Umelá inteligencia:  
Neurónové siete

**Marián Kurčina**

**ID: 127211**

[**xkurcinam@stuba.sk**](mailto:xkurcinam@stuba.sk)

**Cvičenie: Streda 15.00 - 17.00**

Obsah

[MNIST klasifikátor 3](#_Toc183877550)

[Zadanie 3](#_Toc183877551)

[Opis riešenia 3](#_Toc183877552)

[Opis kódu 3](#_Toc183877553)

[Hľadanie optimálnych hyperparametrov modelu 3](#_Toc183877554)

[Architektúra optimálneho modelu 4](#_Toc183877555)

[Optimalizácia parametrov pre trénovacie algoritmy 4](#_Toc183877556)

[Záver 4](#_Toc183877557)

[Backpropagation algoritmus 4](#_Toc183877558)

[Zadanie 4](#_Toc183877559)

[Opis riešenia 4](#_Toc183877560)

[Opis kódu 4](#_Toc183877561)

[Parametre objektov 4](#_Toc183877562)

[Forward metóda 4](#_Toc183877563)

[Backward metóda 4](#_Toc183877564)

[Hľadanie optimálnych hyperparametrov modelu 4](#_Toc183877565)

[Architektúra optimálneho modelu 4](#_Toc183877566)

[Záver 5](#_Toc183877567)

# MNIST klasifikátor

## Zadanie

Úlohou je vytvoriť neurónovú site na klasifikáciu ručne písaných číslic z datasetu MNIST. Na riešenie použite doprednú neurónovú site (viacvrstvový perceptrón) a natrénujte ju pomocou algoritmov SGD, SGD s momentum a ADAM. Okrem trénovacej a testovacej chyby odmerajte aj presnosť modelu. Model by mal mať výslednú presnosť viac ako 97%. V úlohe použite knižnicu PyTorch.

## Opis riešenia

Najprv si nahrám data do programu a transformujem ich aby mali hodnotu 0-1. Nasledovne ich rozdelím na dve časti, jedna časť bude trénovacia a jedna testovacia. Po získaní a upravení datasetu budem model trénovať a nasledovne testovať postupne pre každý algoritmus.  
Trénovanie bude prebiehať niekoľko epoch, v každej epoche po banchoch vpustím obrázky číslic do modelu a získam predikciu. Z rozdielu predikcie a skutočného označenia číslice zistím gradiant, ktorý bude slúžiť na pretrénovanie siete a teda zmenu lineárnych vrstiev modelu. Spôsob menenia lineárnych vrstiev bude závisieť od algoritmu trénovania.   
Po každej epoche vypíšem priemernú odchylku počas trénovania a model otestujem. Informácie o teste si uložím a pokračujem nasledujúcou epochou. Po vykonaní predom určenom počte epoch model posledný krát otestujem, toto je môj výsledný model. Nakoniec vykreslím grafy úspešnosti a trénovacích a testovacích chýb.

## Opis kódu

Na začiatku mám načítanie transformovaného datasetu do train\_dataset a test\_dataset, prvá časť slúži na trénovanie modelu, druhá na jeho testovanie.  
Ďalej si rozdelím dataset do batchov, ktoré budú v train\_loaderi a test\_loaderi, tieto loadery budú slúžiť na testovanie a trénovanie modelu.  
Nasledovne začne proces trénovania a testovania, najprv testujem pomocou SGD – potrebujem parameter learning\_rate, potom pomocou SGD s momentum - potrebujem learning\_rate a momentum a ako posledný využijem ADAM – potrebujem parameter learning rate.  
Samotné trénovanie začne vytvorením si modelu, ktorého vrstvy mám zadefinované v kóde.  
Pre každú epochu najprv resetnem gradianty, zistím output modelu, loss a s lossom vykonám backward funkciu po ktorej môžem zmeniť váhy vineárnych vrstiev pomocou daného optimizeru (SGD/SGD\_momentum/ADAM). Output modelu bude vždy uvedený ako pravdepodobnosť, že ide o dané číslo. Pre každú epochu si program uloží trénovaciu chybu a po trénovaní so všetkými obrázkami datasetu otestuje model pomocou datasetu test a uloží si úspešnosť správnej predikcie a taktiež testovaciu chybu. Tieto informácie na konci zobrazí v grafe a presnosť modelu bližšie ukáže cez confusion maticu.

## Hľadanie optimálnych hyperparametrov modelu

### Architektúra optimálneho modelu

## Optimalizácia parametrov pre trénovacie algoritmy

## Záver

Podarilo sa mi naučiť sa pracovať s knižnicou PyTorch a vytvoriť program na klasifikáciu číslic z datasetu MNIST za pomoci neurónovej siete. Podarilo sa mi identifikovať optimálnu architektúru modelu a optimálne parametere pre trénovacie algoritmy algoritmy SGD, SGD s momentum a ADAM.

# Backpropagation algoritmus

## Zadanie

V tejto úlohe je potrebné implementovať plne funkčný algoritmus backpropagation, treba implementovať doprednú aj spätnú časť pre operatory a funkcie, ako aj aktualizácie parametrov siete. Algoritmus overíte pomocou natrénovania doprednej neurónovej siete. Na riešenie úlohy použite knižnicu NumPy, použitie knižníc PyTorch a TensorFlow je zakázané. Implementujte modulárnu architektúru, v ktorej bude možné jednotlivé moduly reťaziť.Implementácia musí obsahovať lineárnu vrstvu, aplikačné funkcie sigmoid, tanh a relu a chybovú funkciu MSE. Pre validáciu použite problémy AND, OR a XOR, vyskúšajte siete s 1 a 2 skrytými vrstvami.

## Opis riešenia

## Opis kódu

### Parametre objektov

### Forward metóda

### Backward metóda

## Hľadanie optimálnych hyperparametrov modelu

### Architektúra optimálneho modelu

## Záver