ZÁPADOČESKÁ
UNIVERZITA
V PLZNI

Semestrální práce z KIV/UIR

Klasifikace ručně psaných číslic

Xuan Toan Dinh

A19B0027P

dinhgos@students.zcu.cz

16.05.2021

Obsah

1. Zadání

Ve zvoleném programovacím jazyce navrhněte a implementujte program, který umožní klasifikovat ručně psané číslice 0 – 9.

- Datová sada
  - je dostupná ke stažení na http://home.zcu.cz/~pkral/uir/sp/mnist_png.tar.gz. Jedná se o standardní dataset MNIST ve formátu png.
- Implementujte alespoň dva různé algoritmy (z přednášek i vlastní) pro tvorbu příznaků reprezentující číslice.
- Implementujte alespoň dva různé klasifikační algoritmy (vlastní implementace, klasifikace bude s učitelem, např. klasifikátor s min. vzdáleností).
- Funkčnost programu bude následující:
  - Spuštění s parametry: trénovací_množina, testovací_množina, parametrizační_algoritmus, klasifikační_algoritmus, název_modelu program natrénuje klasifikátor na dané trénovací množině, použije zadány parametrizační/klasifikační algoritmus, zároveň vyhodnotí úspěšnost klasifikace a natrénovaný model uloží do souboru pro pozdější použití (např. s GUI).
  - Spuštění s jedním parametrem = název_modelu: program se spustí s jednoduchým GUI a uloženým klasifikačním modelem. Program umožní klasifikovat cifry napsané v GUI pomocí myši.
- Ohodnoťte kvalitu klasifikátoru na vytvořených datech, použijte metriku přesnost(accuracy). Otestujte všechny konfigurace klasifikátorů (tedy celkem 4 výsledky).
- Pro implementaci parametrizačních / klasifikačních algoritmů není možné používat hotové knihovní funkce!

2. Analýza úlohy

a. Základ

Cílem semestrální práce z předmětu umělá inteligence a rozpoznávání je vytvořit program, který umožní klasifikovat ručně psané číslice 0 – 9. Dále je potřeba, aby se z napsané cifry udělaly nějaké příznaky, které by se pak mohli zpracovat pomocí klasifikačního algoritmu.

b. Načítání dat

Program musí být schopen načíst dataset, který obsahuje obrázky o velikosti 28x28. Jednotlivé obrázky nejsou nejlépe pojmenovány, tudíž musím udělat metodu na získání všech souborů ve složce. Dále si taky musím uložit jméno cesty k souboru, jelikož její název taky odpovídá její třídě.

Obrázky jsou šedo tónový a o velikosti 28x28. Pro získání příznaků dobrý si načíst obrázek jako bitmapu, kde jednotlivé pixely jsou hodnoty od 0 do 255.

c. Příznaky

Jako příznaky pro zpracování cifer jsem si vybral počet bílých pixelů ve celém obrázku a histogram řádků. Z obrázku tedy vyjde vektor o velikosti 1 pro bílý pixely a pro histogram vyjde vektor s 28 prvky. Jelikož se ale jedná o učení s učitelem, tak nám nestačí jenom vektory, ale taky potřebujeme jejich třídy. Pro tento účel by bylo vhodné si vytvořit nějakou datovou strukturu pro uchování příznaků a její třídy.

d. Klasifikační algoritmy

Po získání příznaků z obrázků, se program muže natrénovat. Zvolil jsem si metodu k-nejbližších sousedů a minimální vzdálenosti. Pro metodu k-nejbližších sousedů je zapotřebí uložit si všechny příznaky a u metody minimální vzdálenosti stačilo si uložit těžiště jednotlivých tříd. Pro klasifikaci k-nejbližších sousedů jsem se rozhodl použít k=3. Pro funkčnost algoritmů si musíme taky implementovat metodu pro získání vzdálenosti mezi danými vektory. Vzdálenost jsem se rozhodl počítat pomocí Eukleidovské vzdálenosti.

e. Ukládání modelu

Poté co se program natrénuje na datech si musíme tento model někam uložit pro pozdější využití, jelikož načítání obrázků a trénování programu trvá nejdéle. Modely tedy ukládám do textového souboru. Textový soubor bude mít hlavičku, kde budou informace o tom, jaký příznak jsem použil a jaká metoda pro klasifikaci byla použita. Zbytek souboru bude naplněn daty pro klasifikaci.

f. Ohodnocení modelu

Po vytvoření modu bylo zapotřebí ohodnotit kvalitu klasifikátoru. Pro ohodnocení kvality klasifikátoru jsem použil metriku přesnosti. Toto jsem testoval na všechny konfigurace klasifikátoru.

g. GUI

Aplikace musí obsahovat jednoduchý GUI, do kterého muže uživatel kreslit pomocí myši. Jelikož jsou všechny trénovací data o velikost 28x28, tak i nakreslený obrázek musí mít stejnou velikost. Naprogramovat plátno o velikost 28x28 by nebylo vhodné pro uživatele. Proto jsem vytvořil metodu pro zmenšení obrázku na 28x28. Okno taky potřebuje tlačítka na ovládání. Nejdůležitější bude tlačítko pro klasifikaci obrázku a tlačítko pro smazání plátna. Někam do GUI by se měla vypsat výsledek klasifikace.

3. Implementace

a. Vstup

Jako první jsem si musel udělat metodu pro nalezení cesty k obrázkům. Cesty jsem si pak uložil jako String do Arraylist. Dále jsem si udělal metodu pro načítání dat. Metoda pro načítání dat bude spočívat v tom, že se pro všechny cesty v ArrayList načte obrázek jako BufferedImage. Teď můžeme získat příznaky z obrázků. Jako příznaky jsem si vybral počet bílých pixelů a histogram řádků.

b. Zpracování obrázku

Udělal jsem si forcyklus, který prochází jeden řádek obrázku. V tomto focyklu si zjistím, jestli vybraný pixel je černý nebo ne. Barvu pixelu si vyhodnotím tak, že si zjistím jeho rbg složky. Černá barva má rgb složky rovno 0, všechny ostatní budou bílé. Dále sečtu počet bílých pixelů.

Mám tedy funkci pro nalezení počtu bílých pixelů v jedné řádce a stačí nám tedy projít všechny řádky. Jednotlivé hodnoty si uložím do pole. Tento vektor je histogram řádků pro počet bílých pixelů stačí sečíst všechny hodnoty v poli.

c. Trénování

Z obrázku tedy vytvořím vektor, který použiji později pro klasifikaci. Pro trénování jsem si vytvořil datovou strukturu, která se skládá z názvu třídy, do které třídy patří a její vektor. Dále jsem si udělal ArrayList pro naši datovou strukturu. ArrayList naplním daty, a podle klasifikační metody si data zpracuji a uložím do textového souboru.

d. Klasifikace

i. Minimální vzdálenost

Vytvořil jsem si metodu pro klasifikování pomocí minimální vzdálenosti. Nejprve jsem si musel spočítat těžiště jednotlivých tříd. Těžiště jsem si vypočítal pomocí následujícího vzorečku.

$$x_T = \frac{x_1 m_1 + x_2 m_2}{m} = \frac{\sum m_i x_i}{m}$$

Obrázek 1 Těžiště

Jednotlivé těžiště jsem si pak uložil do pole.

ii. Algoritmus k-nejbližších sousedů(knn)

Vytvořil jsem si metodu pro klasifikování pomocí knn. Pro knn stačilo porovnat vzdálenosti mezi vstupním vektorem a natrénovanými daty. Pro knn metodu jsem si vybral k = 3, což znamená že hledám 3 nejbližší vektory ke vstupnímu vektoru.

iii. Výpočet vzdálenosti

Výpočet vzdálenosti jsem spočítal pomocí následujícího vzorečku.

$$d = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Obrázek 2 Euclidovská vzdálenost

U knn stačilo porovnat jednotlivý vzdálenosti a někam si uložit k- nejbližších. U minimální vzdálenosti stačí vybrat nejmenší vzdálenost.

e. GUI

Vytvořil jsem okno s využitím javafx knihovny. Okno obsahuje grafický kontext, do kterého může uživatel kreslit pomocí myši. Plátno se následně uloží a poté se vyhodnotí obrázek pomocí nastaveného klasifikátoru.

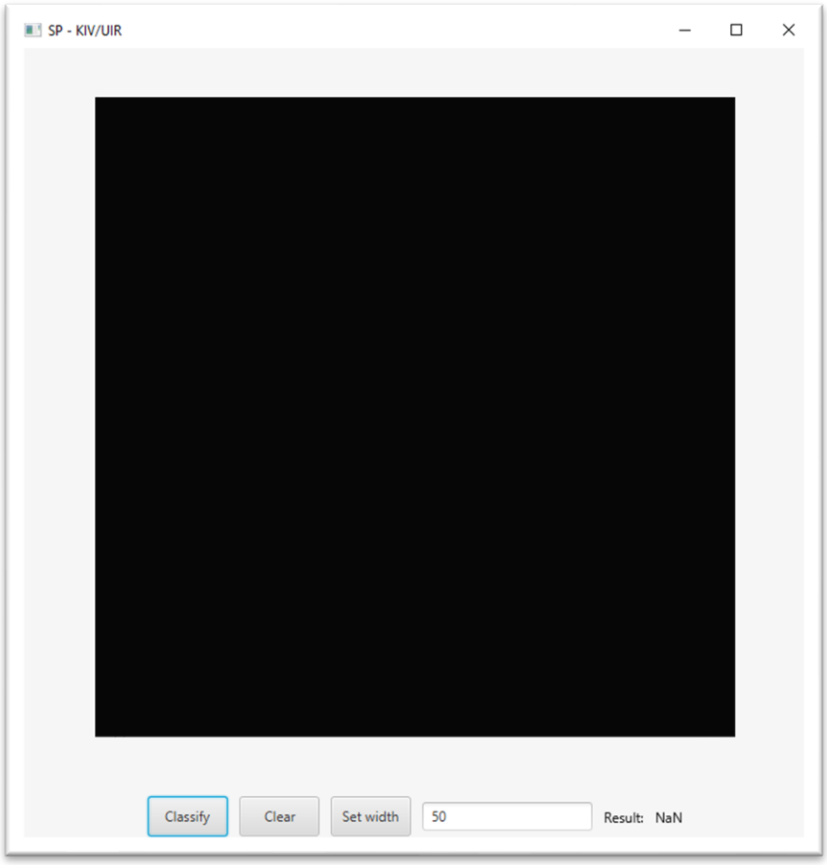
4. Uživatelská příručka

Stáhněte si projekt a někam si ho rozbalte. Poté si otevřete příkazovou řádku a otevřete si projekt. Spuštěním skriptu se vám otevře aplikace. Do skriptu je potřeba přidat parametry (např. Run.cmd mnist_png/training mnist_png/testing white_pixels min_distance model1).

Pokud se program spustí s 5 parametry, tak se vytvoří model a následně se ohodnotí kvalita modelu. Pokud se program spustí s jedním parametrem, tak se spustí okno, do kterého můžete kreslit pomocí myši. Po stisknutí tlačítka classify se vyhodnotí obrázek a vedle result se vypíše výsledek. Zavřením okna se ukončí celá aplikace.

Ovládání:

- Tlačítko “Classify“ – klasifikuje nakreslenou cifru
- Tlačítko “Clear“ – vymaže plátno
- Tlačítko “Set width“ – nastaví velikost pera na zvolenou hodnotu



Obrázek 3 GUI aplikace

Program můžete spustit s parametry:

- trénovací_množina, testovací_množina, parametrizační_algoritmus, klasifikační_algoritmus, název_modelu
    - program natrénuje klasifikátor na dané trénovací množině, použije zadány parametrizační/klasifikační algoritmus, zároveň vyhodnotí úspěšnost klasifikace a natrénovaný model uloží do souboru pro pozdější použití

- název_modelu
    - program se spustí s jednoduchým GUI a uloženým klasifikačním modelem. Program umožní klasifikovat cifry napsané v GUI pomocí myši.

Parametry:

- trénovací_množina – cesta ke složce training
- testovací_množina – cesta ke složce testing
- parametrizační_algoritmus:
    - white_pixels – počet bílých pixelů
    - histogram_row – počet bílých pixelů na řádek
- klasifikační_algoritmus:
    - min_distance – klasifikace pomocí minimální vzdálenosti
    - k-NN – klasifikace pomocí k-nejbližších sousedů
- název_modelu:
    - jakýkoliv String

Struktura adresáře:

- javafx-sdk-11.0.2 - knihovny pro spuštění aplikace
- mnist_png – trénovací a testovací data
- src – zdrojový kód
- SP.jar - spustitelná aplikace
- cases – nápověda pro vytvoření všech modelů
- run – skript pro spuštění aplikace

5. Závěr

Program představuje jednoduchý klasifikátor ručně psaných číslic. Program funguje relativně dobře pro příznaky metodou histogramu (60 % – 70 %). Pro počet bílých pixelů je program nespolehlivý (18 % - 25 %).

Jedná se o jednoduchý program, který v praxi nemá reálný využití, jelikož přesnost tohoto programu je příliš nízká. Dále umí jenom klasifikovat obrázky o velikosti 28x28 a cifry musí být nakresleny do středu plátna, jinak algoritmus nedokáže spolehlivě cifry klasifikovat.