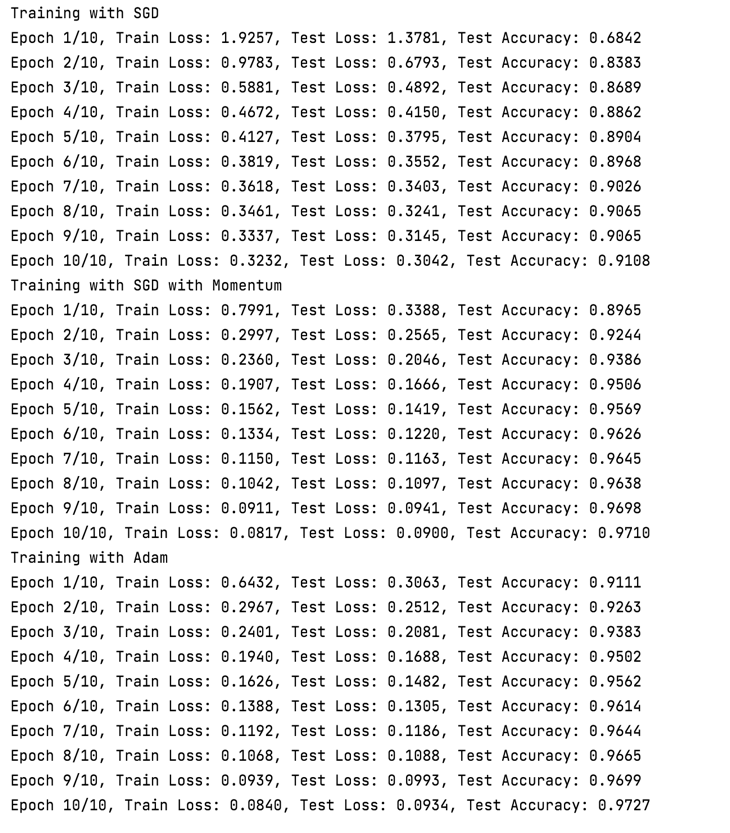
Pre každý model sú vykreslené grafy zmien chýb a presnosti.

Pre najlepší model je vytvorená matica zmätku na analýzu chýb klasifikácie.

**Tabuľka hyperparametrov:**

| Hyperparametr | Význam hyperparametra |
| --- | --- |
| batch | 128 |
| Počet epoch | 10 |
| Miera učenia (SGD) | 0,01 |
| Miera učenia (Adam) | 0,0005 |
| Počet vrstiev | 3 |
| Aktivácia | ReLU |

**Vysledok:**



Toto sú optimálne parametre, ktoré dávajú najlepšiu pravdepodobnosť Nižšie budem demonštrovať zmeny pravdepodobnosti pri zmene týchto hyperparametrov, ale zatiaľ budeme analyzovať výsledok práce s nimi.

Algoritmus SGD funguje najhoršie zo všetkých, poskytuje presnosť iba 91 %, zatiaľ čo ostatné dva algoritmy konzistentne pracujú s presnosťou nad 97 %, hoci zvyšné algoritmy poskytujú plus-mínus rovnakú presnosť, ale najlepší je stále Adam algoritmus. , aj keď nie výrazne lepšie. Na základe výsledkov konfúznej matice z Adam algoritmu môžeme povedať, že algoritmus sa najlepšie naučí uhádnuť číslo 1 a najhoršie zo všetkých 5.

**testovanie s rôznymi parametrami:**

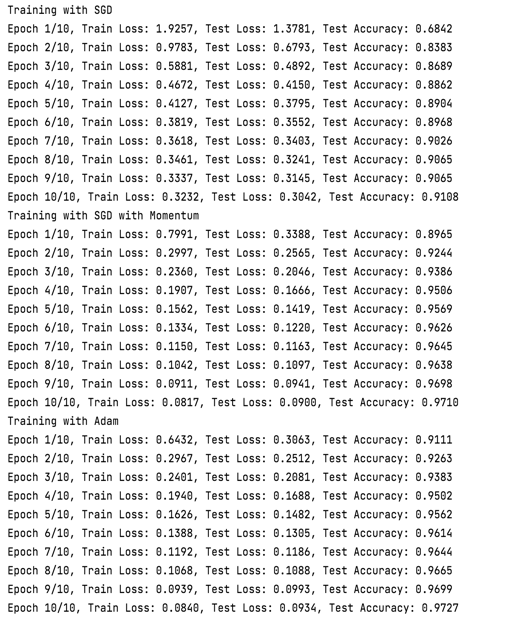
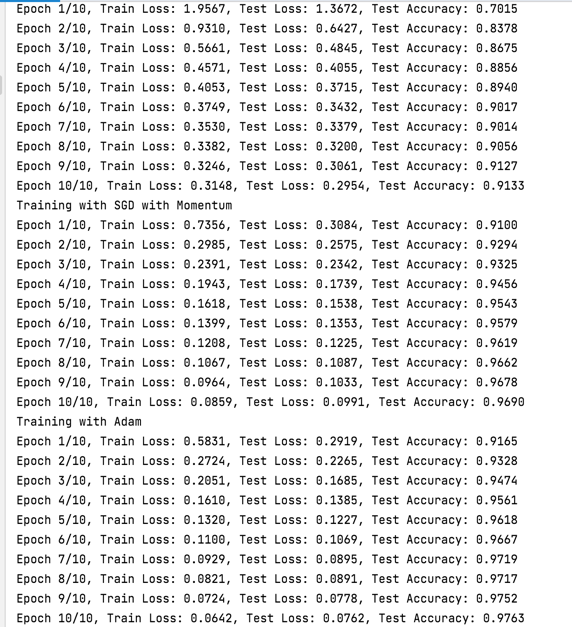
**počet neurónov na vrstve 2**:

nemá zvlášť silný účinok, ale pri 128 adam dáva lepšie výsledky ako pri 64.

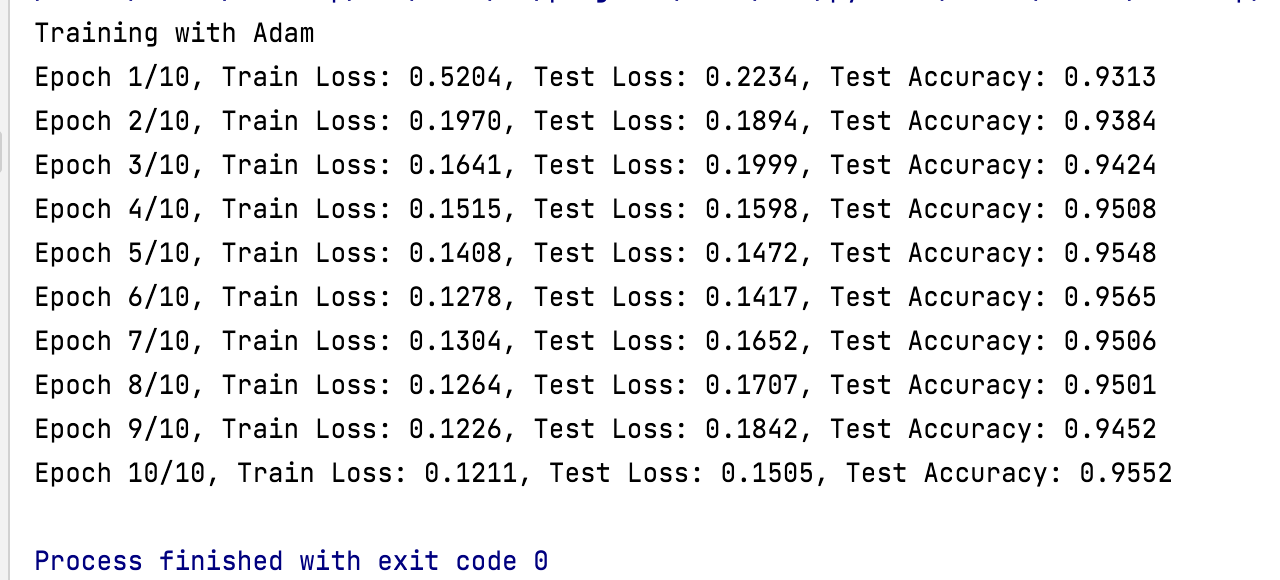
Menší počet neurónov nemusí stačiť na presnú klasifikáciu.

Príliš veľa neurónov zvyšuje čas tréningu a riziko preťaženia.

128 64



Miera ucenia Adam:

Pre adam musíte použiť menšie hodnoty, pretože ak sú príliš veľké, algoritmus prestane správne fungovať, ak s rýchlosťou učenia = 0.0005 algoritmus dáva účinnosť 0.974, potom s mierou učenia = 0.01 iba 0.9552;

Miera ucenia SGD:

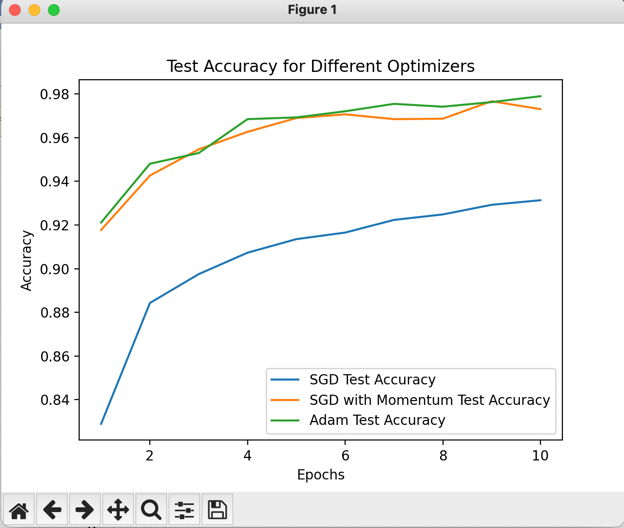
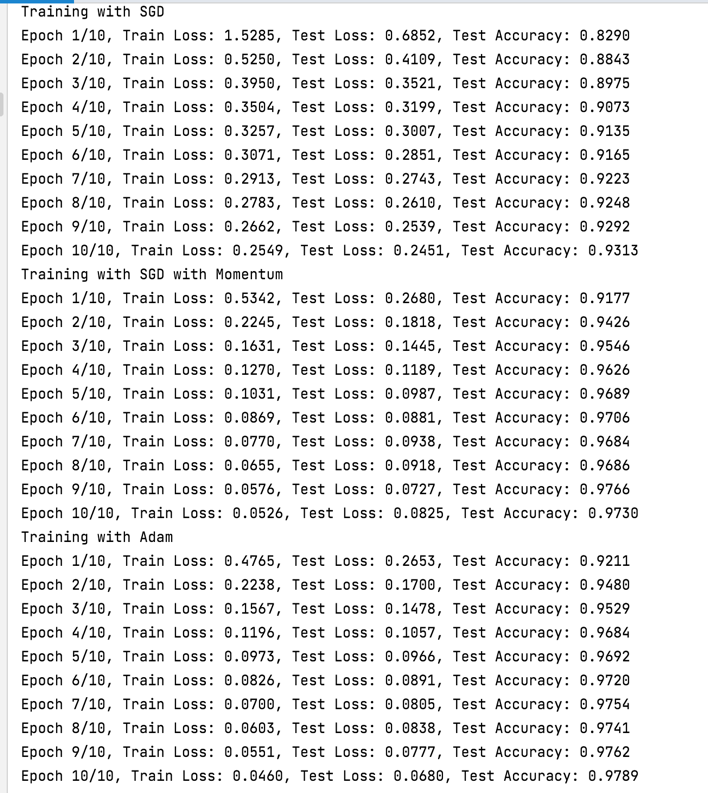
Pre SGD je to naopak, optimálna hodnota je 0,01 pri nižších hodnotách začína algoritmus fungovať oveľa horšie

Ale velke hodnoty môžu spôsobiť skokové skoky gradientov, čím sa zabráni konvergencii.

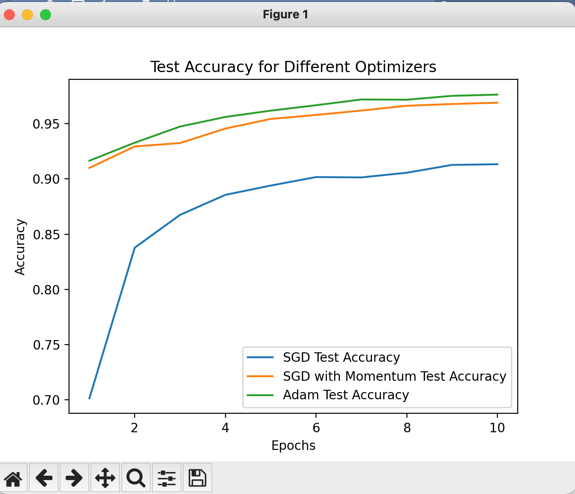
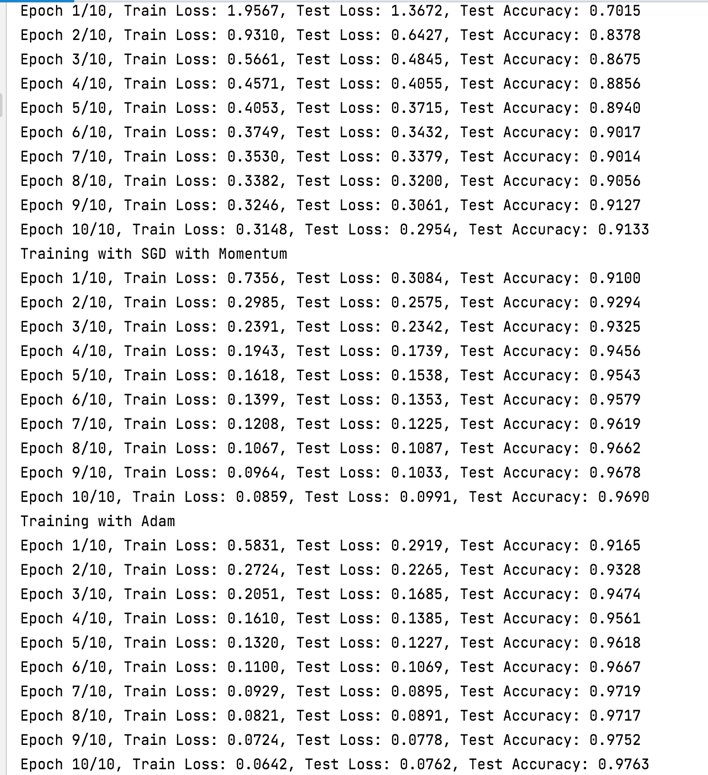
Batch size:

Optimálna veľkosť batch 128, poskytuje najpresnejší výsledok, ale mierne zvyšuje čas tréningu. Čím väčšia veľkosť, tým nižšia presnosť

Batch size 128:



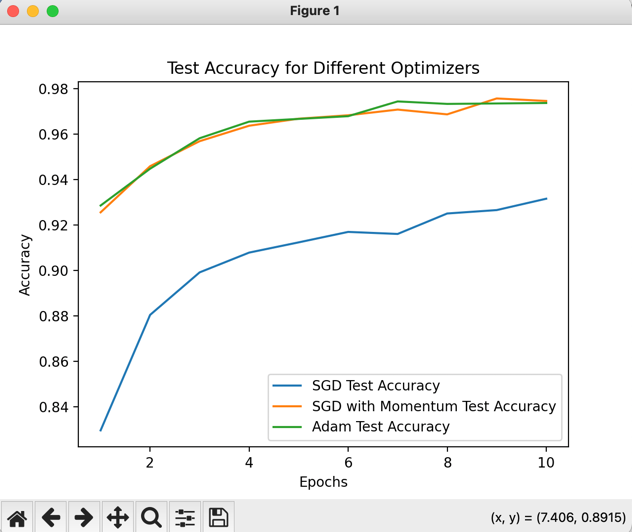
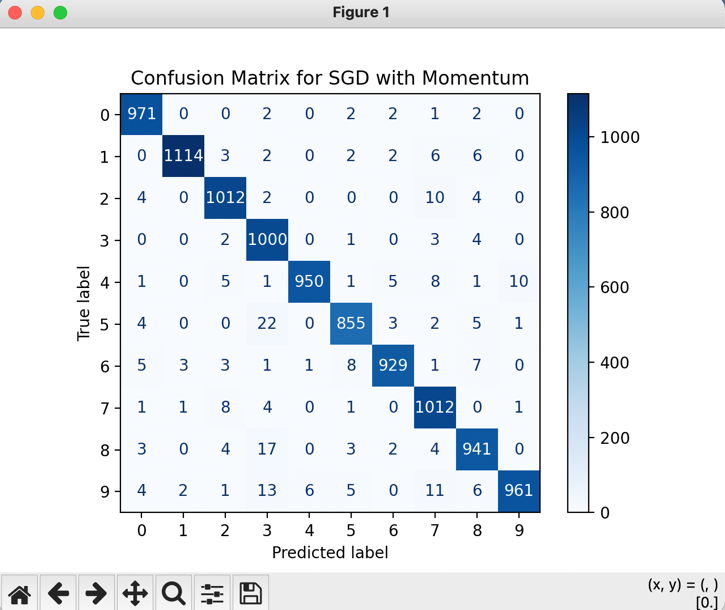
Batch size:256

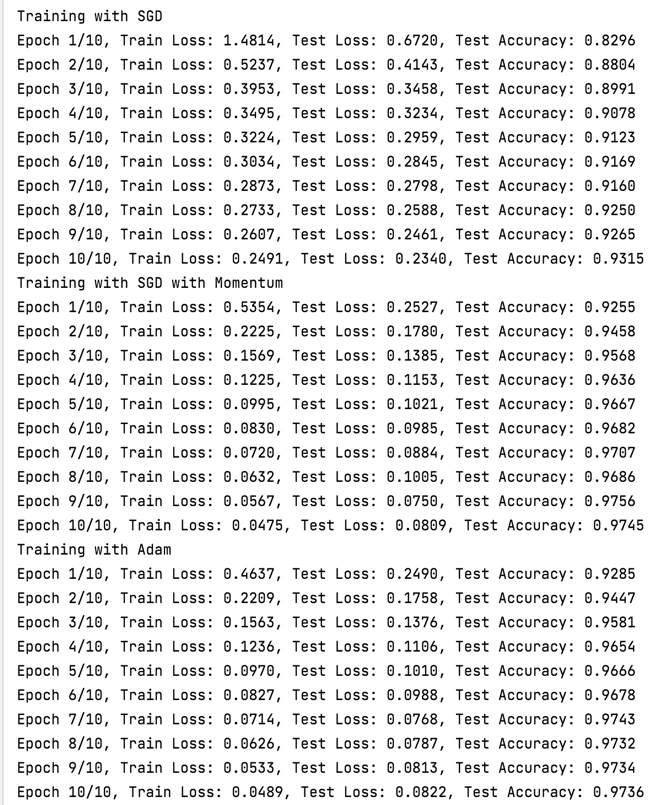


**počet epoch:**

10 epoch postačovalo na dosiahnutie dobrej presnosti (nad 97 %).

Príklad fungovania programu:





**Zaver:**

Počas úlohy bola vyvinutá a testovaná neurónová sieť na klasifikáciu ručne písaných číslic z dátového súboru MNIST. Použitá architektúra pozostávajúca zo vstupnej vrstvy so 784 neurónmi, dvoch skrytých vrstiev s 256 a 128 neurónmi a výstupnej vrstvy s 10 neurónmi preukázala vysokú účinnosť. Na aktiváciu bola použitá funkcia ReLU, ktorá zabezpečuje rýchlu a stabilnú konvergenciu.

Experimentovali sme s tromi optimalizačnými algoritmami (SGD, SGD s krútiacim momentom a Adam) a hodnotili sme ich vplyv na presnosť modelu. Algoritmus Adam ukázal najlepšie výsledky, poskytuje rýchlu konvergenciu a vysokú presnosť klasifikácie (viac ako 97 %) s optimálnou hodnotou rýchlosti učenia lr=0,0005. SGD s krútiacim momentom tiež vykazovalo dobré výsledky, ale konvergovalo o niečo pomalšie. Jednoduché SGD vykazovalo očakávane nižšiu účinnosť.

Vyvinutý model sa úspešne vyrovnal s úlohou klasifikácie ručne písaných číslic. Použité algoritmy a architektúra preukázali svoju účinnosť a optimálne hyperparametre zabezpečili presnú klasifikáciu.

Vo všeobecnosti boli všetky úlohy projektu úspešne dokončené; jediné zlepšenie, ktoré môžem odporučiť, je pohrať sa s rôznymi parametrami