VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2017 Denis Ďuriš

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

ROZPOZNÁVÁNÍ RUČNĚ PSANÉHO TEXTU S VYUŽITÍM POSUVNÉHO OKNA
HANDWRITTEN TEXT RECOGNITION USING A SLIDING WINDOW

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Denis Ďuriš**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Martin Rajnoha**
SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Denis Ďuriš ***ID:*** 173641
Ročník: 3 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Rozpoznávání ručně psaného textu s využitím posuvného okna

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte současné možnosti rozpoznávání textu, zaměřte se především na ručně psaný text. Vytvořte program, umožňující procházet naskenovaný text posuvným oknem a rozpoznávat znaky. Z dosažených výsledků vyjádřete přesnost rozpoznávání.

DOPORUČENÁ LITERATURA:

[1] Gyeonghwan K., Venu G, Sargur N. S. An architecture for handwritten text recognition systems, IJDAR (1999) 2: 37–44

[2] Espana-Boquera S., et. al. Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models, IEEE Transactions on pattern analysis and machine intelligence, vol. 33, no. 4, April 2011

Termín zadání: 1.2.2017 ***Termín odevzdání:*** 8.6.2017

Vedoucí práce: Ing. Martin Rajnoha
Konzultant:

doc. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Táto bakalárska práca sa zaoberá optickým rozpoznávaním znakov. Zameriava sa na rozpoznávanie ručne písaného textu. Teoretický úvod popisuje metódy používané pri optickom rozpoznávaní znakov a vybrané metódy strojového učenia. Následne sú v práci popísané dve metódy na tvorbu výrezov znakov, využívajúce posuvné okno. Výrezy sú použité v datasetoch, určených na trénovanie a testovanie modelov strojového učenia. Práca ďalej obsahuje metódy na zlepšenie presnosti rozpoznávania znakov. V kapitole Výsledky sú zhodnotené merania presnosti modelov. V práci je vytvorený automatizovaný program na rozpoznávanie znakov, ktorý klasifikuje výrezy znakov.

KĽÚČOVÉ SLOVÁ

OCR, optické rozpoznávanie znakov, HWR, rozpoznávanie rukou písaného textu, posuvné okno, ANN, umelá neurónová sieť, k-NN, k-najbližších susedov, MLP, viac vrstvový perceptron, SVM, metóda podporných vektorov, výrezy znakov, strojové učenie

ABSTRACT

This bachelor thesis deals with optical character recognition. It focuses on recognizing hand-written text. The theoretical introduction describes the methods used for optical character recognition and selected machine learning methods. Subsequently, the work describes two methods for making cutouts of characters, using a sliding window. Cutouts are used in training and testing datasets of machine learning models. The document includes methods to improve the accuracy of character recognition. The accuracy of the models is evaluated in conclusion. Charcters in cutouts are clasified by an automated recognition program.

KEYWORDS

OCR, optical character recognition, HWR, handwriting recognition, sliding window, ANN, artificial neural network, k-NN, k-nearest neighbors, MLP, multi layer perceptron, SVM, support vector machine, cutouts of charcters, machine learning

ĎURIŠ, Denis *Rozpoznávání ručně psaného textu s využitím posuvného okna*: bakalárska práca. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2016. 49 s. Vedúci práce bol Ing. Martin Rajnoha

VYHLÁSENIE

Vyhlasujem, že som svoju bakalársku prácu na tému „Rozpoznávání ručně psaného textu s využitím posuvného okna“ vypracoval(a) samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor(ka) uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil(a) autorské práva tretích osôb, najmä som nezasiahol(-la) nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý(-á) následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

POĎAKOVANIE

Rád by som poďakoval vedúcemu bakalárskej práce pánovi Ing. Martinovi Rajnohovi, za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Brno ....

podpis autora(-ky)

OBSAH

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ÚVOD

Optické rozpoznávanie rukou písaného textu je proces, ktorý má za úlohu prevod dokumentu do digitálnej podoby, ktorá umožňuje jeho následnú úpravu. Tento prevod sa nazýva aj digitalizácia. Takto je možné zdieľať dokumenty po sieti, napríklad medzi pobočkami firiem. Zamestnancom je vďaka tomuto procesu uľahčené aj vyhľadávanie záznamov, keďže v digitalizovanom dokumente je možné vyhľadávať napríklad kľúčové slová.

Prvá časť práce obsahuje teoretický úvod do optického rozpoznávania znakov a zahŕňa opisy všeobecne používaných metód, a procesov. Nasledujú opisy vybraných modelov strojového učenia a niekoľkých aktuálnych prístupov k danej problematike.

Ďalej je spomenutý softvér použitý v práci. Jedná sa o softvér Tesseract, ktorý vyhľadáva textové bloky a o program RapidMiner, využitý na trénovanie modelov strojového učenia.

Neskôr sa práca zameriava na rozpoznávanie ručne písaného textu, pri ktorom sa systém nemôže spoliehať na medzery medzi riadkami, slovami a písmenami, ako pri tlačenom texte. Systém rozpoznávajúci ručne písané dokumenty musí byť schopný sa adaptovať na rôzne typy písma a prípady, kedy sa znaky navzájom prekrývajú alebo dotýkajú. V opačnom prípade dochádza k chybnej segmentácií znakov, čo má za následok ich nesprávnu klasifikáciu.

Táto práca preto obsahuje metódy segmentácie znakov, ktoré riešia spomenuté problémy. Dve vybrané metódy sú implementované pomocou programovacieho jazyka Java. Nosným prvkom metód je posuvné okno, ktoré vytvára výrezy spracovávaného dokumentu. Tento spôsob umožňuje vyhnúť sa chybám pri segmentácií znakov, ktoré nie sú dostatočne odlíšené od ostatných a spájajú sa so susedným znakom. Jedno z riešení čiastočne používa open source softvér Tesseract, ktorý je schopný vyhľadávať riadky. Druhá metóda je založená na vyhľadávaní farebných rozdielov predstavujúcich hranice medzi znakmi a podkladom.

Ďalej je v práci návrh metód zlepšujúcich presnosť rozpoznávania znakov, ktoré využívajú úpravy trénovacích dát metód strojového učenia a návrh automatizovaného programu na rozpoznávanie znakov.

Posledná časť práce je venovaná výsledkom a záveru. V tejto časti sú zhodnotené možnosti posuvného okna a jeho vlastností v jazyku Java, spolu s návrhmi metód pre zlepšenie presnosti rozpoznávania. Nasleduje zhodnotenie výsledkov metód posuvného okna a meranie presnosti vybraných modelov strojového učenia, za použitia rôznych úprav ich trénovacích dát.

1 TEORETICKÝ ÚVOD

Táto práca sa zaoberá postupmi a metódami používanými na prevod textu do digitálnej podoby. Táto problematika sa označuje skratkou OCR (z angl. Optical Character Recognition – optické rozpoznávanie znakov). Je súčasťou digitálneho spracovania textov a je možné sa s týmto procesmi stretnúť na rôznych miestach. Ide o prevod informácií, väčšinou textu alebo rôznych číselných údajov, z papierovej, do digitálnej formy, čo umožňuje a tiež zjednodušuje ich ďalšie spracovanie. Typicky sa jedná o digitalizáciu záznamov, formulárov, dokumentov a ich následné zaraďovanie do firemných databáz, a archívov. Princíp spočíva v spracovaní naskenovaného alebo odfotografovaného dokumentu. Ten sa rozdelí na segmenty, u ktorých sa rozhoduje o znaku, ktorý obsahujú. Získané znaky sa zvyčajne spájajú naspäť do slov pomocou slovníkov, ďalej sa skladajú vety a celá štruktúra dokumentu do pôvodnej podoby. Avšak tentoraz už v digitálnom formáte, ktorý je možné ďalej upravovať a spracovávať [1].

1.1 Charakteristika OCR

Optické rozpoznávanie znakov zastupuje súhrn metód, ktoré zabezpečujú zmenu formátu textu, ktorý je tlačený, písaný na stroji alebo písaný rukou. Úloha takéhoto systému spočíva v prevode dokumentov do digitálnej podoby, ktorú je možné editovať pomocou výpočtovej techniky s príslušným softvérom, čo sa dá posúdiť aj ako strojové čítanie. Vstupom sú prevažne naskenované dokumenty, prípadne fotografie rôznych textových plôch, z ktorých je možné spomenúť napríklad dopravné značenie alebo poznávacie značky vozidiel. Rozpoznávanie strojovo tlačených znakov z naskenovaného dokumentu je jednou z najjednoduchších úloh, s ktorou sa OCR systémy zaoberajú. Problematickejšia je však detekcia textu z fotografii, ktoré nemusia mať dostatočnú kvalitu a tiež zvyčajne obsahujú bloky textu, ktoré nie sú správne zarovnané, prípadne sú inak tvarovo skreslené. Jednou z najťažších úloh je tiež rukou písaný text, pri ktorom je vysoká pravdepodobnosť výskytu atypických znakov a anomálii tvoriacich ťažké podmienky pre systémy zaoberajúce sa vyhľadávaním písma v dokumentoch tohto typu. Ako typické vlastnosti rukopisu je možné spomenúť napríklad sklon písma, premenlivú veľkosť znakov alebo rôznu veľkosť medzi znakmi, a slovami. Často nastane prípad, kedy sú znaky spolu spojené alebo do seba navzájom zasahujú, prekrývajú sa [1].

1.2 Blok prvotných úprav

Na odstránenie niektorých porúch dokumentu slúži blok prvotných úprav (angl. preprocessing). Pri čerpaní zdrojových informácií z naskenovaného tlačeného textu zvyčajne nie sú nutné geometrické korekcie, keďže riadky sú vodorovné, medzery medzi slovami a znakmi sú zreteľné, a konštantné. Fotografie je vhodné upraviť, aby rozpoznávací softvér mohol dosiahnuť spoľahlivejšie výsledky. Modifikácie sú zväčša geometrické a farebné. Ako jeden z typických zástupcov môžeme spomenúť korekciu skreslenia, teda vyrovnanie obrazu v horizontálnej a vertikálnej osi, a to podľa sklonu riadkov alebo časti predstavujúcej textový blok, ktorý snímka obsahuje. Farebné modifikácie spočívajú v izolovaní textu od pozadia, čím sa zlepší zreteľnosť znakov. Ak získavame informáciu z fotografie zhotovenej za horších svetelných podmienok, je vhodné uplatniť aj proces postavený na inteligentnom zostrený hrán znakov a eliminácii rozostrenia spôsobeného pohybom fotoaparátu. Je nutné však brať do úvahy, že tieto úpravy majú obmedzené možnosti a nedokážu vytvoriť z akejkoľvek fotografie obraz porovnateľný s naskenovaným dokumentom. Takže je vhodné zvážiť podmienky, v ktorých bude OCR systém pracovať a na ich základe vyhodnotiť nutnosť, a rozsah použitia spomenutých procesov. v oblasti rukou písaného textu je potrebné napríklad korigovať zmeny smeru a sklonu riadkov, skreslenie bloku textu aj ak sa jedná o naskenovaný vstup, keďže všetky rukopisy sa v mnohých bodoch líšia. Čo sa týka farebných regulácii, platia podobné úpravy ako pri fotografii [2].

V jednom z prístupov sa využíva neurónová sieť a skrytý markov model na posúdenie sivých odtieňov písma, a to pre každý pixel v porovnaní s jeho okolitými pixelmi. Umelé neurónové siete (ANN z angl. Artificial Neural Network) pozostávajú z vstupných neurónov, ďalej z niekoľkých skrytých vrstiev neurónov a z poslednej výstupnej neurónovej vrstvy. Každé spojenie má pridelenú váhu – číselný údaj. Takéto siete dokážu spracovať nezávislé premenné. Na výstupe sa následne vyskytujú premenné závislé na aproximačnej funkcii neurónovej siete. Vstupné aj výstupné údaje môžu byť rôznych tvarov a sústav. Jeden s prístupov využíva ANN typu MLP [1] [3].

Skrytý Markov model (HMM z angl. Hidden Markov Model) je dvojnásobný stochastický proces s podkladovým stochastickým procesom, ktorý nie je pozorovateľný, je skrytý. Na jeho pozorovanie je potrebná sada stochastických procesov, ktoré vyprodukujú sekvenciu pozorovateľných symbolov [4].

Vybraný pixel spolu s niekoľkými okolitými pixelmi predstavuje okno, ktoré vstupuje do neurónovej siete a výstupom sú zreteľnejšie pixely v znaku. Takto je možné upravovať niektoré geometrické parametre dokumentu.

[1] Viac vrstvový perceptron – Multi Layer Perceptron – typ ANN

Napríklad proces odstránenia náklonu slov. Ten využíva histogram výrezu, teda okna, každého znaku a vyhľadáva lokálne extrémy, ktoré sú následne klasifikované pomocou techník strojového učenia. Za proces je zodpovedná neurónová sieť typu MLP. Systém je založený na oddelení centrálnej časti znaku, dolného základu a horného základu. Lokalizácii týchto častí napomáhajú body lokálnych extrémov, ktoré tvoria obrysy znakov. Takto upravený obraz je jednoduchšie upraviť tak, že histogram zobrazí medzeru medzi slovami, a tak obraz rozdelí na segmenty, v ktorých sa berie do úvahy spodná základná vrstva, tvoriaca líniu sklonu slov. Náklon slov sa následne koriguje ich pootočením. Upravuje sa tiež sklon znakov. Tento proces zabezpečuje neurónová sieť typu MLP, natrénovaná na odhad toho, či je výrez obrazu, teda znak naklonený alebo nie. Pomocou posuvného vstupného okna je MLP aplikovaná na každý výrez obrazu s každým uhlom z vybraného rozsahu (napr. -50° až +50°). Dynamický algoritmus zaobstará cestu s najlepším hodnotením. Na tomto základe sa vytvorí upravená postupnosť znakov bez náklonu [5].

1.3 Segmentácia znakov

Po predošlých úpravách prichádza na rad oddeľovanie jednotlivých znakov, ktoré systém následne rozpoznáva. Segmentácia textu zvyčajne pracuje tak, že vyhľadáva v bloku textu medzery medzi písmenami. Na základe tejto informácie potom rozdeľuje jednotlivé znaky, teda vytvára ich výrezy. o tento proces sa stará príslušný blok programu vykonávajúceho rozpoznávanie. Tlačený text opäť nie je tak náročný ako rukou písaný, pri ktorom sa nevyskytujú konštantné alebo žiadne medzery. Tieto prípady potrebujú náležité ošetrenie, aby bolo možné rozpoznávať text s požadovanou presnosťou[5].

Ďalší z prístupov môže postupovať tak, že nevytvorí ihneď výrezy jednotlivých písmen, ale rozdelí riadok na mnoho okien. Tieto vzniknuté okná preverí napríklad neurónová sieť. Výstupom takejto kontroly budú s najväčšou pravdepodobnosťou nezmyselné postupnosti znakov. Tie však poslúžia ako vstupná informácia pre ďalšiu neurónovú sieť, vopred natrénovanú pomocou slovníkov, ktoré môžu byť uzatvorené alebo otvorené. Otvorený slovník sa skladá z určitého počtu najčastejšie sa vyskytujúcich slov z trénovacieho datasetu. Uzatvorený slovník tvorí daný počet slov z testovacieho textu. Tieto možnosti ovplyvňujú rýchlosť procesu, ale na druhú stranu aj kvalitu a presnosť výstupu, teda spoľahlivosť prevodu dokumentu[5].

1.4 Praktický prínos a využitie OCR

Vo všeobecnosti sa OCR využíva na prevod textu do digitálnej podoby. Tento krok umožní jednoduchú archiváciu a uľahčuje pridávanie nových informácií z formulárov do digitálnych, zdieľaných databáz. v prípade, že zamestnanec potrebuje na akejkoľvek pobočke vyhľadať napríklad klienta, stačí zadať do databázy jeho meno a nie je nutné prehľadávať záznamy manuálne, čím systémy OCR šetria čas [1].

Optické rozpoznávanie znakov má mnoho praktických využití. Vďaka tomu, že prevádza text do podoby, ktorú spracuje výpočtová technika, je táto metóda často využívaná na prevádzanie kníh, najmä starších, do digitálnej podoby. v takto vytvorených elektronických knihách je možné jednoducho vyhľadávať kľúčové slová. Tiež je možné týmto spôsobom uchovávať kópie historických knižných alebo hudobných diel [2].

Jedným z využití je aj získavanie evidenčných čísel vozidiel z fotografii vyhotovených dopravnými kamerami, prípadne kamerami policajných vozidiel, čo umožňuje ich rýchle porovnávanie s registrom vozidiel alebo napríklad so zoznamom kradnutých áut. Tento spôsob automatizácie využívajú aj spoločnosti zaoberajúce sa výberom mýta na cestách [6].

Rozpoznávanie textu je vhodné aj pre poštový sektor, kde sa z fotografií alebo snímok obstaraných pomocou skeneru získavajú adresy. Tie je možné ďalej spracovávať v databázach alebo v elektronickej komunikácii s klientmi a zamestnancami [2].

Digitalizácia firemných databáz je ďalšou z možností použitia optického rozpoznávania znakov, pričom tento proces významne zjednoduší, a aj urýchli prácu s takýmito databázami. Zadaním niekoľkých kľúčových slov je totižto možné vyhľadať požadovaný záznam. Tento proces sa využíva napríklad na digitálnu klasifikáciu faktúr, či iných početných dokumentov. To sa môže týkať aj zdravotných záznamov nemocníc a poisťovní. Bankový sektor používa rozpoznávanie znakov pri automatickom spracovávaní platieb bez zásahu človeka. Napríklad digitálny prevod šekových transakcií, či skenovanie čísel IBAN[2] [1].

Metódy, ktoré sú zamerané na rozpoznávanie ručne písaného textu, sú prezentované pod skratkou HWR (z angl. HandWriting Recognition – rozpoznávanie rukopisu). Tento proces je možné uskutočniť aktívne, to zahŕňa detekciu a porovnávanie kriviek písaných na dotykový displej, alebo grafický tablet, a to v reálnom čase. Takto môžu byť zhotovené elektronické podpisy alebo rukou písaný vstup pre zadávanie textu do mobilných telefónov, či iných zariadení. Druhou možnosťou je pasívne rozpoznávanie, ktoré neprebieha v reálnom čase. To znamená, že spísaný dokument

[2]Medzinárodný formát bankového účtu – International Bank Account Number

je najskôr naskenovaný alebo sa zhotoví jeho fotografia. Až takto vytvorený dokument je upravený a následne spracovaný pomocou programu rozpoznávanie textu. Tento systém je využívaný väčším množstvom zariadení než ten pracujúci v reálnom čase [2].

1.5 Metódy používané pri rozpoznávaní znakov

K tomuto problému existujú rôzne prístupy, podrobnejšie budú spomenuté dve metódy prístupu k rozpoznávaniu znakov. Jednou z nich je porovnávanie matíc (anglicky matrix matching) a ďalšou výber charakteristických znakov (anglicky feature extraction). Menej podrobne budú spomenuté ďalšie riešenia: štruktúrna analýza (anglicky structural analysis), nejasná, presnejšie nedefinovaná logika (anglicky fuzzy logic) a neurónové siete (anglicky neural networks) [7],[8].

1.5.1 Porovnávanie matíc

Porovnávanie matíc je tou jednoduchšou metódou ak ho porovnávame s metódou výberu charakteristických znakov. Niekde sa tiež nazýva pattern recognition, v preklade rozpoznávanie vzorov alebo image correlation, čo predstavuje koreláciu (vzájomný vzťah) obrazov. Jeho princíp spočíva v porovnávaní informácií z dvoch zdrojov. Prvým je databáza pozostávajúca z matíc, šablón a obrazov jednotlivých znakov. Druhým zdrojom sú výrezy samostatných písmen získané segmentáciou dokumentu. Tieto zdroje sa navzájom porovnávajú postupne znak po znaku na úrovni pixelov, kriviek, bodov alebo segmentov matíc. v tomto procese sa hľadá štatisticky najlepšia zhoda medzi týmito dvomi vstupmi. Výsledkom je informácia o danom znaku v digitálnej podobe, spracovateľnej zvyčajne v kóde ASCII,[3] prípadne inej forme, ktorú následne dokáže spracovať systém kompilujúci z týchto informácii slová [8].

1.5.2 Výber charakteristických znakov

Sofistikovanejším, avšak náročnejším spôsobom je už spomenutý postup výberu charakteristických znakov. Metóda je tiež známa aj ako inteligentné rozpoznávanie znakov pod skratkou ICR (z ang. Intelligent Character Recognition). v tomto prípade sa presnosť výsledku odráža od toho, do akej miery je do tohto procesu zapojené strojové učenie. v tomto prípade nie sú totižto nastavené nijaké striktné medze v podobe daných šablón, ale stroj plniaci túto úlohu vyhľadáva významné znaky (napríklad: otvorené plochy, uzatvorené tvary, rôzne vedené krivky, ich kríženia a podobné).

[3] Americký štandardizovaný kód pre výmenu informácií – z angl. American Standard Code for Information Interchange

Vďaka spomenutým vlastnostiam je systém schopný spracovať a vyhodnotiť aj menej očakávané tvary a štylizácie znakov. Táto adaptabilita vedie k spoľahlivejším a kvalitnejším výsledkom aj za zhoršených podmienok, čo sa týka vstupnej informácie (obrazu). Značne však záleží na databáze použitej pre porovnávanie s vlastnosťami znakov zo vstupného obrazu, teda na trénovacom datasete (údaje použité na trénovanie siete). Nesprávny výber výrazne znižuje presnosť rozpoznávania znakov. Zvyčajne sa táto metóda používa v kombinácii so strojovým učením neurónových sietí. Na druhú stranu, takto sofistikovaný proces nie je nutné používať, ak rozpoznávané znaky pochádzajú z malej databázy a sú dostatočne graficky oddelené. Ako príklad je možné použiť poštové služby, kde sú na dokumentoch vopred určené a dostatočne graficky vyznačené miesta na vpisovanie požadovaných informácii (typicky poštové smerové číslo). Takto zabezpečený vstup pre OCR uľahčuje prácu rozpoznávacieho systému, ktorý na vyznačených miestach očakáva číslice zo stanoveného súboru možností. z toho vyplýva úspora nákladov na zariadenia špecifikované na jednu úlohu [7], [9].

1.5.3 Štruktúrna analýza

Proces štruktúrnej analýzy skúma čiastkové vlastnosti obrazu. Znaky identifikuje pomocou ich tvarov a horizontálnych aj vertikálnych histogramov výrezov znakov. Je vhodná pre použitie na menej kvalitné vstupné snímky [8].

1.5.4 Nejasná logika

Metóda nejasnej logiky (anglicky fuzzy logic) predstavuje pokus popisovať vlastnosti viacerými úrovňami informácie a nie len logickými nulami, a jednotkami, teda priblížiť sa ľudskému logickému mysleniu v programovaní. To znamená že je to logika viacerých hodnôt, ktorými je možné opísať stav medzi formálnou logickou nulou a jednotkou. Využíva sa keď je potreba popísať neurčitý stav. Príkladom môže byť šedá, ktorá by bola bežnou binárnou logikou priradená čiernej alebo bielej, čo by mohlo negatívne ovplyvniť výsledok rozpoznávania [8].

1.5.5 Krížová validácia

Krížová validácia rozdelí dataset na niekoľko rovnakých častí a vždy použije jednu z nich na validáciu, a zvyšné časti sú použité na trénovanie metódy. Tento proces sa opakuje s tým, že každá časť je raz využitá na validáciu. Čiastkové výsledky sa spriemerujú do jednej výslednej hodnoty určujúcej presnosť metódy. Výhodou krížovej validácie je využitie všetkých prvkov datasetu na trénovanie aj validáciu [10].

1.5.6 Porovnanie metód

Metóda analýzy štruktúr (structural analysis) nie je dostatočne schopná riešiť zložitejšie situácie, a tak sa táto sekcia sústredí na porovnanie metód matrix matching a feature extraction. v poradí druhý postup, teda feature extraction je možné omnoho lepšie adaptovať na rozličné typy písma, či už v tlačenej forme alebo v podobe rukou písaného textu. Je to jedným z hlavých rozdielov oproti procesu matrix matching, keďže ten je zvyčajne limitovaný na určitú uzavretú sadu typov písma s malou schopnosťou prispôsobenia, čo obmedzuje jeho využitie. z toho vyplýva, že pre rukou písaný text je z pomedzi spomínaných dvoch metód jednoznačne vhodnejšia druhá metóda „feature extraction“, keďže za podmienky dostatočného použitia strojového učenia zaručuje výrazne lepšie výsledky v kratšom čase. Tento stav zabezpečuje práca inteligentného systému opierajúceho sa o výrazné črty, ktoré obsahujú všetky znaky, a tak je dostatočne flexibilný na to, aby sa prispôsobil rukopisu rôznych osôb, či už v tlačenej forme, alebo za pomoci grafického tabletu v reálnom čase [7],[11].

1.6 Metódy strojového učenia

Pri OCR sa využívajú rôzne metódy strojového učenia. v tejto časti je popísaných niekoľko z nich.

1.6.1 k-NN (k-Najbližších susedov)

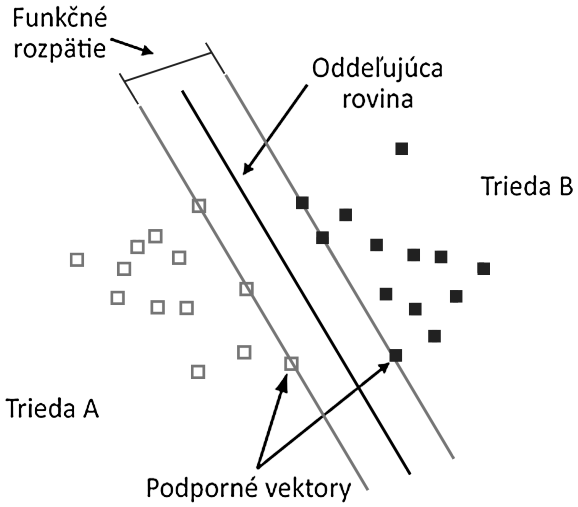
Metóda k-Najbližších susedov (z angl. k-Nearest Neighbors) je založená na porovnávaní testovacích a trénovacích dát. Metóda vyhľadáva podobnosti medzi dátami z trénovacieho datasetu a analyzovanými vzormi. Písmeno k značí počet najbližších susedov, to znamená najbližších podobných vzorov. Pri vyhľadávaní susedov sa porovnávajú vzory dát z oboch skupín. Vzory sú informácie zapísané v n-dimenzionálnom priestore, napr. vektor. Vzdialenosť medzi vzormi je popísaná metrikou, podľa ktorej sa rozhoduje o najbližších vzoroch. Metrika je definovaná napríklad Euklidovskou, Hammingovou alebo Manhattan vzdialenosťou. Je to jedna z najjednoduchších metód strojového učenia. o výsledku rozhoduje väčšinové zastúpenie jednej triedy medzi vybranými susedmi[12].

1.6.2 SVM (Metóda podporných vektorov)

Metóda podporných vektorov so skratkou SVM (z angl. Support Vector Machine), funguje tak, že vytvorí hyperrovinu alebo ich množinu v multidimenzionálnom alebo

nekonečnom priestore. Rovina je neskôr použitá na klasifikáciu alebo regresiu. Hlavným cieľom pri jej vytváraní je vytvorenie čo najväčšej vzdialenosti medzi najbližšími bodmi trénovacích dát jednotlivých tried. Táto vzdialenosť sa nazýva aj funkčné rozpätie a je znázornená na obrázku 1.1. Platí, že so zvyšovaním vzdialenosti bodov dát sa znižuje miera generalizačnej chyby klasifikátoru. Body na okraji pásma jednotlivých tried sa nazývajú podporné vektory a sú kľúčové pri popise danej triedy. Pri rozlišovaní vstupných dát, natrénovaná metóda predpovedá, do ktorej z dvoch možných tried dáta spadajú [[13],[14]].

Metóda SVM predstavuje body v priestore, namapované tak, aby dáta ňou analyzované, bolo možné rozdeliť jasne do tried oddelených čo najväčšou medzerou. Spracovaná informácia je vložená do už vytvoreného priestoru a priradí sa jej nová hodnota triedy, do ktorej spadá v priestore. v rámci udržania náročnosti výpočtov v rozumnej miere, mapovanie použité pre SVM popisuje body dát tak, aby boli jednoducho spracovateľné, čo sa týka ich premenných. Takže sú definované vybranými kernelovými funkciami jadra K(x,y). [13].



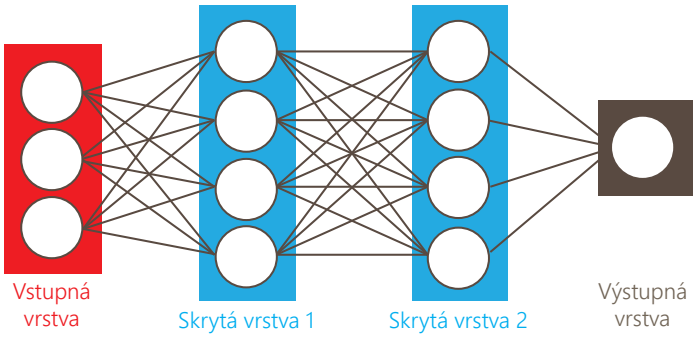
Obr. 1.1: Metóda SVM [15].

1.6.3 ANN (Umelá neurónová sieť)

Neurónové siete zvládajú množstvo rôznych úloh. Je ich možné využiť na väčšinu krokov pri rozpoznávaní textov. Pre svoju činnosť potrebujú trénovací dataset, ktorého obsah určuje ich zameranie. Od prvotných geometrických či farebných úprav, cez klasifikáciu znakov, až po skladanie slov. Umelá neurónová sieť so skratkou ANN prípadne len neurónová sieť NN (z angl. Artificial Neural Network), je metóda inšpirovaná biologickou štruktúrou a funkcionalitou neurónov. Informácie sú spracované

využitím spojovacieho prístupu k výpočtom, to znamená že stav môže byť popísaný vzájomným prepojením jednoduchých a uniformných premenných do siete. Vo väčšine prípadov sa neurónová sieť štrukturálne prispôsobuje dátam, ktoré spracováva pri trénovaní. Zvyčajne vyhľadáva komplexné spojitosti a vzory v prepojeniach informácií [16].

Rozloženie neurónovej siete spočíva v sieti prepojených neurónov. Základné neurónové vrstvy sú vstupná a výstupná. Medzi nimi je jedna alebo viac skrytých vrstiev (nie je možné priamo nahliadnuť do procesov týchto vrstiev) čo zobrazuje aj obrázok 1.2.

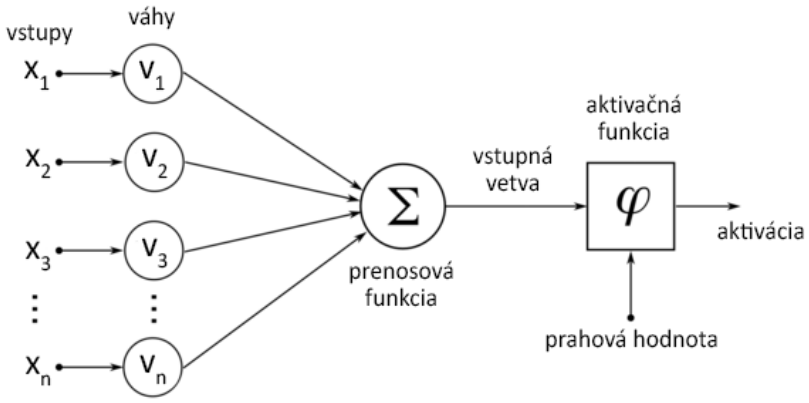


Obr. 1.2: Štruktúra neurónovej siete [17].

Vstupná vrstva slúži na vkladanie informácií do neurónovej siete. Skryté vrstvy pozostávajú z určitého množstva navzájom prepojených neurónov, a tie zastupujú rôzne prenosové funkcie, ktoré sieť vytvára pri učení. Neuróny spájajú cesty, ktorým neurónová sieť prideľuje váhy. Váhy sú vstupom prenosovej funkcie. Výsledok prenosovej funkcie je pomocou aktivačnej funkcie prehodnotený vzhľadom k prahovej hodnote, ktorá rozhoduje o aktivácii. Zloženie neurónu je popísané aj na obrázku 1.3[5],[9].

Skrytá vrstva nemusí byť len jedna, a zároveň môže obsahovať rôzne množstvo neurónov, čo zaručuje variabilitu, tým pádom je potrebné vybrať správnu konfiguráciu, ktorá najefektívnejšie splní požadovanú úlohu. Napríklad je možné uskutočniť pokusné merania presnosti pre niekoľko konfigurácií a definitívne vybrať tú najpresnejšiu. Neurónovú sieť je nutné najprv tieto vlastnosti a parametre spracovávaných údajov naučiť, takže sa jedná o strojové učenie. Na základe naučených informácií metóda posudzuje vstupné, testované údaje. Následne sieť klasifikuje vstupné dáta do výsledných tried. Pre správne fungovanie je potrebná správna sada dát určená na trénovanie siete[5],[9].

Vstupy a výstupy môžu nadobúdať rôzne tvary, avšak ak sa jedná o využitie v OCR je pravdepodobným vstupom okno predstavujúce výrez získaný z textu a to



Obr. 1.3: Štruktúra jedného neurónu [18].

v maticovom alebo vektorovom tvare. Spomenuté posuvné okno je jedna z možností, ako zabezpečiť takýto vstupný údaj. Tvary výstupov záležia na úlohe danej neurónovej siete. Môže to byť obraz, z ktorého bol odstránený šum, spoľahlivejšia hodnota rozhodujúca o častiach znaku, kde nie je možné jednoznačne určiť binárny údaj farby (pomer šedej farby, anglicky gray scale level) alebo konkrétny znak napríklad v ASCI kóde. Jednou z výhod použitia neurónových sietí je vysoká tolerancia čo sa týka kvality a šumu vstupného obrazu. Spoľahlivosť riešenia sa zlepšuje použitím inteligentného učiaceho sa systému nielen pre rozpoznávanie znakov, ale aj pri získavaní celých slov spájaných zo zistených znakov. Zaručuje to možnosť automatického zlepšovania výsledkov samočinným učením zariadenia [5],[9].

1.6.4 MLP (Viac vrstvový perceptron)

Viac vrstvový perceptron, so skratkou MLP (z angl. MultiLayer Perceptron), je dopredná (feed forvard) neurónová sieť, ktorá mapuje vstupné údaje k vhodným výstupom. Dopredná sieť sa vyznačuje tým, že neuróny siete sú prepojované v jednom smere, od vstupu na výstup. Obecná ANN má uzly prepojené v rôznych smeroch, čím vytvára zložitejšiu sieť ako MLP. MLP pozostáva z niekoľkých navzájom prepojených vrstiev uzlov. Výnimkou sú vstupné uzly tvorené neurónmi s nelineárnou aktivačnou funkciou. Na trénovanie siete sa využíva spätná distribúcia (backpropagation). Spätná distribúcia je proces, ktorý pozostáva z dvoch krokov: propagácia a aktualizácia váh. Váhy sa upravujú na základe funkcie popisujúcej chybu, ktorá vzniká pri porovnávaní výstupných dát s užívateľom preddefinovanými správnymi možnosťami. Sieť typu MLP pozostáva z viacerých vrstiev výpočtových jednotiek, ktoré sú navzájom prepojené v jednom smere[16].

1.7 Použitý softvér

Na trhu existuje veľké množstvo softvéru, ktorý sa zameriava na rozpoznávanie znakov.

Delenie OCR systémov podľa účelu:

- samotné rozpoznávanie znakov
- delenie dokumentu na bloky textu, riadky a znaky

Tesseract ([19]) je jedným z najvýznamnejších zástupcov OCR programov a zaoberá sa ako delením dokumentu tak samotným rozpoznávaním, preto sa nasledujúci text zameriava práve naň.

1.7.1 Tesseract

Tesseract je „open source“ softvér pod „Apache License 2.0“, čo znamená že je voľne šíriteľný, je ho možné použiť akýmkoľvek spôsobom, kód je možné upravovať a následne distribuovať aj jeho modifikované verzie v súlade s podmienkami licencie. Znaky nerozpoznáva v reálnom čase a je výsledkom výskumu oddelenia Hewlett Packard Labs s neskoršou spoluprácou s firmou Google. Pracuje ako nadstavba alebo vrcholný prvok využívajúci kľúčové funkčné moduly, ktoré predstavujú jednotlivé kroky prevodu textu do digitálnej podoby[20],[21].

Primárne neobsahuje žiadne grafické prostredie, ale pracuje pomocou príkazov v príkazovom riadku. Existujú však grafické nadstavby od vývojárov tretích strán. Pre rok 2017 je aktuálna verzia 3.05 [22].

Prvým stupňom je analýza prepojených súčastí (anglicky connected component analysis), ktorá je vo všeobecnosti zodpovedná za špeciálne označenie jednotlivých objektov obsiahnutých v analyzovanom dokumente. v tomto konkrétnom prípade pracuje na uložení obrysov komponentov. Tvorcovia použili túto analýzu aj napriek jej náročnej implementácii v dobe kedy Tesseract vznikal. Tento krok však zabezpečil jednoduchým spôsobom spracovanie bieleho textu na čiernom pozadí, pričom v čase kedy vznikol, bol jediný z pomedzi OCR programov, ktorý to dokázal. Vďaka tomu že pozná obrysy znakov, posudzuje ich za akýchkoľvek farebných variácii ako binárny čierno biely vstup [19].

V tomto bode sú obrysy znakov prepojené pomocou procesu sieťovania (anglicky nesting) do jednotných útvarov, takzvaných „blobs“ (viď. Obrázok 1.4). Blobs sú usporiadané do textových riadkov. Následne sa zisťuje buď fixné stúpanie písma v riadku alebo sa zavedie proporcionálny text. Riadky sa definitívne rozdelia na jednotlivé slová, pričom sa vychádza zo šírky medzier medzi znakmi. Ak je nastavené

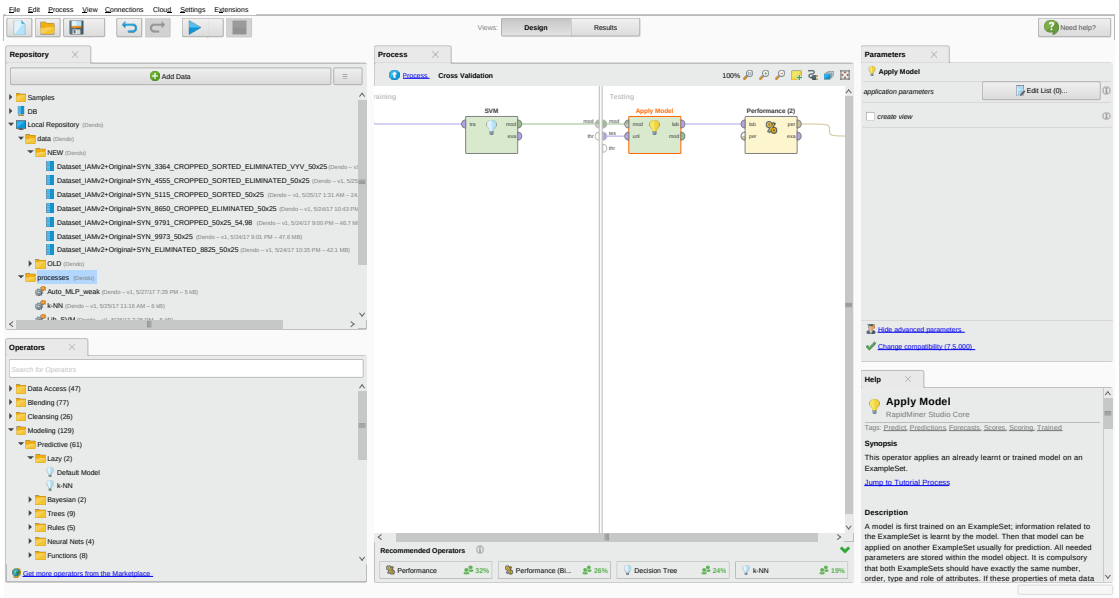
fixné stúpanie textu, tak sú riadky ihneď rozdelené na bunky znakov. Pre proporcionálny text platí rozdelenie do slov pomocou definovaných medzier a medzier, ktoré sú neurčité [23],[24].

Obr. 1.4: Znak a jeho blob útvar.

Rozpoznávanie samotné potom prebieha v dvoch totožných procesoch. Prvý z nich prebieha tak, že sa prechádzajú všetky predom rozdelené slová od začiatku až po koniec textu, pričom cieľom je rozpoznať každé slovo na prvý krát. Každé slovo, ktoré má dostatočne uspokojivý výsledok a zmysel, teda existuje v známej databáze slov, je odoslané do adaptívneho zoraďovača a predstavuje trénovacie dáta. Takto sa zväčšuje šanca že slová, ktoré sú na strane nižšie, teda ďalšie v poradí, rozozná adaptívny zoraďovač presnejšie a spoľahlivejšie. Druhý krok prejde tú istú analýzu ešte raz, takže dáva ďalšiu šancu pre lepšie zoradenie, čo zaručí správne vyhodnotenie slov aj v hornej, počiatočnej časti textu, keďže sa zoraďovač mohol naučiť dôležitú súvislosť až neskôr v texte. Konečná fáza rieši neurčité medzery a kontroluje alternatívne možnosti výšky textu aby mohol nájsť aj malé písmo [19], [24].

1.7.2 RapidMiner

RapidMiner je softvérový nástroj na vykonávanie, úpravu a spravovanie prediktívnych analýz. Umožňuje aplikáciu rôznych metód strojového učenia. Na komunikáciu s užívateľom a dátami slúži grafické prostredie založené na vytváraní procesov spájaním blokov. Tie predstavujú jednotlivé metódy, matematické, logické a organizačné operácie, rôzne úpravy dát a iné. Bloky je možné jednoducho prepájať navzájom a prepájať s importovanými dátami. Výsledné natrénované metódy a procesy je možné exportovať do formátu RMP (RapidMiner proces) alebo XML a následne ich aplikovať do vlastných aplikácií. RapidMiner podporuje skriptovanie vo viacerých jazykoch. v práci bol použitý klient RapidMiner Studio vo verzií 7.5, ukážka grafického prostredia je na obrázku 1.5[25].



Obr. 1.5: Grafické prostredie klienta RapidMiner Studio.

1.8 Aktuálny prístup k OCR

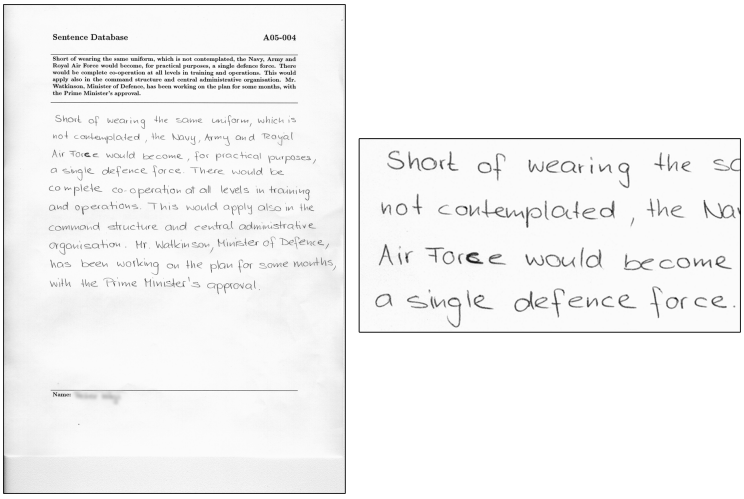
1.8.1 Prístup s využitím variabilných vertikálnych hraníc výrezu

Práca opisovaná v [26] je mierená na rozpoznávanie prirodzených scén, ale vychádza z prístupov používaných pri rozpoznávaní rukou písaného textu. Tento systém pracuje tak, že celý dokument prechádza posuvným oknom, typicky krokom o veľkosti jednej osminy výšky dokumentu. Tento krok predstavuje kontrolu dokazujúcu, že aspoň jedno okno je zarovnané s každým znakom. Keďže text nadobúda rôzne rozmery, obraz sa následne prechádza viackrát za použitia rôznych šírok posuvného okna. Ďalším krokom je klasifikácia znakov, ktorá identifikuje neplatné znaky a rozpozná tie platné. Posuvné okno však môže zachytiť aj okraje susedných znakov, čo zhoršuje výsledky rozpoznávania. Autori sa pokúsili tomuto problému vyhnúť, a to úpravou vertikálnych hraníc výrezu, vždy keď je to možné. Informácie obsiahnuté v pixeloch sú kategorizované do dvoch tried: text a pozadie. Tento krok zabezpečuje preprocessing, ktorý pre každý pixel vytvorí mapu neurčitých hodnôt. Na túto mapu sa aplikuje algoritmus najkratšej cesty. Nelineárne vertikálne hranice sú prepočítané na základe pravdepodobnosti výskytu pixelov patriacich do triedy text. v prípade, kedy znaky nie sú oddelené, algoritmus vytvorí rovné vertikálne hranice, keďže pixely v lokálnej oblasti majú rovnakú pravdepodobnosť. Na klasifikáciu znakov sa využívajú kontextovo konvolučné neurónové siete. Tento typ sietí je schopný sa na-

učiť vizuálne vzory z obrazu bez korekcií. Sú málo náchylné na šum a zakrivenia textu, a boli už testované aj na rozpoznávanie ručne písaného textu. Tento prístup dosiahol, na tvorcami zvolenom testovacom súbore, presnosť rozpoznávania 66,19 % a tvorcami bol porovnaný so softvérom Tesseract s výsledkom 35 % [26].

1.8.2 Prístup s využitím ANN a HMM

Jedným z aktuálnych prístupov je práca s hybridnými HMM (Hidden Markov model – skrytý markov model) a ANN (Artificial Neural Network – umelá neurónová sieť). Posuvné okno sa tu využíva na dodávanie vstupných informácií do neurónových sietí. Táto metóda je opísaná v článku [20]. Trénovací a testovací dataset využíva dokumenty z IAM databázy [27]. Je to databáza naskenovaných, anglických, rukou písaných textov. Na obrázku 1.6 je vzor dokumentu z IAM databázy. Slúži na trénovanie OCR programov.



Obr. 1.6: Vzor dokumentu z IAM databázy.

V tomto systéme je riadok ručne písaného textu prevedený na sekvenciu vektorov príznakov. Prípady gramaticky zložitých textov vyžadujú rozpoznanie jednotlivých znakov, ktoré je nutné skladať do kompletných slov, patriacich do vopred danej slovnej zásoby. Metóda HMM sa využíva pri ručne písanom texte tak, že jeho počiatočnou informáciou je sekvencia vektorov príznakov a štatistickým spôsobom vyhľadávania vzorov sa vyhľadáva najlepšia zhoda so znakovou sekvenciou slova zo slovnej zásoby. Vstupné informácie pre neurónové siete v tomto prístupe predstavujú okná pozostávajúce z vybraného pixelu a jeho okolitých pixelov v požadovanom rozsahu. Geometrické úpravy ako napríklad odstránenie náklonu slov alebo sklonu znakov využívajú MLP. Histogram sa používa na zobrazenie medzery medzi znakmi[20].

Výsledky hybridných HMM/ANN modelov boli merané pomocou miery chybných slov WER (z angl. Word Error Rate), získanej ako pomer chýb vzniknutých pri rozpoznávaní k celkovému počtu slov v referenčnej sade. Finálne testy obsahovali aj meranie miery chybných znakov CER (z angl. Character Error Rate), ktorá sa od WER líši tým že slová v pomere sa nahradia znakmi. Tvorcovia metódy dosiahli najlepšie výsledky 15,5% WER a 6,9% CER, pri použití 7-stavového HMM, MLP 192-128 s uzavretým slovníkom obsahujúcim 4953 slov. [5], [28], [29].

1.8.3 Prístup s využitím BHMM

Jeden z prístupov spomínaných v [30], ktorý je tvorcami pomenovaný UPV-PRHLT a je založený na Bernoulliho HMM (BHMM), čo je HMM, v ktorom sa využívajú Bernoulliho zmesi pravdepodobnostných funkcií. Systém využíva posuvné okno o adekvátnej šírke, tak aby bol lepšie zachytený kontext obrazu znaku na oboch jeho stranách. Trénovacie obrazy sa zmenšia na výšku 30 pixelov, pričom je zachovaný pomer strán. Následne prebieha prevod obrazu do binárnej podoby. Použité je posuvné okno s výškou 9 pixelov, ktorého výstup je v podobe vektorov príznakov a predstavuje vstup do BHMM. Spomenuté sú tu dva spôsoby použitia posuvného okna. Prvým z nich je okno aplikované na každý stĺpec vstupného obrazu. v druhom prípade sa vytvorí výrez a posúdi sa rozloženie informácie jeho obsahu. Ak sa veľká časť informácie, ktorú predstavuje text, nachádza v pravom dolnom rohu, stred nasledujúceho okna sa presunie na toto miesto. a tak sa to deje po každom použití posuvného okna. Týmto spôsobom sa zabezpečí, že okno neopustí riadok aj ak je naklonený. Presnosť rozpoznávania bola pri použití oboch typov pohybu okna 80% až 90% [30].

1.8.4 Prístup s využitím viacerých veľkostí posuvného okna

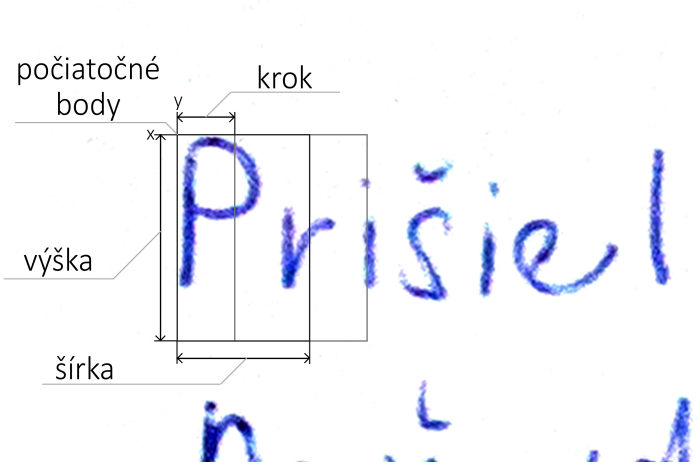
Prístup z [31] využíva posuvné okno na získavanie významných vlastností. Vstupný obraz je najprv zmenšený na jednotnú výšku 64 pixelov. Vstupný obraz môže mať rôzne rozlíšenia a tento krok zabezpečí jednotnosť vstupu. Tvorcovia opísali viac typov posuvného okna. Prvé popísané okno preberá celú šírku a postupuje s výškou 8 pixelov, druhé zase preberá celú výšku a prechádza obraz so šírkou 8 pixelov. Pomocou oboch okien sa prepočítava pomer medzi čiernou a bielou na obraze. Tretie okno predstavuje štvorec s dĺžkou strany 8 pixelov. Štvrtý spôsob používa tiež štvorcové okno, ale s počiatočnou veľkosťou strany 2 pixely. Prechádza celý obraz a ak narazí na čierny pixel, prekonvertuje sa celá plocha okna na čiernu farbu. Robí sa to z dôvodu, že posuvné okno následne lepšie reaguje na vyrovnané hrany znakov. Ďalej sa na takto konvertovaný obraz aplikuje ďalšie okno, s tou istou úlohou, ako prvý alebo druhý typ. Presnosť rozpoznávania bola je v tomto prípade 97,12% [31].

2 NÁVRH RIEŠENIA

V tejto kapitole sú popísane možnosti vytvárania výrezov znakov v rukou písanom texte pomocou vytvoreného okna s danou výškou, šírkou, krokom a počiatočnými bodmi. Toto okno musí následne prejsť všetky riadky textu vstupného obrazu. Vytvorené výrezy sú ukladané pre ďalšie spracovanie metódou rozpoznávania znakov. Ručne písaný text obsahuje charakteristické prvky, ktoré bolo potrebné zohľadniť pri vytváraní metódy na získavanie výrezov znakov.

2.1 Zhodnotenie možných riešení

Základným princípom je okno, ktoré reprezentuje výrez získaný zo vstupného obrazu. Aby bolo možné toto okno aplikovať, je nutné preň zabezpečiť niekoľko parametrov, ktoré zobrazuje obrázok 2.1.



Obr. 2.1: Parametre posuvného okna.

Jedným z nich je šírka okna. Príliš úzke okno spôsobí to, že metóda, ktorá rozoznáva znaky nemá dostatok informácie na to, aby správne rozhodla o danom znaku. Príliš široký výrez môže obsahovať aj viac ako jeden znak, a tak nie je dostatočne jasné, o ktorý znak sa jedná. Oba prípady môžu mať za následok chybné vyhodnotenie, a tak aj zníženie presnosti rozpoznania. Najjednoduchší spôsob je nastavenie pevnej šírky pre všetky analyzované dokumenty. v tomto prípade však nie je možné dostatočne pokryť líšiace sa charaktery písma. Variabilitu programu je možné rozšíriť použitím niekoľkých šírok okna pre jeden dokument. v tom prípade je potrebné vyhodnotiť výrezy a vybrať ten, ktorý je za daných okolností, najviac vhodný. Ďalšie riešenie nastaví šírku pomocou rozdielu dvoch bodov ohraničujúcich znak. Za hranicu znaku môže byť považovaná napríklad zmena farby v dokumente.

Výška okna musí byť dostatočná na to, aby bolo možné zachytiť aj vysoké znaky a veľké písmená. Najlepšie je použiť maximálnu výšku riadku. Problém tvoria texty, ktorých riadky sa prekrývajú. Keďže interpunkciu nie je nutné zachovávať, môžeme pristúpiť k orezaniu hornej časti okna. Jednou z možností je rozdelenie výrezov na dve časti podľa obsahu, pričom sa predpokladá, že nižšia časť je interpunkcia alebo zásah do iného riadku, a tak sa odstráni. v kritických prípadoch textov ani táto podmienka nezaručí, že výrez bude dostatočne zreteľný.

Počiatočné body X a Y, od ktorých postupuje okno je potrebné určiť. Je možné hľadať grafické zmeny medzi znakmi a ich podkladom, čo značí prvý znak v riadku, a tak je možné použiť jeho súradnice výšky a šírky ako počiatočný bod pre okno. Ďalším prístupom je princíp pohyblivého okna s podmienkou, ktorá zaručí zapísanie počiatočných súradníc v prípade že narazí na znak a teda už spomínanú grafickú zmenu, napríklad na zmenu farby.

Krok posuvného okna udáva vzdialenosť, po ktorej sa vytvárajú výrezy. Táto vlastnosť rozhoduje o dostatočnej čitateľnosti výsledného znaku. Veľký krok zaručí rýchlejšie delenie dokumentu na výrezy. Na druhú stranu však spôsobí nedostatočné pokrytie znakov. Krok, ktorý je väčší než šírka okna znamená, že medzi výrezmi vzniknú nepokryté miesta. Aby bol zaručený prenos celého dokumentu, veľkosť kroku by mala byť menšia než je šírka okna. Toto nastavenie ošetrí aj niektoré prípady prekrývajúcich sa znakov. Príklad na obrázku 2.2 znázorňuje prekrytie znakov, kedy je veľkosť kroku rozhodujúca pre ich správnu klasifikáciu.

Obr. 2.2: Prekrytie znakov.

2.2 Navrhované metódy posuvného okna

2.2.1 Posuvné okno s vyraďovaním výrezov

Prvá z možných metód využíva premenlivé parametre posuvného okna a aplikuje ho na celý dokument. Najprv sa nastavia parametre okna podľa potreby alebo sa me-

nia v nastavenom rozsahu. Takto sa vytvorí sada parametrov posuvného okna. Okno postupuje od počiatku dokumentu a výrezy sú ukladané s informáciou o riadku, a poradovom čísle znaku. Proces sa opakuje pre všetky sady parametrov. Ďalej sa vyhodnocuje obsah výrezov podľa pomeru čiernych a bielych pixelov, pričom podmienka vyradí výrezy obsahujúce nedostatočne veľkú oblasť znaku. Posuvné okno však nemusí mať správne parametre na to, aby dostatočne pokrylo celú výšku riadkov. Premenlivá medzera medzi riadkami môže spôsobovať ďalšie problémy. Nesprávne nastavenie sady parametrov spôsobí vytváranie výrezov obsahujúcich len polovicu výšky znaku, čo vedie k nesprávnej klasifikácií znakov vo výrezoch. Metóda sa nedokáže automaticky prispôsobiť charakteristike spracovávaného textu.

2.2.2 Posuvné okno s vyhľadávaním prvého znaku

Táto metóda pracuje so štvorcovým posuvným oknom, ktorého účelom je vyhľadávanie prvého znaku. Okno nevytvára výrezy, ale jeho obsah sa po každom kroku kontroluje. Ak vybraná oblasť obsahuje len bielu plochu, okno postupuje ďalej. V prípade že okno obsahuje nejaké čierne pixely, vytvorí sa ďalšie posuvné okno, ktoré vytvára výrezy. Jeho parametre sa pevne nastavia na začiatku. Na konci riadku sa metóda vráti k malému štvorcovému oknu a pokračuje vo vyhľadávaní ďalšieho riadku, kde sa začnú znovu ukladať výrezy. Takto sa pokračuje až na koniec dokumentu.

Výsledkom sú sady znakov patriacich do jedného riadku, popísané ich poradovými číslami. Nesprávne nastavenie štvorcového posuvného okna vedie k prípadu, kedy sa počiatočný bod riadku vyhodnotí nesprávne a okno vytvárajúce výrezy tak nepokryje dostatočnú plochu znakov.

2.2.3 Posuvné okno s prevodom farieb

Prvá z implementovaných metód posudzuje okraje znaku na základe farebnej odlišnosti. Naskenovaný text je prevedený do binárnej podoby. Každému pixelu sa pridelí čierna alebo biela farba na základe porovnania priemeru množstva jeho červenej, modrej a zelenej farby s prahovou hodnotou. Šírka okna sa nastavuje postupne v požadovanom rozsahu. Ďalším krokom je vytvorenie okna s výškou celého dokumentu, v ktorom sa vyhľadávajú čierne pixely. Prvý čierny pixel predstavuje vrchnú hranicu znaku na osi Y. Ak je prvý čierny pixel zapísaný, vyhľadáva sa súvislý riadok bielych pixelov reprezentujúci spodnú hranicu znaku. Výška znaku je rozdielom súradníc týchto dvoch bodov. Okno vytvárajúce výrezy znakov preberie nastavenú šírku, výslednú výšku a počiatočné body, ktoré tvorí X súradnica okna a Y súradnica čierneho bodu. Jeden výrez sa uloží do vektoru svojho riadku a tie sa ukladajú

do dátovej štruktúry mapa spolu s číslom riadku. Súradnice potrebné na výpočet výšky sa premažú a okno s celou výškou dokumentu sa posunie o jeden krok.

Výsledkom je mapa vektorov s poradovými číslami riadkov. z mapy sa následne podľa čísla riadku vyvolávajú vektory, ktorých výrezy sa ukladajú pre ďalšie spracovanie. Výrezy majú v názve poradové číslo v riadku a nachádzajú sa v priečinkoch rozdelených podľa riadkov, z ktorých pochádzajú.

2.2.4 Posuvné okno s použitím Tesseract

Druhá implementovaná metóda obsahuje posuvné okno s využitím softvéru Tesseract. Využíva sa jeho vlastnosť vyhľadávania riadkov textu. Obraz sa načíta do premennej typu `BufferedImage`. Následne Tesseract vyhľadá riadky v texte a ich parametre (šírka, výška, súradnica X a Y) zapíše do premennej typu `Rectangle`. Vytvorí sa okno, ktoré preberá výšku získaného riadku a jeho počiatočné súradnice X a Y. Krok a šírka okna sa prepočítavajú na základe výšky riadku. Všetky parametre a súradnice sú udávané v pixeloch. Takto vytvorené okno prechádza celú šírku riadku. Jednotlivé výrezy vznikajú zapisovaním nového obrazu typu `BufferedImage` s parametrami okna. Následne sa ukladajú do vektoru pod jedinečným názvom. Názov je zložený z čísla riadku a čísla znaku. Každý ďalší výrez sa zaznamenáva o jeden krok ďalej. Ukážka kódu vytvárajúceho výrezy:

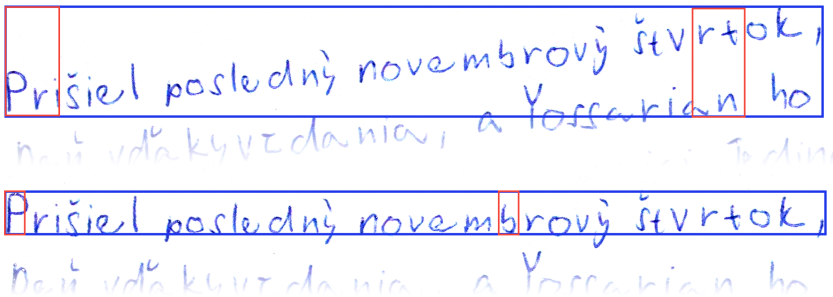
```
Tesseract instance = Tesseract.getInstance();
List<Rectangle> rect = instance.getSegmentedRegions(image, ->
-> TessPageIteratorLevel.RIL_TEXTLINE);
Rectangle r = rect.get(i)
Okno okno = new Okno();
okno.setSirka((r.height*10)/28);
okno.setVyska(r.height);
okno.setKrok((r.height*10)/95);
int pociatok = r.x;
BufferedImage image1 = image.getSubimage(pociatok, ->
-> r.y, okno.getSirka(), okno.getVyska());
```

2.3 Ďalšie pomocné metódy

2.3.1 Vyrovnávanie náklonu dokumentov

Pre lepšie vyhodnotenie výšky riadku a získavanie kvalitnejších výrezov znakov je vhodné korigovať náklon celého dokumentu. Na tento účel bol vytvorený proces

upravujúci naskenovaný dokument. Pomocou funkcie softvéru Tesseract sa vyhľadáva riadok dokumentu a dočasne sa uloží jeho výška. v ďalšom kroku sa obraz otáča po jednom stupni a znovu získaná výška riadku sa porovnáva s predchádzajúcou. Ak je nová výška menšia než predchádzajúca, pokračuje sa v otáčaní dokumentu, v opačnom prípade sa obraz otočí o jeden stupeň späť a uloží sa. Otáčanie prebieha pomocou metódy rotate knižnice `Graphics2D`, ktorej parametre sú súradnice osi otáčania a uhol natočenia v stupňoch. Získaný obraz má vyrovnaný náklon riadkov, čím sa predchádza problému s nerovnomerným rozložením znaku v okne výrezu. Tento prípad je znázornený na obrázku 2.3.



Obr. 2.3: Vyrovnávanie náklonu.

2.3.2 Konverzia do formátu CSV

Formát CSV (z angl. Comma-separated values) je pomenovaný podľa toho, že obsahuje hodnoty oddelené čiarkami. v tomto formáte sa vkladajú vstupné údaje, datasety, do softvéru RapidMiner1.7.2, pomocou ktorého sú trénované metódy pre rozpoznávanie znakov.

Metóda ukladá výrezy z hlavného priečinku datasetu do zoznamu súborov. Tie sú neskôr po jednom vyvolávané a každý obraz sa prispôsobí na pevne dané rozlíšenie. Príklad kódu pre zmenu rozlíšenia obrazu:

```
int new_height = 50;
int new_width = 25;
BufferedImage image = new BufferedImage(new_width, ->
-> new_height, BufferedImage.TYPE_INT_ARGB);
Graphics2D g = image.createGraphics();
g.drawImage(RawImage, 0, 0, new_width, new_height, null);
g.dispose();
```

Výsledok tejto úpravy sa nachádza na obrázku 2.5. Nasleduje cyklus, ktorý prechádza novo vytvorený obraz po jednotlivých pixeloch a do textového reťazca zapisuje

ich hodnotu. Pozadie reprezentuje 0 a znak 1. O tom kam spadá aktuálny pixel rozhoduje podmienka, ktorá porovnáva jeho priemernú hodnotu farby s prahovou hodnotou. Za každý zápis v reťazci sa pripíše oddeľovací znak a za informáciou o poslednom pixely sa pripíše štítok (label) popisujúci triedu, z ktorej znak pochádza. Túto informáciu proces čerpá z popisu priečinku, z ktorého daný obraz pochádza. Vo výslednom súbore sú znaky zapísané ako postupnosť ich atribútov vo vektorovom tvare, pričom jeden riadok predstavuje jeden znak.

2.4 Metódy úprav datasetu

Pre zlepšenie presnosti metód rozpoznávajúcich ručne písané znaky, bolo v tejto práci vytvorených niekoľko prídavných metód spracovávajúcich získané dáta.

2.4.1 Syntetické znaky

Zlepšenie presnosti zvýšením počtu znakov je možné docieliť doplnením nových výrezov do datasetu alebo vytvorením syntetických znakov. Tie môžu vzniknúť pokrivením alebo pootočením pôvodného znaku, čím sa odlišujú od svojho originálu. Týmto spôsobom je možné zlepšiť aj robustnosť metódy rozpoznávajúcej text, keďže bude natrénovaná na datasete s rôznymi variáciami znakov.

Obr. 2.4: Porovnanie pôvodného a syntetických znaku.

Implementovaná metóda otáča každý obraz a vytvorí tak nový vzor do datasetu. Do parametrov príkazu rotate knižnice `Graphics2D` sa zapíše požadovaný uhol natočenia a pôvodný obraz. v tomto prípade bol pokusne zvolený uhol 15°. Menší uhol nedostatočne odlišoval niektoré znaky a pri použití väčšieho uhla dochádzalo k orezaniu častí znakov. Výsledok sa ukladá do nového priečinku, pričom sú zachované triedy znakov. Vzorové porovnanie pôvodného a syntetického znaku je na obrázku 2.4.

2.4.2 Orezanie výrezu znaku

V tejto metóde sa upravujú okraje získaného výrezu znaku tak, aby boli čo najbližšie k telesu znaku. Tento krok uľahčí trénovanie metód pracujúcich na základe spojitostí medzi bodmi znaku, atribútmi. Telesá znakov sa totižto nachádzajú približne na rovnakých miestach v celkovom obraze, a tak sú si ich vektorové zápisy viac podobné. To sa netýka prípadu neorezaného znaku. Tento problém znázorňuje porovnanie na obrázku 2.5. Na vyhľadanie najbližších hraníc telesa znaku vo výrezoch sa využíva funkcia softvéru Tesseract, vracajúca parametre štvoruholníku ohraničujúceho znak.

Obr. 2.5: Pôvodné výrezy, ich orezané znaky, po úprave rozlíšenia.

2.4.3 Triedenie výrezov znakov

Zlepšenie spoľahlivosti rozpoznávania znakov je možné dosiahnuť manuálnym vytriedením nevhodných prvkov datasetu. Po automatizovanom získavaní a orezávaní znakov je vhodné vyhľadať, a vylúčiť z datasetu také obrazy, ktoré nezodpovedajú triede znaku. Takéto obrazy môžu obsahovať orezané, neúplné znaky, interpunkciu alebo výrezy malých častí písmen a iné. Spomenuté výrezy zhoršujú schopnosť vyhľadávania vzorov medzi znakmi, keďže so zvyškom triedy nemajú žiadne spoločné vlastnosti. v práci boli takto vytriedené niekoľké datasety. Ich porovnanie sa nachádza v tabuľke s výsledkami 3.3.

2.4.4 Vyváženie datasetu

Vyváženie datasetu znamená vyrovnanie počtov znakov v jeho triedach. To je možné docieliť zmazaním určitého počtu znakov v triedach s ich vysokým počtom alebo doplnením nových výrezov, prípadne syntetických znakov, do tried s malým počtom výrezov. Alternatívnym riešením je pridelenie rôznych váh jednotlivým triedam výrezov. v takom prípade je nutné použiť takú metódu pre rozpoznávanie znakov, ktorá zohľadňuje nastavené váhy. Použitý softvér RapidMiner podporuje nastavenie váh jednotlivým atribútom, nie však triedam, takže táto možnosť nie je zohľadnená vo výsledkoch tabuľky 3.3.

2.4.5 Eliminácia málo početných tried

Obnos informácií datasetu je zvyčajne medzi triedami rozložený nerovnomerne. To znamená, že trieda písmena „e“ obsahuje napríklad 300 znakov, zatiaľ čo trieda „f“ zahŕňa len 20 výrezov. Vďaka týmto rozdielom môže dôjsť k problémom pri rozpoznávaní znakov. Natrénovaná metóda pridelí znaku z málo početnej triedy štítok triedy, ktorá má prevahu počtu znakov v datasete.

Tomuto problému je možné predísť vyvážením datasetu alebo elimináciou málo početných tried. Výsledky týchto metód sú v tabuľke 3.3.

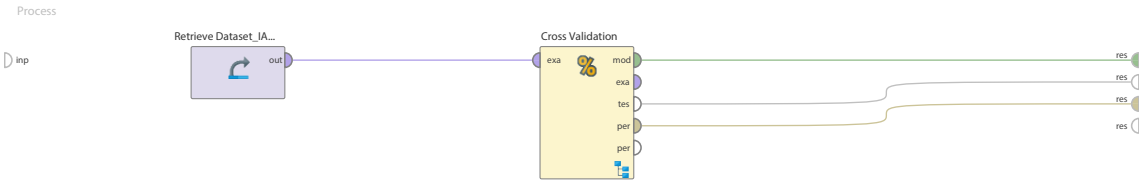
2.5 Tvorba datasetov

Dataset predstavuje súhrn získaných dát, ktoré sú združené v jednom celku, v tomto prípade výrezov znakov. Získaný dataset je možné využiť na účely trénovania alebo testovanie použitých metód. Jeho formát záleží na programe, ktorý trénuje metódu, v tejto práci sa využíva formát CSV. Pomocou úprav popísaných v kapitole 2.4 vzniklo niekoľko variant datasetov. Všetky výrezy v datasetoch sú vytvorené metódou posuvného okna s využitím softvéru Tesseract 2.2.4. Zdrojové súbory pre získavanie výrezov pochádzajú z IAM databázy1.8.2. Niekoľko ďalších výrezov pochádza z textov poskytnutých vedúcim práce.

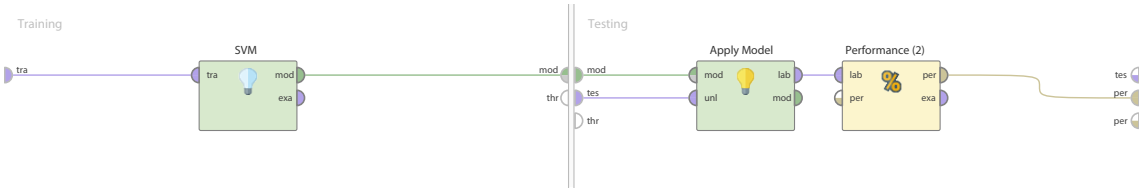
2.6 Trénovanie OCR metód

Aby bolo možné niektorú z OCR metód použiť v automatizovanom procese, je nutné uskutočniť jej trénovanie. Jeden zo softvérov, ktoré umožňujú tento proces je RapidMiner popísaný v časti 1.7.2. Je použitý aj v tejto práci. Disponuje grafickým prostredím, kde sa vytvorí proces na trénovanie a vyhodnotenie zvolenej metódy strojového učenia. Celý proces spočíva v aplikácií vybraného modelu na dataset, ktorý bol rozdelený na trénovaciu a testovaciu časť. Program RapidMiner natrénuje model podľa zvolenej metódy.

Obrázky 2.6 a 2.7 zobrazujú príklad procesu krížovej validácie metódy SVM popísanej v 1.6.2. Na obrázku 2.6 je zobrazený blok vstupného datasetu a krížovej validácie. Obrázok 2.7 zobrazuje blok SVM, ktorý poskytne natrénovaný model pre rozpoznávanie znakov, blok aplikujúci natrénovaný model na testovací dataset a blok posudzujúci presnosť metódy.



Obr. 2.6: RapidMiner proces krížovej validácie metódy SVM.



Obr. 2.7: RapidMiner proces metódy SVM.

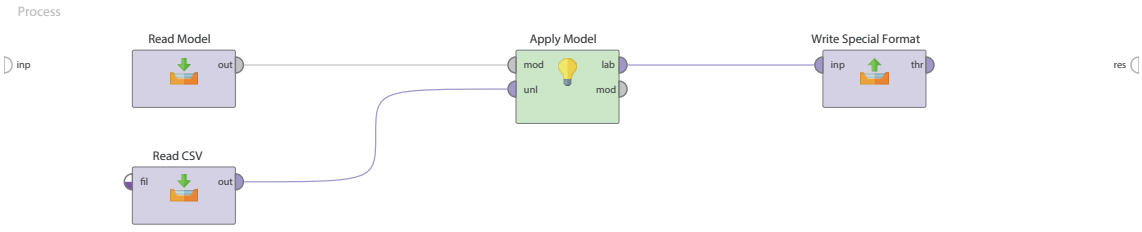
2.7 Automatizácia procesu

Automatický program rozpoznávania znakov využíva proces vytvorený v softvéri RapidMiner (1.7.2). Proces potrebuje prístup k natrénovanému modelu, ktorý je tiež vytvorený pomocou softvéru RapidMier.

Metóda pred samotným spustením procesu vyrovná náklon dokumentu, vytvorí výrezy znakov, tie sa orezávajú a ich informácie sa zapíšu do súboru typu CSV. Tieto úpravy sú popísané v kapitolách 2.3 a 2.4.

Následne sa načíta súbor s uloženým procesom, ktorý otvorí vytvorený súbor CSV a pomocou natrénovaného modelu strojového učenia klasifikuje jeho dáta. Výsledkom je nový dokument so štítkami rozpoznaných znakov.

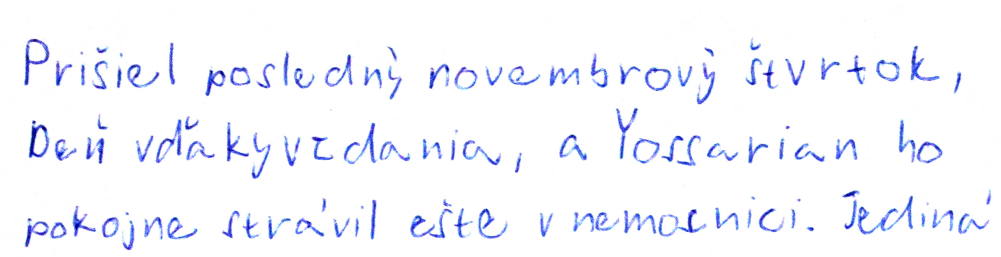
Obrázok (2.8) zobrazuje proces na rozpoznávanie znakov v prostredí RapidMiner Studio.



Obr. 2.8: RapidMiner proces na rozpoznávanie znakov.

3 VÝSLEDKY

V tejto časti práce sú prezentované výsledky metód vytvárania výrezov. Posudzované sú nastavenia rôznych parametrov. Sú tu tiež popísané problémy, ktoré spôsobuje nesprávne nastavenie parametrov metód. Pre nepostačujúce výsledky a náchylnosť na zložitosť textu, boli v sekcií výsledky vynechané tieto metódy: Posuvné okno s vyraďovaním výrezov 2.2.1 a Posuvné okno s vyhľadávaním prvého znaku 2.2.2. Na testovacie a demonštračné účely bol použitý nasledujúci vzorový text.



Obr. 3.1: Úryvok testovacieho textu.

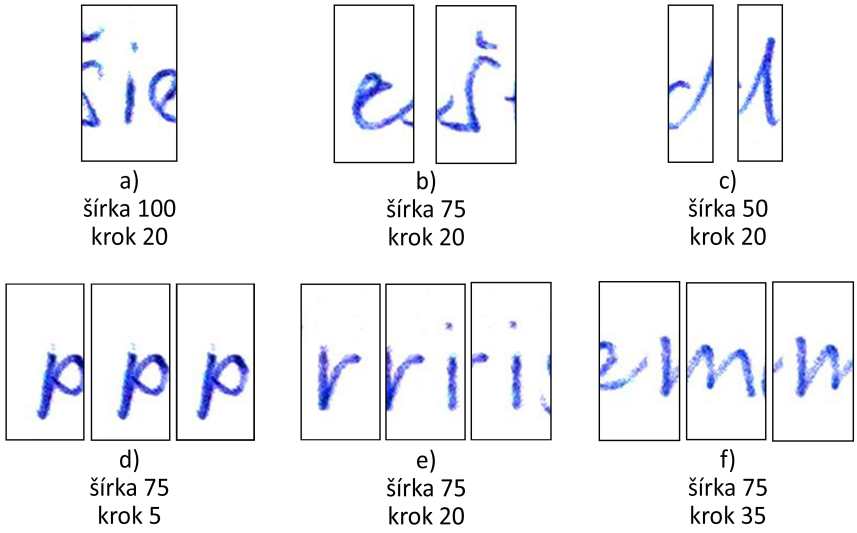
3.1 Posuvné okno s použitím Tesseract

Metóda bola testovaná s rôznymi nastaveniami parametrov šírky okna a dĺžky kroku. Ostatné parametre okna metóda nastavuje automaticky pre každý spracovávaný text. Prípad a) na obrázku 3.2 má nastavené príliš široké okno, a tak sa pri niektorých znakoch stane, že výrez ich obsahuje niekoľko naraz. Na obrázku 3.2 (c) je zobrazené veľmi úzke okno, ktoré nemusí pokryť celý znak. Malý krok na obrázku 3.2 (d), vytvára zbytočne veľa výrezov, ktoré obsahujú ten istý znak. Veľký krok posuvného okna na obrázku 3.2 (f), vytvorí výrezy obsahujúce malú časť znaku. Všetky nesprávne nastavenia môžu viesť k chybnej segmentácií znaku. Pre testovací text 3.1 dosahovala táto metóda najlepšie výsledky pri šírke okna 75 pixelov a kroku s dĺžkou 20 pixelov, zobrazené na obrázku 3.2 (b) a (e). Pri tomto nastavení je väčšina znakov dostatočne čitateľná a dostatočne oddelená.

3.2 Posuvné okno s prevodom farieb

V tomto prístupe nie je výška znaku jednotná pre celý riadok, ale každý znak má svoju vlastnú výšku. Ovplyvniť je však možné krok a šírku výrezu.

Zbytočne široké okno bude vytvárať výrezy obsahujúce viac znakov, čo zároveň obmedzí možnosť vlastného nastavenia výšky pre každý znak. Výška sa totiž v takomto prípade nastaví pre vyšší zo znakov. Pre nastavenie šírky a kroku platí to isté



Obr. 3.2: Ukážky nastavenia parametrov posuvného okna s použitím Tesseract.

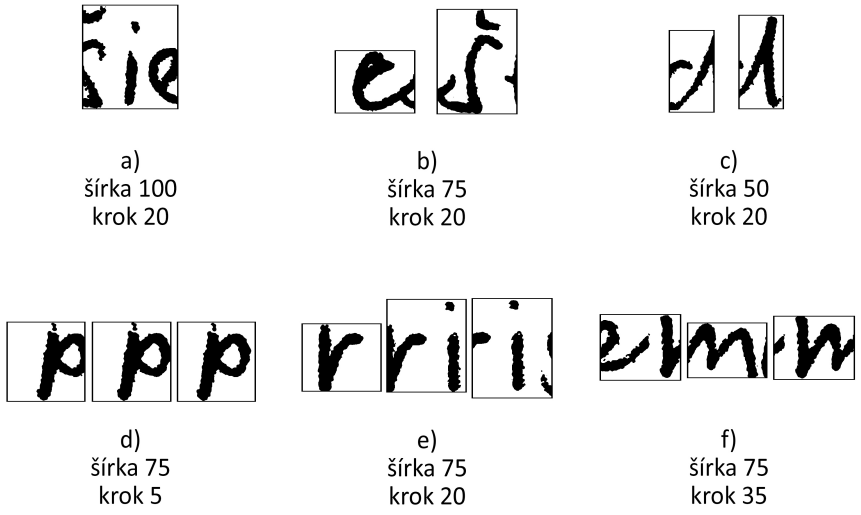
ako v predchádzajúcej metóde.
Najlepšie výsledky sú zhodné ako v predchádzajúcom prípade pri nastavení šírky okna 75 pixelov a veľkosti kroku 20 pixelov.

3.3 Porovnanie metód posuvného okna

Výsledné výrezy metód posuvného okna s prevodom farieb a posuvného okna s použitím Tesseract sú si podobné, ak sa zameriame na zretelnosť výrezov spojených alebo prekrývajúcich sa znakov. Výhodou metódy posuvného okna s prevodom farieb je schopnosť spracovať aj text s naklonenými riadkami. Ak majú byť výrezy zhodnotené všeobecne, prvým rozdielom bude ich výška.

Posuvné okno s prevodom farieb vytvára výrezy s premenlivou výškou, na základe výšky jednotlivých znakov. Posuvné okno s použitím Tesseract vytvára výrezy s jednotnou výškou riadku. Na tomto poznatku by bolo možné založiť systém na rozpoznávanie znakov využívajúci jednotnú výšku výrezov na to, aby rozdelil znaky do troch skupín: s hornou časťou (napríklad písmeno f), s dolnou časťou (napríklad písmeno p) a s centrálnou časťou (napríklad písmeno a). Bolo by tak možné rýchlejšie rozpoznávanie znakov, keďže by sa zameriaval len na jednu z týchto kategórií a nie na celú množinu znakov.

Práca ďalej využíva metódu posuvného okna s použitím Tesseract 2.2.4, pretože umožňuje jednoduchšiu a rýchlejšiu modifikáciu vlastných parametrov metódy, a väčšiu variabilitu pri ďalšom spracovaní a úpravách výrezov.



Obr. 3.3: Ukážky nastavenia parametrov posuvného okna s prevodom farieb.

3.4 Vytvorené datasety

V práci boli vytvorené datasety, pomocou ktorých boli merané výsledky presnosti metód a vyhodnocované ich možnosti. Vlastnosti použitých datasetov popisuje tabuľka 3.1. Najmenší rozdiel medzi minimálnym a maximálnym počtom znakov v jednej triede predstavuje najlepšie vyvážený dataset. Výsledky presnosti metód všetkých vytvorených datasetov sú v tabuľkách 3.2 a 3.3.

Tab. 3.1: Parametre vytvorených datasetov.

Úpravy datasetu	**Počet Znakov**	**Počet tried**	**Min. počet znakov v jednej triede**	**Max. počet znakov v jednej triede**
Pôvodný dataset	671	50	1	76
Zvýšenie počtu znakov	9973	52	2	705
Eliminácia málo poč. tried	8825	19	278	705
Orezanie znakov	9791	48	2	703
Orez. + Eliminácia tried	8650	19	274	705
Orez. + Triedenie znakov	5115	38	4	401
Orez. + Triedenie + Elim. tr.	4555	19	73	401
Všetky + Vyváženie datasetu	3364	18	99	209

Pôvodný dataset je získaný z 11 naskenovaných ručne písaných dokumentov a nie je nijak upravený. Obsahuje málo znakov, len 671 a počty znakov v triedach sú nevyvážené.

Dataset so zvýšeným počtom znakov vychádza z pôvodného datasetu a pridáva k nemu výrezy z 24 dokumentov IAM databázy (1.8.2), plus syntetické znaky získané

metódou zo sekcie 2.4.1. Nie je žiadnym spôsobom upravovaný a výsledný počet znakov je 9973.

Orezanie znakov prebehlo úpravou výrezov z datasetu so zvýšeným počtom znakov. Postup orezávania znakov je popísaný v časti 2.4.2.

Eliminácia málo početných tried je vysvetlená v úseku 2.4.5 a počet znakov takto upraveného datasetu klesol na 4555.

Triedenie výrezov prebiehalo na datasete orezaných znakov. Počet vzoriek tým klesol na 5115. Metóda vyraďovania nevhodných znakov je vysvetlená v časti 2.4.3.

Vyváženie datasetu bolo docielené odstránením znakov z tried s ich vysokým počtom, ako popisuje sekcia 2.4.4. Tým sa vyrovnali početnosti znakov vo všetkých triedach. Počet znakov klesol na 3364. Úprava bola aplikovaná na vyvážený dataset orezaných znakov s elimináciou tried.

3.5 Výsledky rozpoznávania znakov

Výsledkom získaným pomocou krížovej validácie softvéru RapidMiner je matica zámen (confusion matrix) porovnávajúca predikciu znaku a jeho správny štítok. Krížová validácia je popísaná v sekcií 1.5.5. RapidMiner ďalej poskytuje aj údaj o presnosti, vypočítanej z matice zámen. Presnosť je vyjadrená v percentách a pre rôzne metódy strojového učenia zobrazená v tabuľkách 3.2 a 3.3. Všetky použité datasety obsahujú výrezy znakov s rozlíšením 25x50 pixelov.

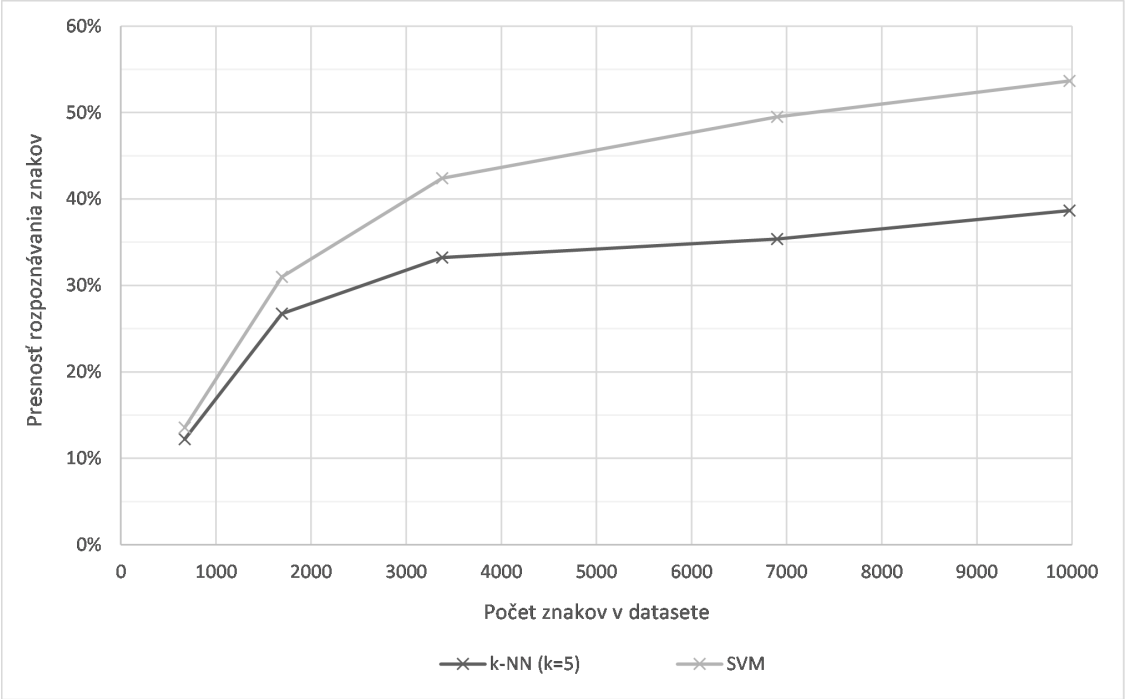
3.5.1 Závislosť presnosti na počte znakov v datasete

Táto časť obsahuje meranie presnosti niekoľkých vytvorených datasetov. Všetky výrezy v nich obsiahnuté majú rovnaký zdroj a formu, mení sa len ich počet. Zvýšenie počtu vzorov je docielené pridávaním nových znakov do datasetov a zaradením syntetických znakov.

Tabuľka 3.2 obsahuje meranie presnosti metód na počte znakov v datasete. Graf 3.4 zobrazuje závislosť tohoto merania. Meranie dokazuje, že so zvyšujúcim sa počtom znakov rastie aj presnosť rozpoznávania znakov. Percentuálny prírastok presnosti medzi bodmi merania sa však postupne znižuje. Preto je vhodné zaviesť úpravy datasetov. Upravené datasety sú zhodnotené v časti 3.5.2.

Tab. 3.2: Meranie presnosti metód v závislosti na počte znakov.

Počet znakov	**Presnosť OCR metód** [%]	
	k-NN	**SVM**
671	12,22	13,56
1694	26,74	30,99
3378	33,24	42,39
6900	35,38	49,49
9973	38,67	53,65



Obr. 3.4: Závislosť presnosti na počte znakov v datasete.

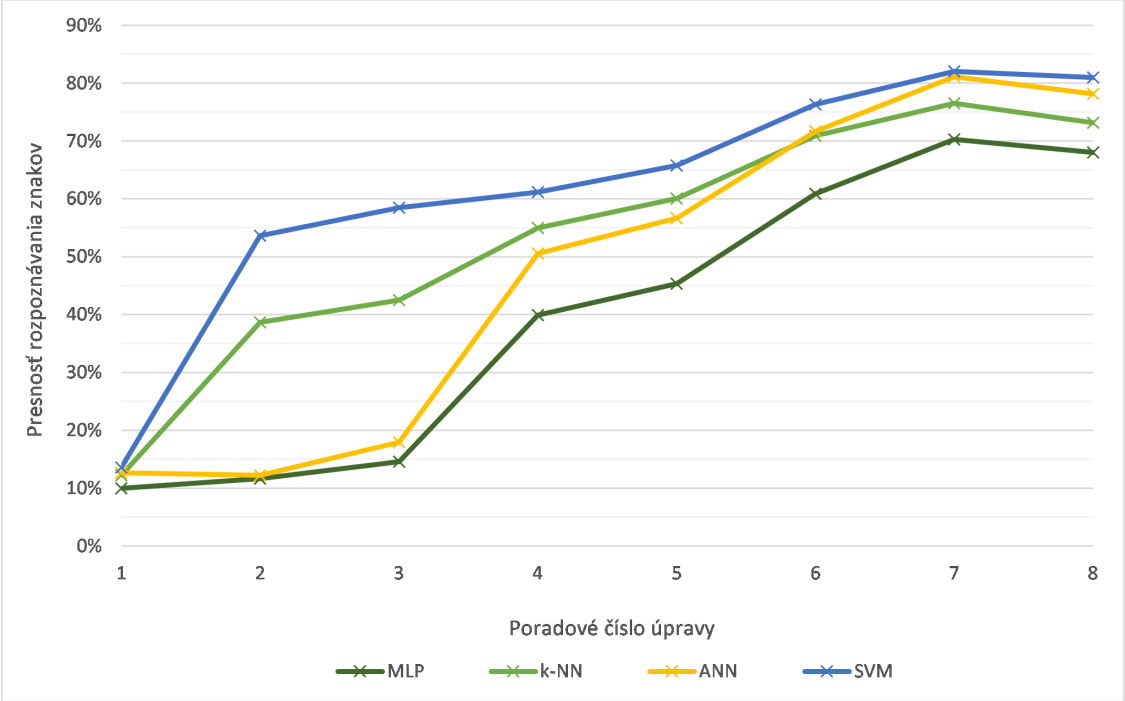
3.5.2 Závislosť presnosti na úpravách datasetu

Tabuľka 3.3 a graf 3.5 zobrazujú vplyvy metód úprav datasetu zo sekcie 2.4 na výslednú presnosť rozpoznávania znakov. V prípade použitia metód k-NN a SVM má značný vplyv na výsledok zvýšenie počtu znakov. Metódy ANN a MLP zaznamenali nárast úspešnosti rozpoznávania po orezaní znakov. Triedenie znakov a eliminácia tried majú podobný vplyv na všetky metódy. Vyváženie datasetu mierne zhoršilo úspešnosť všetkých metód, pretože sa týmto krokom znížil počet znakov v datasete. Vyváženie bolo docielené vynechaním málo početných tried a znížením počtu znakov v triedach s ich vysokým počtom. Merané metódy sú popísané v časti 1.6.

Metóda k-NN bola meraná s nastavením parameteru k=5. Použitá metóda MLP má 10 trénovacích cyklov, 3 generácie a 5 MLP prvkov na jeden celok. Meraná metóda SVM je typu C-SVC, s polynomiálnym jadrom, stupňom 3, gamou 5, metrikou C=1 a parameterom epsilon=0.001. Metóda ANN má 100 trénovacích cyklov, jej miera učenia (learning rate) je nastavená na 0,3, moment (momentum) na 0,2 a parameter epsilon chyby (error epsilon) na 0,00001.

Tab. 3.3: Meranie presnosti metód podľa úprav.

P.č.	Úpravy datasetu	Poč. Zn.	Poč. Tried	Presnosť OCR metód [%]			
				k-NN	SVM	ANN	MLP
1	Pôvodný dataset	671	50	12,22	13,56	12,67	9,98
2	Zvýšenie počtu znakov	9973	52	38,67	53,65	12,26	11,68
3	Eliminácia málo poč. tried	8825	19	42,48	58,49	17,95	14,57
4	Orezanie znakov	9791	48	54,98	61,15	50,58	39,91
5	Orez. + Eliminácia tried	8650	19	60,08	65,78	56,67	45,35
6	Orez. + Triedenie znakov	5115	38	70,91	76,32	71,65	60,86
7	Orez. + Triedenie + Elim. tr.	4555	19	76,49	82,02	81,12	70,27
8	Všetky + Vyváženie datasetu	3364	18	73,16	80,98	78,15	68,01



Obr. 3.5: Závislosť presnosti na úpravách.

4 ZÁVER

Cieľom práce bol návrh metód pre rozpoznávanie rukou písaných textov, ich automatizácia a zhodnotenie výsledkov.

V teoretickom úvode boli zhodnotené metódy a postupy využívane pri optickom rozpoznávaní znakov. Patria k nim aj metódy strojového učenia, ktoré boli v práci využité na samotné rozpoznávanie znakov.

Na účel segmentácie znakov boli navrhnuté metódy využívajúce posuvné okno a dve z nich boli implementované pomocou programovacieho jazyka Java. Výrezy získané pomocou metódy využívajúcej softvér Tesseract, boli použité na tvorbu datasetov, na ktorých boli trénované a testované vybrané metódy strojového učenia. Meranie ich presnosti prebehalo pomocou krížovej validácie v programe RapidMiner.

Presnosť rozpoznávania znakov bola vyhodnotená pre rôzne typy datasetov. Prvým meraným aspektom bol počet znakov, pričom meranie dokázalo, že so zvyšujúcim sa počtom výrezov v datasete, rastie aj presnosť metód strojového učenia. Metóda SVM dosiahla pri použití datasetu so 671 výrezmi presnosť 13,56%. Po zvýšení počtu výrezov na 9973 presnosť stúpla na 53,65%. v prípade metódy k-NN nastalo zlepšenie z 12,22% na 38,67%. Výsledky sa nachádzajú v tabuľke 3.2.

Ďalej boli v práci vytvorené a použité metódy pre zlepšenie presnosti rozpoznávania znakov. Tieto metódy upravujú datasety. Značné zlepšenie presnosti metód ANN a MLP nastalo po orezaní znakov. Pre metódu MLP to bolo zlepšenie zo 14,57% na 39,91% a výsledok metódy ANN stúpol zo 17,95% na 50,58%.

Vo vytvorených datasetoch boli vytriedené nevhodné znaky a eliminované málo početné triedy znakov. Vďaka kombinácií týchto úprav stúpla u všetkých metód rozpoznávania znakov ich presnosť. Pri použití metódy SVM konkrétne na 80,98%.

Vyváženie datasetu naopak o niekoľko percent zhoršilo presnosť rozpoznávania, dôsledkom toho, že sa týmto krokom znížil počet výrezov. Porovnanie úprav datasetov sa nachádza v tabuľke 3.3.

Vtvorené metódy získavania a úprav výrezov, spolu s natrénovaným modelom strojového učenia, boli použité v automatizovanom programe na rozpoznávanie znakov. Tento proces popisuje kapitola 2.7.

LITERATÚRA

[1] XUE, Yafang. *Optical Character Recognition.* [online]. 2014 [cit. 19. 10. 2016]. Dostupné z URL: <https://pdfs.semanticscholar.org/5b6b/e3357dbdbac38e92515a7b7aebb7e622f635.pdf>.

[2] VERMA, Abhishek; ARORA, Suket; VERMA, Preeti. *OCR-OPTICAL CHARACTER RECOGNITION.* [online]. 2016 s. 230-240. [cit. 19. 10. 2016]. Dostupné z URL: <http://data.conferenceworld.in/ICRISEM7/P230-240.pdf>.

[3] WANG, Sun-Chong. *Wang, S. C. (2003). Artificial neural network.* IEEE transactions on pattern analysis and machine intelligence [online]. Boston, MA: Springer US, 2003, 81-100 [cit. 28. 10. 2016]. Dostupné z URL: <http://link.springer.com/10.1007/978-1-4615-0377-4_5>.

[4] RABINER, Lawrence; JUANG, B. *An introduction to hidden Markov models.* ieee assp magazine [online]. 1986, 3(1), 4-16 [cit. 28. 10. 2016]. Dostupné z URL: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1165342>.

[5] ESPANA-BOQUERA, Salvador, a iní. *Improving offline handwritten text recognition with hybrid HMM/ANN models.* IEEE transactions on pattern analysis and machine intelligence [online]. 2011, 33(4), 767-779 [cit. 28. 10. 2016]. Dostupné z URL: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5551147>.

[6] NEUMANN, Lukas; MATAS, Jiri. *A method for text localization and recognition in real-world images.* Asian Conference on Computer Vision. [online] 2010 s. 770-783. [cit. 19. 10. 2016]. Dostupné z URL: <http://cmp.felk.cvut.cz/~matas/papers/neumann-text-accv10.pdf>.

[7] BARVE, Sameeksha. *Optical character recognition using artificial neural network.* International Journal of Advanced Research in Computer Engineering and Technology. [online] 2016 s. 2278-1323. [cit. 19. 10. 2016]. Dostupné z URL: <http://ijarcet.org/wp-content/uploads/IJARCET-VOL-1-ISSUE-4-131-133.pdf>.

[8] SINGH, Raghuraj, a iní. *Optical character recognition (OCR) for printed devnagari script using artificial neural network.* International Journal of Computer Science & Communication [online]. 2010, 1(1), 91-95 [cit. 28. 10. 2016]. Dostupné z URL: <http://www.csjournals.com/IJCSC/PDF1-1/18.pdf>.

[9] PATIL, V. V.; SANAP, Rajharsh Vishnu; KHARATE, Rohini Babanrao. *Optical character recognition using artificial neural network.* Int. J. Eng. Res. General Sci [online]. 2005, 3(1), 73-76 [cit. 28. 10. 2016]. Dostupné z URL: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.675.1803&rep=rep1&type=pdf>.

[10] *10-fold Crossvalidation.* Openml [online]. [cit. 2017-06-01]. Dostupné z URL: <https://www.openml.org/a/estimation-procedures/1>.

[11] SINGH, Sukhpreet. *Optical character recognition techniques: a survey.* Journal of Emerging Trends in Computing and Information Sciences [online]. 2013, 4(6), 545-550 [cit. 28. 10. 2016]. Dostupné z URL: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.676.9685&rep=rep1&type=pdf>.

[12] *K-NN.* RapidMiner Documentation [online]. 2017 [cit. 2017-05-29]. Dostupné z URL: <https://docs.rapidminer.com/studio/operators/modeling/predictive/lazy/k_nn.html>.

[13] *Support Vector Machine.* RapidMiner Documentation [online]. 2017 [cit. 2017-05-29]. Dostupné z URL: <https://docs.rapidminer.com/studio/operators/modeling/predictive/support_vector_machines/support_vector_machine.html>.

[14] MARJANOVIĆ, Miloš, Miloš KOVAČEVIĆ, Branislav BAJAT a Vít VOŽENÍLEK. *Landslide susceptibility assessment using SVM machine learning algorithm.* Engineering Geology [online]. 2011, 3(123), 225-234 [cit. 2017-05-29]. DOI: 10.1016/j.enggeo.2011.09.006. ISBN 10.1016/j.enggeo.2011.09.006. Dostupné z URL: <http://linkinghub.elsevier.com/retrieve/pii/S0013795211002195>.

[15] *SVM-Planes.* Dni-institute: Building Predictive Model using SVM [online]. 2015 [cit. 2017-05-29]. Dostupné z URL: <http://dni-institute.in/blogs/wp-content/uploads/2015/09/SVM-Planes.png>.

[16] *Neural Net.* RapidMiner Documentation [online]. 2017 [cit. 2017-05-29]. Dostupné z URL: <https://docs.rapidminer.com/studio/operators/modeling/predictive/neural_nets/neural_net.html>.

[17] *A 3-layer neural network with three inputs, two hidden layers of 4 neurons each and one output layer.* Cs231n: neural-networks-1 [online]. [cit. 2016-11-16]. Dostupné z URL: <http://cs231n.github.io/neural-networks-1/>.

[18] *A neuron with p inputs and a nonlinear activation function.* Scientific & Academic Publishing [online]. [cit. 2016-11-16]. Dostupné z URL: <http://article.sapub.org/10.5923.j.control.20130303.03.html>.

[19] SMITH, Ray. *An overview of the Tesseract OCR engine.* [online]. 2007 [cit. 19. 10. 2016]. Dostupné z URL: <https://static.googleusercontent.com/media/research.google.com/sk//pubs/archive/33418.pdf>.

[20] PATEL, Chirag; PATEL, Atul; PATEL, Dharmendra. *Optical character recognition by open source OCR tool tesseract: A case study.* International Journal of Computer Applications, 2012, 55.10. [online] 2012 s. 50-56. [cit. 19. 10. 2016]. Dostupné z URL: <https://www.researchgate.net/profile/Chirag_Patel27/publication/235956427_Optical_Character_Recognition_by_Open_source_OCR_Tool_Tesseract_A_Case_Study/links/00463516fa43a64739000000.pdf>.

[21] RAKSHIT, Sandip, a iní. *Recognition of handwritten Roman Numerals using Tesseract open source OCR engine.* Proc. Int. Conf. on Advances in Computer Vision and Information Technology (2009)[online] 2009 s. 572-577. [cit. 19. 10. 2016]. Dostupné z URL: <https://arxiv.org/ftp/arxiv/papers/1003/1003.5898.pdf>.

[22] *Tesseract Release Notes. Github.com* [online]. 2016 [cit. 19. 10. 2016]. Dostupné z URL: <https://github.com/tesseract-ocr/tesseract/wiki/ReleaseNotes>

[23] HU, Qingmao; QIAN, Guoyu; NOWINSKI, Wieslaw L. *Fast connected-component labelling in three-dimensional binary images based on iterative recursion* [online]. 2005 s. 414-434. [cit. 09. 10. 2016]. Dostupné z URL: <http://ac.els-cdn.com/S1077314205000378/1-s2.0-S1077314205000378-main.pdf?_tid=2242c476-a4d3-11e6-a8fe-00000aab0f6b&acdnat=1478513920_30aceb9e984befa8c35272b1f69c4765>.

[24] DHIMAN, Shivani; SINGH, A. *Tesseract Vs Gocr, a comparative study.* International Journal of Recent Technology and Engineering (IJRTE).[online] 2013, 2(4), 80-83. ISSN 2277-387. [cit. 19. 10. 2016]. Dostupné z URL: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.676.9685&rep=rep1&type=pdf>.

[25] *RapidMiner 7.3 Data Science Platform Fact Shheet.* RapidMiner [online]. 2016 [cit. 2017-05-28]. Dostupné z URL: <https:

//1xltkxylmzx3z8gd647akcdvov-wpengine.netdna-ssl.com/wp-content/uploads/2016/12/rm-platform-fact-sheet-12-2.pdf>.

[26] ELAGOUNI, Khaoula, a iní. *Combining multi-scale character recognition and linguistic knowledge for natural scene text OCR.* Document Analysis Systems (DAS), 2012 10th IAPR International Workshop [online]. 2012, 120-124 [cit. 10. 12. 2016]. Dostupné z URL: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6195347>.

[27] *IAM Handwriting Database.* Research Group on Computer Vision and Artificial Intelligence [online]. 2016 [cit. 2017-05-28]. Dostupné z URL: <http://www.fki.inf.unibe.ch/databases/iam-handwriting-database>.

[28] MARTI, U.-V.; BUNKE, Horst. *The IAM-database: an English sentence database for offline handwriting recognition.* International Journal on Document Analysis and Recognition [online]. 2002, 5(1), 39-46 [cit. 2016-11-08]. Dostupné z URL: <https://link.springer.com/article/10.1007/s100320200071>.

[29] NATARAJAN, Prem, a iní. *Multi-lingual offline handwriting recognition using hidden Markov models: A script-independent approach.* Arabic and Chinese Handwriting Recognition [online]. 2008, 231-250 [cit. 28. 10. 2016]. Dostupné z URL: <https://link.springer.com/chapter/10.1007/978-3-540-78199-8_14>.

[30] MARGNER, Volker; EL ABED, Haikal. *Arabic handwriting recognition competition.* Frontiers in Handwriting Recognition (ICFHR), International Conference [online]. 2010, 709-714 [cit. 10. 12. 2016]. Dostupné z URL: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5693647>.

[31] SARDAR, Shuwair; WAHAB, Abdul. *Optical character recognition system for Urdu.* Information and Emerging Technologies (ICIET), 2010 International Conference [online]. 2012, 1-5 [cit. 10. 12. 2016]. Dostupné z URL: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5625694>.

ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

ANN	Artificial Neural Network – umelá neurónová sieť
ASCII	American Standard Code for Information Interchange – americký štandardizovný kód pre výmenu informácií
BHMM	Bernoulli Hidden Markov Model – Bernoulliho skrytý markov model
CER	Character Error Rate – miera chybných znakov
HMM	Hidden Markov Model – skrytý markov model
HWR	HandWriting Recognition – rozpoznávanie rukopisu
IBAN	International Bank Account Number – medzinárodný formát bankového účtu
k-NN	k–Nearest Neighbors – k-najbližších susedov
MLP	Multi Layer Perceptron – viac vrstvový perceptron
OCR	Optical Character Recognition – optické rozpoznávanie znakov
SVM	Support Vector Machine – metóda podporných vektorov
WER	Word Error Rate – miera chybných slov

ZOZNAM PRÍLOH

A OBSAH PRILOŽENÉHO CD

A.1 Elektronická forma práce

A.2 Zdrojové súbory vytvorených metód