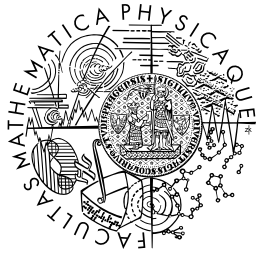


MATEMATICKO-FYZIKÁLNÍ
FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Lukáš Salak

Detekcia anonymizovaných častí v zmluvách

Katedra softwaru a výuky informatiky

Vedoucí bakalářské práce:	doc. RNDr. Elena Šikudová, Ph.D.
Studijní program:	Informatika
Studijní obor:	Počítačová grafika, vidění a vývoj her

Praha 2024

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ....

Podpis autora

Dedikácia. Nesmierne si cením podpory a pomoci, ktorú som dostal od mnohých ľudí, menovite od mojej rodiny, mojej priateľky, priateľov, spolužiakov a kolegov. Najviac však by som chcel poďakovať vedúcej mojej bakalárskej práce, doc. RNDr. Elene Šikudovej, Ph.D., ktorá mi pomohla pri všetkom, čo som potreboval.

Název práce: Detekcia anonymizovaných častí v zmluvách

Autor: Lukáš Salak

Katedra: Katedra softwaru a výuky informatiky

Vedoucí bakalářské práce: doc. RNDr. Elena Šikudová, Ph.D., Katedra softwaru a výuky informatiky

Abstrakt: Práca skúma problém detegovania anonymizovaných častí v PDF dokumentoch. Preskúmané boli rôzne prístupy na detekciu, primárne analýza obrazu a s ňou spojené rôzne algoritmy počítačového videnia. Najlepší z týchto prístupov sme implementovali a vyhodnotili na testovacích dátach. Výsledky ukázali, že implementovaný prístup dosiahol vysokú presnosť a predbehol iné prístupy aj vzhľadom na efektivitu. Tento výskum prispieva k rozvoju nástrojov pomáhajúcich analyzovať dokumenty, ktoré môžu byť aplikované v rôznych právnych či finančných oblastiach na zaručenie ochrany dát v súlade s reguláciami.

Klíčová slova: PDF, Segmentace, Detekce

Title: Detection of anonymized parts in PDFs

Author: Lukáš Salak

Department: Department of Software and Computer Science Education

Supervisor: doc. RNDr. Elena Šikudová, Ph.D., Department of Software and Computer Science Education

Abstract: The work examines the problem of detecting anonymized parts in PDF documents. Various detection approaches, primarily image analysis and related computer vision algorithms, were explored. We implemented and evaluated the best of these approaches on test data. The results showed that the implemented approach achieved high accuracy and outperformed other approaches also in terms of efficiency. This research contributes to the development of tools to help analyze documents that can be applied in various legal or financial areas to guarantee data protection in accordance with regulations.

Keywords: PDF, Segmentation, Detection

Obsah

Úvod

Problém anonymizácie dát je dôležitý v rôznych oblastiach, napríklad v oblasti verejnej správy či v oblasti marketingu. Pod pojmom anonymizácia dokumentov si môžeme predstaviť vymazanie či skrytie údajov alebo iných citlivých informácií. Možností, ako pristupovať k anonymizácii dokumentov, je veľa.

Typickým miestom, kde sa stretávame s anonymizáciou dokumentov, je oblasť verejnej správy. V Českej republike majú organizácie verejnej správy povinnosť zverejňovať informácie o svojej činnosti, k čomu patrí aj zverejňovanie uzavretých zmlúv nad určitú čiastku do *registra zmlúv*, ktorý je verejne prístupný. Nachádzajú sa tu nielen informácie o predmete zmlúv, zmluvných stranách a cene, ale takisto všetky súbory, ktoré sú súčasťou zmlúv. Register zmlúv je významným nástrojom, ktorý zlepšuje transparentnosť; podstatou je kontrolovať a mať možnosť obmedziť korupciu a zneužívanie verejnej moci kvôli uzatváraniu nevýhodných zmlúv.

Aj napriek tomu, že zverejňovanie dát v registri zmlúv je právne vynútiteľné, nezabezpečuje to automaticky možnosť jednoduchého vyhľadávania či analýzy týchto dát. K tomu bol vytvorený projekt, webový portál *Hlídač smluv*, ktorý má za úlohu zlepšiť prístup k registru zmlúv. Neskôr, po skombinovaní ďalších verejne prístupných dát z registrov a databáz, sa vytvoril projekt Hlídač státu [1], ktorý má za úlohu zlepšiť prístup k verejným informáciám. Poskytuje napríklad plnohodnotné vyhľadávanie v texte zmlúv.

V registri zmlúv sú dokumenty z rôznych oblastí, napríklad z oblasti zdravotníctva, školstva, realitných služieb alebo IT projektov. V prípade, že dokumenty obsahujú citlivé údaje, sú častokrát anonymizované. V súčasnej dobe neexistuje štatistický nástroj, ktorý by znázorňoval koľko percent v takýchto dokumentoch je zanonymizovaných.

Cieľom práce je zaoberať sa anonymizovanými PDF dokumentmi a vytvorenie nástroja, ktorý bude schopný detegovať anonymizované časti dokumentu, využiť grafické metódy používané pri počítačovom videní a ďalších algoritmov na spracovanie obrazu a navrhnúť tak systém, ktorý umožní na základe dostupných dát vyhodnotiť percento anonymizácie jednotlivých zmlúv pri použití konkrétnych implementačných metód a následne nasadiť túto implementáciu na webový portál Hlídače státu.

Hlavným prínosom práce je vytvorenie systému na porovnávanie jednotlivých odvetví, ktoré zverejňujú zmluvy vzhľadom na percento anonymizácie a tvorba štatistiky vzhľadom na anonymizáciu dokumentov relatívne k jednotlivým oblastiam.

Práca je štruktúrovaná nasledovne: 1. kapitola rieši definíciu anonymizácie, právne aspekty a dôvody pre anonymizáciu a možné metódy a techniky, ktorými sa dokumenty anonymizujú. V 2. kapitole je špecifikovaný konkrétny problém a zadanie, ktorému sa v práci venujeme. 3. kapitola je venovaná popisu vstupných dát, ich obsah a štruktúra. Takisto je tu popísaný proces získavania dát a ich príprava na ďalšie spracovanie a extrakcia relevantných informácií. V 4. kapitole je popísaný proces detekcie anonymizovaných častí dokumentov. V kapitole 5 je popísaný algoritmus, ktorý v systéme používame. Kapitola 6 je venovaná implementácii riešenia, použité technológie a nástroje, architektúra a dizajn systému, prípadové štúdie a ďalšie detaily implementácie. V závere kapitoly 6.1.1 zhŕňame mimo dosiahnutých výsledkov aj osobné zistenia a odporúčania pre ďalší výskum.

Kapitola 1

Anonymizácia dokumentov

1.1 Základné pojmy

1.1.1 Definícia anonymizácie

Anonymizácia je technika ochrany údajov, ktorá zahŕňa odstránenie všetkých identifikačných informácií z osobných údajov tak, aby údaje nebolo možné spojiť s jednotlivcom. Anonymizáciou sa údaje stanú úplne anonymnými a už sa nepovažujú za osobné údaje. Anonymizácia sa často používa v situáciách, keď osobné údaje už nie sú potrebné, ale údaje sa stále môžu použiť na výskumné alebo štatistické účely.[2]

1.1.2 Typy údajov na anonymizáciu

Medzi údaje, ktoré sú často predmetom anonymizácie, patria mená, adresy, telefónne čísla, a ďalšie osobné identifikátory. Niektoré údaje majú inú informačnú hodnotu, napríklad, že dátum narodenia osoby je menej cenný než rodné číslo danej osoby. V našom prípade sú anonymizovanými údajmi spravidla kupované predmety a sumy, za ktoré boli predmety kúpené (napr. pri zmluvách ministerstva obrany) a mená a podpisy osôb či firiem, ktoré tieto zmluvy uzavreli.

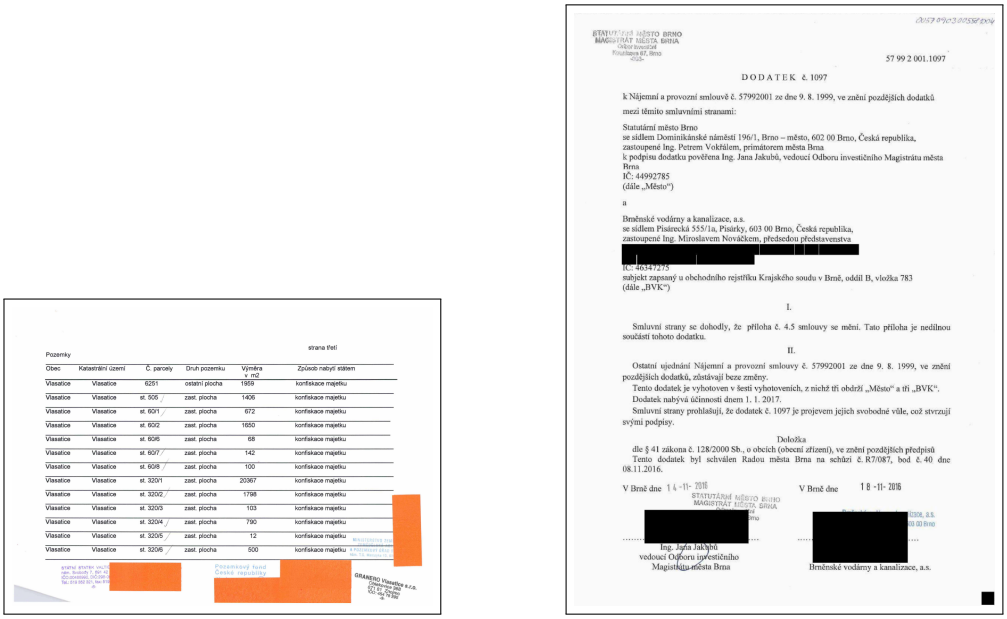
1.1.3 Právne a etické dôvody

Právne predpisy, ako napríklad GDPR v Európskej únii, a etické normy nútia organizácie anonymizovať určité typy údajov. K anonymizácii údajov v Českej republike existuje Zákon o ochrane údajov.[3] Takisto na úrovni Európskej únie existuje smernica upravujúca ochranu fyzických osôb pri spracovaní osobných údajov.[4] Vzhľadom na skúmané zmluvy a dokumenty je dôležitý zákon č. 412/2005 Sb. o ochraně utajovaných informací a o bezpečnostní způsobilosti.[5]

1.2 Metódy a techniky anonymizácie

1.2.1 Manuálne vs. digitálne metódy

V našej práci rozlišujeme medzi manuálnou a digitálnou anonymizáciou. Manuálna anonymizácia je založená na princípe ručného odstránenia či prelepenia fyzickej zmluvy (obr. 1.1 vľavo), a digitálnymi úpravami, ako je napríklad editácia PDF dokumentu pridaním čierneho obdĺžnika či zašumením pixelov v danej oblasti, ktorá má byť anonymizovaná (obr. 1.1 vpravo.)



Obr. 1.1 Porovnanie medzi manuálnou (vľavo) a digitálnou anonymizáciou (vpravo).

1.2.2 Technologické nástroje

Nástroje, ktoré umožňujú automaticky odstraňovať podpisy, časové pečiatky a iné dôverné informácie, a ich výber závisia od konkrétnych požiadaviek a objemu dokumentov, ktoré je potrebné anonymizovať. Medzi najznámejšie patrí softvér Signer od spoločnosti Software602 [6] alebo softvér Syntho [7]. Moderné nástroje ako spomínaný Syntho využívajú technológiu AI na detegovanie údajov, ktoré je potrebné anonymizovať.

Medzi kľúčové vlastnosti týchto nástrojov patrí najmä odstránenie osobných údajov, podpisov či časových pečiatok priamo z metadát dokumentov a možnosť označiť a prekryť určité časti dokumentu.

Vo verejnej správe v Českej republike je používaný nástroj, ktorý je priamo pod správou Ministerství vnitra ČR [8]. Tento nástroj umožňuje vyššie spomínané funkcionality a je jedným z najčastejších spôsobov anonymizácie dokumentov, ktoré v tejto práci riešime.

Kapitola 2

Špecifikácia problému

2.1 Identifikácia kľúčových výziev

2.1.1 Komplexita anonymizovaných dokumentov

Keďže zákon [9] presne neukladá, v akom formáte majú byť zmluvy zverejňované a neexistuje ani právna úprava, ktorá by regulovala spôsoby anonymizácie, v registri zmlúv preto nájdeme mnoho rôznych foriem a typov dokumentov. Základné rozdelenie zverejnených zmlúv je, či sú zmluvy digitálne, t. j. či sú originály zmlúv v digitálnej podobe, alebo sú zmluvy skenované z fyzických originálov (najčastejšie sken A4 dokumentov).

Z pohľadu detekcie anonymizovaných častí dokumentov je vhodnejšie analyzovať digitálne zmluvy, pretože neobsahujú artefakty spôsobené skenovaním a z dôvodu neprítomnosti šumu je preto jednoduchšie analyzovať takéto dokumenty. Pri preskenovaných fyzických originálochčastokrát dochádza k nedokonalému skenu, kedy sa pri skenovaní dokumentu nedôkladne preskenuje daný dokument. Konkrétnymi príkladmi takýchto artefaktov môžu byť rohy papierov, kde sú viacstranové zmluvy zospinkované, a teda nedôjde k dôkladnému priloženiu skenovanej predlohy na plochu skenera, alebo sa pri skenovaní stratí informácia o farbe (ak je daný dokument skenovaný do čiernobielej), čo môže spôsobiť problémy pri hľadaní začiernených plôch. Zväčša sa jedná o úradné pečiatky, logá firiem, obrázkové prílohy či záhlavia tabuliek.

2.1.2 Rozpoznanie anonymizovaných oblastí

Vzhľadom na vyššie spomínané spôsoby anonymizácie je náročné kategorizovať jednotlivé typy a na základe toho určiť pomer anonymizovaných oblastí vzhľadom na obsah dokumentu. Pri rozpoznávaní daných oblastí je dôležité mať na pamäti

túto rôznorodosť a adekvátne navrhnúť algoritmus tak, aby dokázal detegovať čo najviac techník a čo najpresnejšie určiť ich rozsah. Keďže sa v práci zameriavame na algoritmické riešenie bez použitia umelej inteligencie a machine learning, spôsob na rozpoznanie týchto oblastí je určený prevažne implementáciou rôznych algoritmov počítačového videnia, ktoré sú bližšie popísané v 4. kapitole.

2.2 Prehľad existujúcich riešení a ich obmedzenia

2.2.1 Súčasné metódy a nástroje

Vzhľadom na špecifickosť problému a rozsah uplatnenia v súčasnej dobe neexistujú plnohodnotné verejne dostupné nástroje, ktoré by boli zamerané na detekciu anonymizovaných oblastí v zmluvách. Každopádne s technológiami ako machine learning a AI je pravdepodobné, že v blízkej dobe vzniknú nástroje využívajúce práve tieto metódy na riešenie tohto typu problému.

2.3 Definovanie požiadaviek na riešenie

Žiadanými parametrami sú správnosť detekcie oblasti, kde došlo k anonymizácii, presnosť detekcie anonymizovaných oblastí a presný odhad anonymizovanej oblasti vzhľadom na obsah dokumentu. Správnosť detekcie oblasti zaručuje, že nedôjde k označeniu miesta na dokumente ako anonymizované, keď v skutočnosti nie je, napríklad záhlavie tabuľky, obrázok, zvýraznený text, pečiatka či logo. Pod presnosťou rozumieme čo najpresnejšie ohraničenie anonymizácie. Napríklad, ak sa jedná o prelepenie nálepkou alebo ak oblasť anonymizácie hraničí s textom, chceme, aby sme ohraničili oblasť presne celú bez čo najmenšej aproximácie. Presný odhad anonymizovanej oblasti je požiadavka vzťahujúca sa na možný odhad toho, koľko daná oblasť zakrýva. V niektorých prípadoch, kedy je zamazaný celý riadok alebo nejaká časť riadku, vieme odhadnúť, koľko znakov, respektíve percent vzhľadom na celý text je prekrytých. Ak je prekrytých viac riadkov, je zložitejšie určiť percento prekrytia vzhľadom na to, že text nemusí byť formálne rozložený (napr. koniec odseku, vynechaný riadok alebo viac riadkov). Čím viac plochy je prekrytej, tým ťažšie je odhadnúť, koľko údajov bolo anonymizovaných.

Z užívateľského hľadiska je požadovaná okrem spomínaných požiadaviek aj jednoduchosť používania softvéru a rýchlosť analýzy daného dokumentu, resp. dokumentov. Požadujeme, aby systém dokázal spracúvať väčšie množstvo dokumentov v rýchlom čase a v prípade chyby (problém pri sťahovaní či otváraní dokumentu a pod.) pokračoval bez obmedzení a o chybe uživateľa informoval.

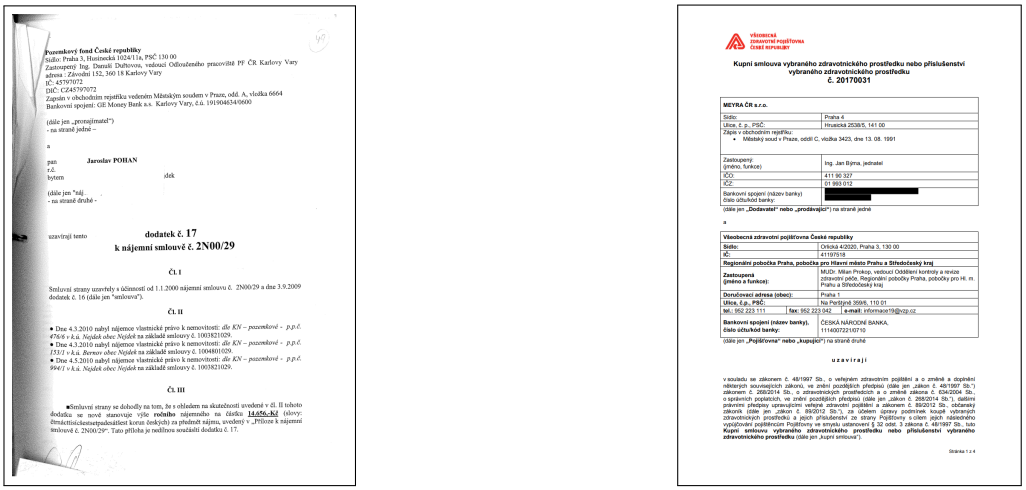
Kapitola 3

Popis vstupných dát a ich spracovanie

3.1 Charakteristika vstupných PDF dokumentov

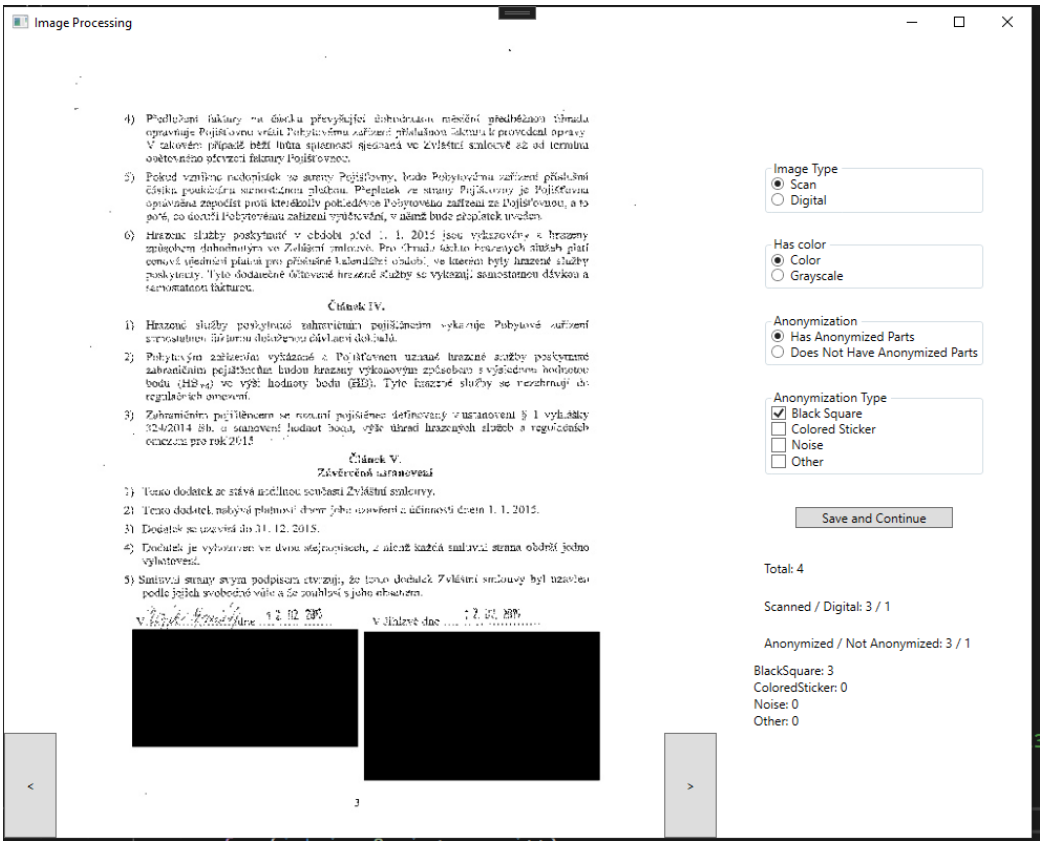
3.1.1 Typy PDF dokumentov

Ako už bolo spomenuté v predchádzajúcej kapitole, typy PDF dokumentov, ktorým sa v práci venujeme, sú dva; prvým typom sú skenované fyzické originály dokumentov, druhým sú digitálne dokumenty. Zásadným rozdielom je šum pri skenovaných dokumentoch, ktorý sa v digitálnych dokumentoch nenachádza. Môže sa napríklad jednať o nedokonalosti spôsobené nesprávnym priložením papierovej predlohy na skener (obr. 3.1 vľavo). Vpravo je ukážka digitálnej zmluvy.



Obr. 3.1 Porovnanie rozdielu medzi skenovaným (vľavo) a digitálnym dokumentom (vpravo).

K analýze vstupných dát sme si pripravili jednoduchý program (obr. 3.2) (viac v podkapitole 6.3.1), ktorý priebežne z desaťtisíc odkazov na stiahnutie zmlúv, ktoré nám boli na vyžiadanie zaslané z portálu `https://smlouvy.gov.cz`[10], sťahoval dokumenty a zobrazoval jednotlivé stránky.



Obr. 3.2 Grafické rozhranie skriptu `ManualCheckerUtility` (6.3.1).

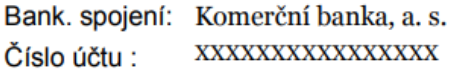
Následne sme pomocou tejto aplikácie vytvorili štatistiku nad týmito dokumentmi, kde sme zistili nasledovné:

Tabuľka 3.1 Štatistika nad manuálne skúmanými PDF dokumentmi.

Celkový počet:	161
Sken / Digitál :	129 / 32
Anonymizované / Neanonymizované:	122 / 39
Čierny obdĺžnik:	118
Farebná nálepka:	1
Šum:	2
Ostatné:	1

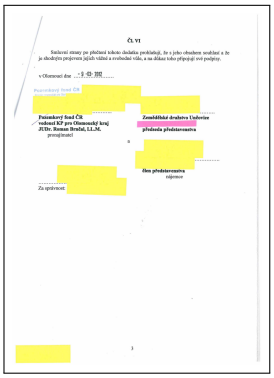
Príklady typov anonymizácií, ktoré sme manuálnym prehľadávaním našli:

Tabuľka 3.2 Typy anonymizácií, ktoré sme manuálnym prehľadávaním našli.

Typ	Obrázok
Šum	
Čierny obdĺžnik	
Ostatné	
Farebná nálepka	

3.1.2 Typické anonymizované oblasti

Vzhľadom na to, že dokumenty pred anonymizáciou nie sú dostupné, môžeme len s pravdepodobnosťou na základe logického uváženia určovať, čo bolo predmetom anonymizovania. Zväčša sa jednalo o začiernenie podpisov, mien (obr. 3.3 vľavo) a rodných čísel, vzhľadom na kontext okolitého textu sa vyskytlo niekoľkokrát aj zamazanie názvov firiem, čísel účtov či súm, ak sa jednalo o kúpne či predajné zmluvy a v niekoľkých špecifických prípadoch boli prekryté čiernym štvorcom celé strany, takže nebolo možné identifikovať ani to, čo bolo obsahom, resp. predmetom zmluvy (obr. 3.3 vpravo).



Obr. 3.3 Príklady anonymizovaných oblastí, vľavo bežný výskyt, vpravo výskyt, kde bola začiernená celá strana.

Keďže len zhruba 25 % dokumentov, nad ktorými sme urobili štatistiku, obsahovalo anonymizované dáta, usudzujeme, že tieto údaje sú najčastejšími údajmi, ktoré sú anonymizované.

Vzhľadom na prístup k detekcii anonymizovaných oblastí sa nebudeme zaoberať detekciou typu "Ostatné" znázornenej v tabuľke 3.2.

3.2 Predspracovanie dát

3.2.1 Predpríprava dokumentov

Keďže jednotlivé dokumenty spravidla obsahujú viac strán, je nutné analyzovať každú stranu. Z tohoto dôvodu sme sa rozhodli zo vstupných PDF dokumentov vytvárať obrázky jednotlivých strán pomocou knižnice `MagickImage` a ukladať ich ako `byte[]`. Týmto rozhodnutím síce prídeme o informácie a metadata z digitálnych originálov PDF dokumentov, každopádne vzhľadom na to, že majorita dokumentov (tabuľka 3.1) sú skeny a typ anonymizácie je spravidla čierny obdĺžnik (tabuľka 3.1, ktorý je relatívne jednoducho detegovaný, je jednoduchšie pracovať s jedným formátom dokumentu, a to vo forme obrázku jednotlivej strany. Tento formát je pre nás najjednoduchší na spracovanie pomocou algoritmov počítačového videnia.

Obrázky sme ďalej needitovali, pre ďalšie spracovanie sme použili obrázky o veľkosti približne 850x600 pixelov (čo zodpovedá rozmerom štandardného papiera ISO A[11]), resp. 600x850 pixelov ak sa jednalo o formát na šírku.

3.2.2 Výzvy a riešenia

Jednou z výziev pri analýze dokumentov je rôznorodá kvalita skenov. S týmto sa spája viac problémov, ktoré môžu nastať, zväčša sa jedná o spomínaný šum, ktorý vznikne nedokonalým priložením predlohy na skener. Problém so šumom sme riešili pomocou filtračných techník na zníženie šumu v obrázkoch.

Ďalším z problémov je zarovnanie strany. Pri skenovaní sa môže stať, že výsledný sken nebude zarovnaný. Orientácia strany je dôležitá pri výpočte pomeru anonymizovaných oblastí vzhľadom k celej strane dokumentu. Orientáciu strán sme v našom algoritme ani v predpríprave dokumentov neriešili, nakoľko všetky testované dokumenty boli spravidla zarovnané a len v pár prípadoch boli skeny vychýlené natoľko, že by to mohlo závažne ovplyvniť výsledné percento.

Kapitola 4

Proces detekcie anonymizovaných častí dokumentov

4.1 Algoritmy a techniky detekcie

Všetky popisy algoritmov v sekcii 4.1.1 sú citované z knihy *Počítačové videnie Detekcia a rozpoznávanie objektov*[12]. V prípade, že citujeme z iného zdroja, bude tento zdroj explicitne referencovaný.

4.1.1 Použité algoritmy

Celý priebeh detekcie anonymizovaných oblastí je zložený na snahe redukovať čo naviac šumu a vyčistiť obraz tak, aby bolo jendoduché využiť za pomoci prahovania morfologické operácie dilatáciu a eróziu.

Morfologické operácie

"*Morfologické spracovanie obrazu využíva informáciu o susedných pixeloch v topologickom okolí spracovávaného pixela.*" Pre správne pochopenie fungovania týchto morfologických operácií je dôležité najprv zadefinovať tzv. štrukturálny element. Predtým však ešte spomenieme Minkowského sumu.

Definícia 1. *Minkowského suma*
Minkowského suma bodových množín A a B je bodová množina

$$C = \bigcup_{b \in B} A_b$$

kde

A_b je množina A posunutá o vektor b, teda množina $A_b = a + b|a \in A$. $\bigcup$ označuje zjednotenie (union) množín. Minkowského sumu množín A a B označujeme

$$C = A \oplus B$$

Štrukturálny element

Štrukturálny element S je bodová množina s veľkosťou menšou než vstupný obraz. Jeho typický rozmer býva pomerne malý, napr. $3 \times 3, 5 \times 5, \ldots 21 \times 21$ atď. V Minkowského sume je vstupný obraz označený A a štrukturálny element je tu označený ako B.

Morfologickú transformáciu je možné definovať ako matematickú reláciu medzi dvomi bodovými množinami, a to množinou obrazu a množinou použitého štrukturálneho elementu.

Definícia 2. *Dilatácia*
Operáciu binárnej dilatácie spracovávaného obrazu A a štrukturálneho elementu S značíme

$$A \oplus S.$$

Binárna dilatácia vyplýva z uvedenej definície Minkowského sumy zjednotenia posunutých bodových množín A a S. Binárnu dilatáciu môžeme zapísať ako

$$A \oplus S = \bigcup_{s \in S} A_s,$$

kde A_s je množina A posunutá o vektor s, teda množina

$$A_s = a + s|a \in A.$$

Definícia 3. *Erózia*
Operáciu erózie spracovávaného obrazu A a štrukturálneho elementu S značíme

$$A \ominus S.$$

Binárnu eróziu môžeme zapísať ako prienik všetkých posunov obrazu A o vektory $-s$, kde $s \in S$:

$$A \ominus S = \bigcap_{s \in S} A_s,$$

kde A_s je množina A posunutá o vektor s, teda množina

$$A_s = a + s|a \in A.$$

Obr. 4.1 Vľavo je vstupný obraz, vpravo dilatovaný výstupný obraz.

Obr. 4.2 Vľavo je vstupný obraz, vpravo erodovaný výstupný obraz.

Morfologické otvorenie a uzavretie

Nosnými operáciami použitého algoritmu na detegovanie anonymizovaných oblastí sú využitia kombinácií dilatácie a erózie.

Morfologické otvorenie

Pod morfologickým otvorením rozumieme použitie erózie a následne dilatácie. Túto operáciu môžeme zapísať ako

$$(A \ominus S) \oplus S.$$

Morfologické uzavretie

Pod morfologickým uzavretím rozumieme použitie dilatácie a následne erózie, teda v opačnom poradí. Túto operáciu môžeme zapísať ako

$$(A \oplus S) \ominus S.$$

Obr. 4.3 Vľavo je vstupný, vpravo výstupný obraz po operácii morfologického otvorenia.

Obr. 4.4 Vľavo je vstupný, vpravo výstupný obraz po operácii morfologického uzavretia.

Všimnime si, aký efekt má použitie otvorenia, resp. uzavretia na vstupný obraz. Pri morfologickom otvorení na obrázku 4.3 vidíme, že sú odstránené malé izolované oblasti, resp. oblasti, ktoré sú tenké a menšie než použitý štrukturálny element. Naopak, pri uzavretí (obr. 4.4) vidíme "spájanie" izolovaných častí, ktoré sú dostatočne blízko seba, aby boli pokryté štrukturálnym elementom. Výsledný obraz je teda veľmi ovplyvnený tým, aký tvar, resp. aký veľký je použitý štrukturálny element.

Prahovanie

Prahovanie je jednoduchý koncept, kde na základe určených parametrov ohraničíme intenzitu jednotlivých pixelov. V prípade šedotónového obrazu, kde má každý pixel hodnotu od 0 do 255, ak použijeme prahovanie, napr. horný prah 150 a spodný prah 20, všetky pixely vo výslednom obraze budú v tomto rozmedzí, a teda pixely s hodnotou nižšou než 20 sa nastavia na 20 a naopak, pixely s hodnotou vyššou než 150 sa nastavia na 150.

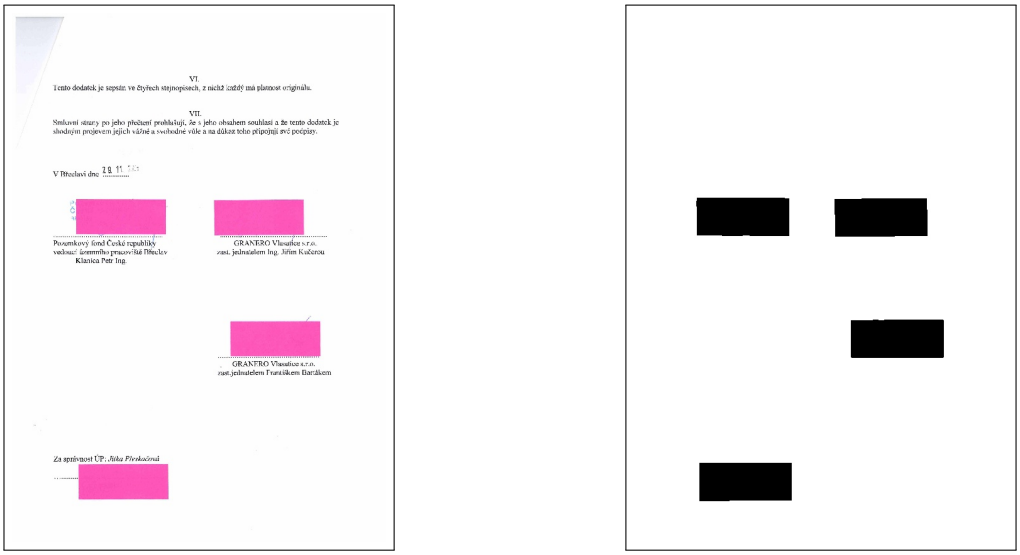
Na prahovanie využívame OTSU algoritmus[13], ktorý je založený na princípe hľadania najlepšej hodnoty prahu na rozdelenie obrázka na tmavé a svetlé časti tak, aby bol rozdiel medzi nimi čo najväčší.

Na filtráciu šumu a detekciu hľadaných oblastí sme vyskúšali aj tzv. korekciu neuniformného osvetlenia[14], každopádne výsledky boli na našich testovacích dátach zhodné s použitím kombinácie vyššie spomenutých morfologických operácií.

4.1.2 Zvolená kombinácia algoritmov

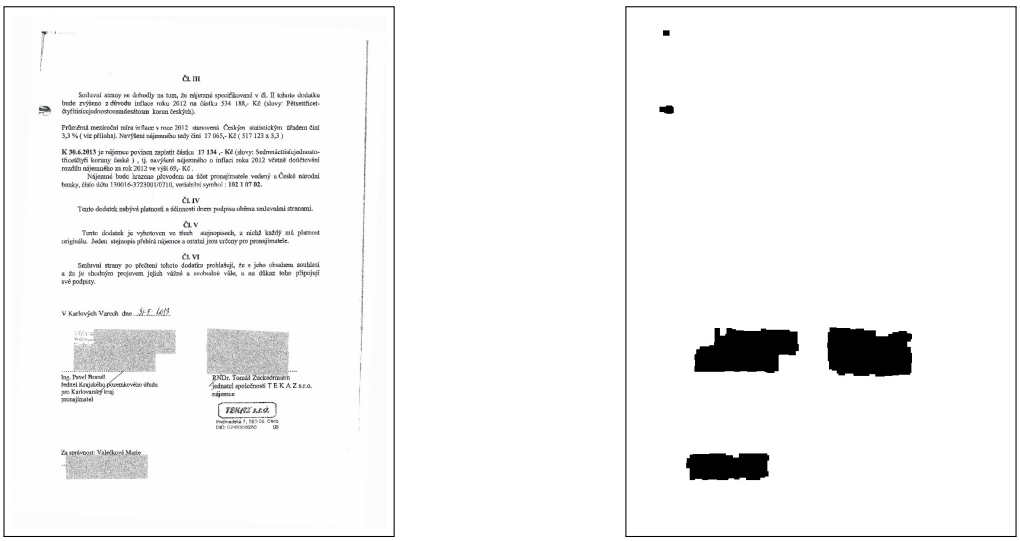
Po niekoľkých iteráciách, ktoré sú popísané v ďalšej podkapitole, sme pristúpili na finálny algoritmus pozostávajúci z prahovania a následnej kombinácie viacnásobného použitia dilatácie a erózie. To nám umožnilo efektívne odstrániť šum a text, ktorý sa v dokumentoch vyskytoval a výsledným obrazom boli anonymizované oblasti. V prípade, že bola detegovaná farba v obrázku, pracovali sme s variantom, že sa jedná o typ anonymizácie farebnou nálepkou a týmto farebným pixelom sme zmenili farbu na fialovú a zvýšili saturáciu tak, aby boli jednoznačne identifikovateľné. Fialová farba bola zvolená z dôvodu, že sa v skúmaných dokumentoch táto farba nikde nevyskytovala.

Celý algoritmus je zložený nasledovne: V prvom kroku zistíme či je obraz šedotónový alebo obsahuje nejaké farebné pixely. Ak obraz má nejaké farebné pixely, tieto pixely saturujeme. Následne obraz dilatujeme východzím štvorcovým štrukturálnym elementom o veľkosti 3×3 pixely z knižnice OpenCVSharp[15]. Po dilatácii obraz prahujeme a následne opakovane dilatujeme. Vďaka viacnásobnej dilatácii sa nám spoja izolované pixely, vďaka čomu sme schopní následne viacnásobne erodovať, čím odstraníme šum a aj väčšinu textu. Po viacnásobnej erózii znovu dilatujeme, aby sme vykompenzovali stratu pixelov na hranách oblastí, ktoré chceme detegovať, pre lepší výpočet pomeru oblastí vzhľadom na celú stranu. Výsledkom je obraz, ktorý obsahuje len hľadané anonymizované oblasti (obr. 4.5).



Obr. 4.5 Vľavo vstupný obraz, vpravo výstupný.

Tento algoritmus je schopný detegovať aj typ anonymizácie, kde je použitý šum, avšak nie úplne dokonale (obr. 4.6).



Obr. 4.6 Vľavo vstupný obraz, vpravo výstupný.

4.1.3 Výpočet anonymizácie

Na záver vypočítavame celkové percento pokrytia anonymizovanej oblasti relatívne k obsahu strany. Keďže neexistuje jednoznačne najlepšia metrika, podľa čoho takéto percento počítať, rozhodli sme sa, že použijeme nasledovný výpočet:

V originálnom obraze spočítame nenulové pixely, teda len pixely, ktoré nesú nejakú informáciu (časť písmena textu, tabuľka. . .) a touto hodnotou predelíme počet pixelov, ktoré sme na analyzovanej strane detegovali ako anonymizované oblasti, prenásobené koeficientom, keďže je zjavné, že nie celá prekrytá časť obsahuje relevantné informácie.

Pretože informácie prekryté anonymizovanou oblasťou sú dôležité, je potrebná ich kompenzácia vhodným koeficientom. Pri analýze a testovaní sme zvolili ako najvhodnejšiu hodnotu kompenzačného koeficientu 0.83. Výsledná hodnota je v rozsahu od 0 do 1, a teda percento údajov, ktoré boli anonymizované.

Väčšinou sa však jedná len o podpisy alebo mená, preto sa výsledné hodnoty pohybujú zväčša v rozmedzí od 0.01 do 0.10. Napríklad výsledná hodnota pre vyššie zobrazené výstupy (obr. 4.5) je 0.064, a teda 6,4%, (obr. 4.6) je to 0.042, resp. 4,2%.

4.2 Validácia a testovanie

4.2.1 Testovacie stratégie

Keďže sme nemali k dispozícii referenčné riešenie, oproti ktorému by sme mohli porovnať naše výsledky, zvolili sme stratégiu, ktorá zarhňovala manuálne porovnávanie a empirické odhady správnosti. Výsledky sme porovnávali v rámci vybraného datasetu siedmich dokumentov, ktorý obsahoval ako skenované, tak digitálne dokumenty spoločne so všetkými nájdenými typmi anonymizácie.

Na generáciu výsledkov bol vytvorený projekt `GenerateTestResults`, vďaka ktorému sme mohli pozorovať zmeny vzhľadom na jednotlivé úpravy. Mimo kontroly správnosti samotných výsledkov sme aplikáciu pokryli unit a integračnými testami, aby sme zaručili správnosť a funkčnosť aj v prípade nevalidných dokumentov či zlyhania siete, v prípade sťahovania dokumentov cez internet.

4.2.2 Analýza výsledkov a iteratívne zlepšovanie

Prínosom v práci bola možnosť iteratívne skúmať rôzne kombinácie skladania morfologických operácií, vďaka čomu sme boli schopní prísť s najlepšou možnou kombináciou v rámci relatívne objektívneho hodnotenia detekcie oblastí.

Kapitola 5

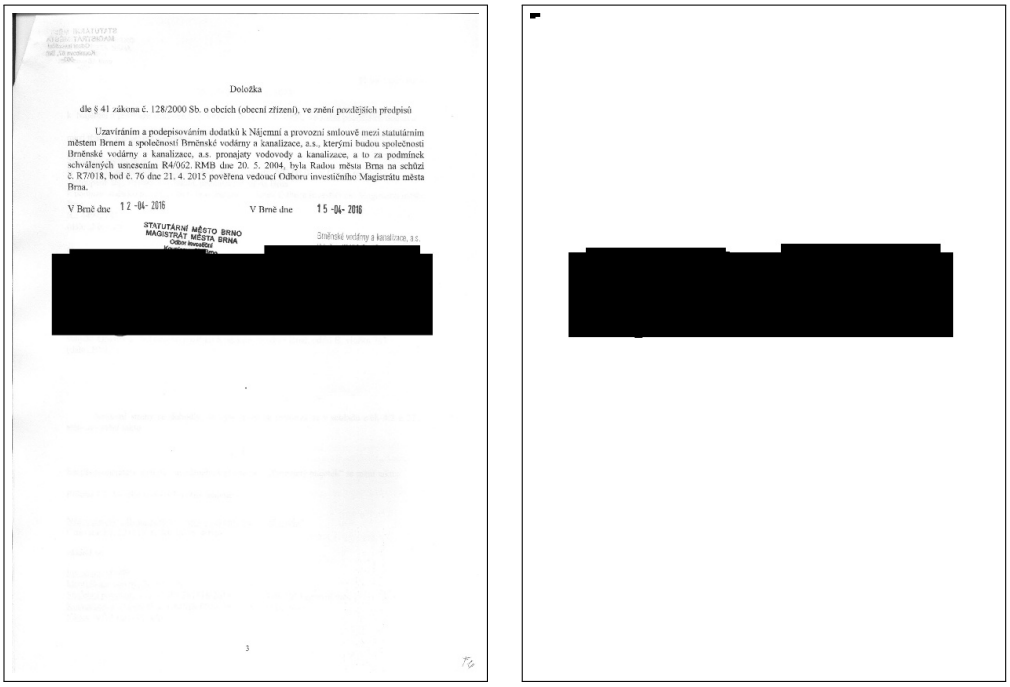
Vizualizácia výsledkov

Keďže obsahom práce je niečo, čo je veľmi jednoducho vizualizovateľné, a to jednoduchým porovnaním vstupov a výstupov, v ďalšej podkapitole je uvedených niekoľko príkladov. Uvedieme príklady z testovacieho datasetu a aj niekoľko iných dokumentov, ktoré pouvažujeme za zaujímavé na zhodnotenie. Ku každému príkladu uvedieme komentár.

5.1 Interpretácia získaných štatistík

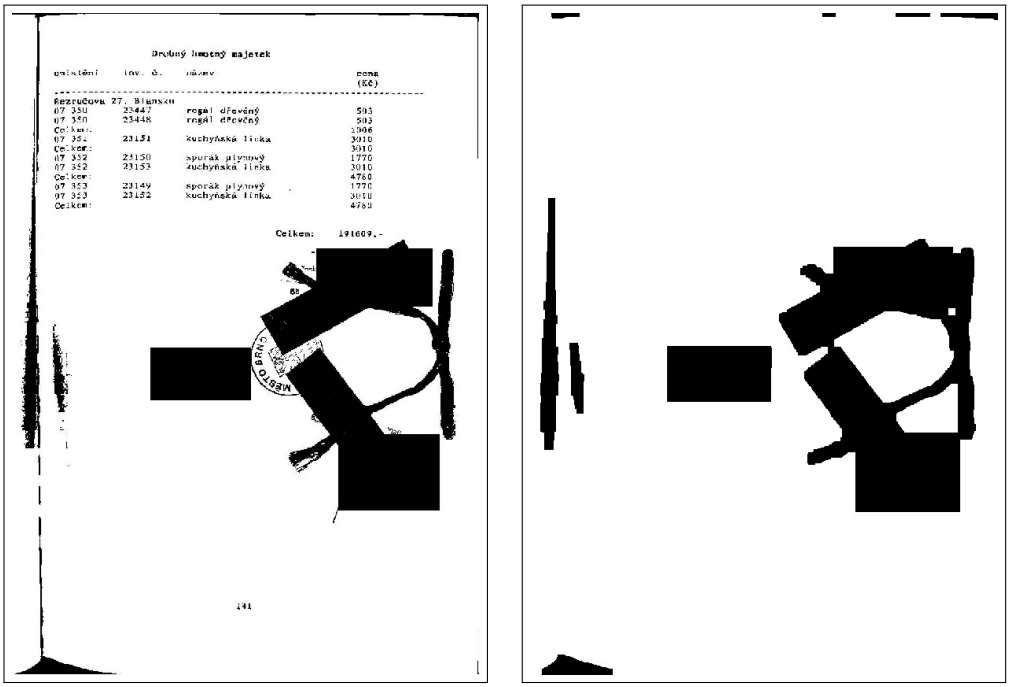
Dosiahnuté výstupné detegované oblasti prezentujeme s dôrazom na to, ako tieto dáta ovplyvňujú celkové pochopenie problému a efektivitu riešenia. Pre prehľadnosť uvedieme len vybrané strany dokumentov, nakoľko väčšina dokumentov je viacstranová a nie na každej strane sa objavuje anonymizovaná oblasť. Vľavo budeme uvádzať vstup, vpravo výstup.

Ako prvé uvedieme príklady, kde detekcia funguje správne podľa očakávaní. Na ďalšej strane na obrázku 5.1 môžeme vidieť, že na dokumentoch, kde bol použitý typ anonymizácie čierny obdĺžnik, náš algoritmus presne deteguje tieto miesta.



Obr. 5.1 Percento anonymizovania : 10,61 %

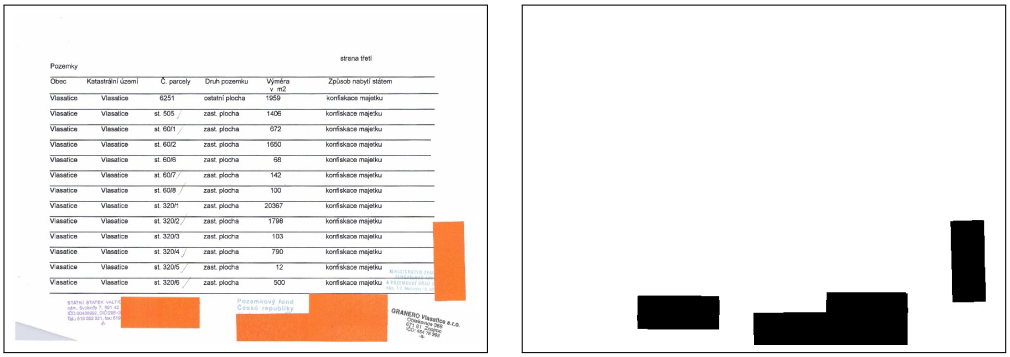
Za zmienku tu stojí všimnúť si ľavý horný roh, kde sa vyskytuje relatívne tmavý trojuholník, ktorý vznikol nedokonalým priložením papiera na skener. Náš algoritmus v tomto prípade správne detegoval len miesto, ktoré je naozaj začiernené. Môžeme si všimnúť komplexitu začiernenej plochy. Nejedná sa o štvoruholník, ale o viac komplexný tvar. V prípade použitia prístupu detekcie bounding boxov by sme museli počítať s tým, že by bol bounding box väčší a zahŕňal by aj miesta, ktoré začiernené nie sú. To by znamenalo viac algoritmickej práce na overenie, že bounding box naozaj obsahuje len tú plochu, ktorú má a v takomto prípade by bolo potrebné rozparcelovať bounding boxy tak, aby dokonale obkreslili nepravidelné tvary. Problém s použitím prístupu bounding boxov by bol ešte viac viditeľný, ak by bola na vstupnom obraze šikmá anonymizovaná oblasť, ako môžeme vidieť v ďalšom príklade na obr. 5.2.



Obr. 5.2 Percento anonymizovania : 12,68 %

V tomto prípade sa jedná o dokument z roku 1994 a môžeme si všimnúť, že to je nekvalitný sken. V tomto prípade je zobrazená posledná strana a vzhľadom na použitý algoritmus je vidieť, že za anonymizovanú oblasť boli detegované aj oblasti, ktoré nimi reálne nie sú. V tomto konkrétnom príklade bolo pre použitý algoritmus obtiažne rozlíšiť medzi nedokonalosťami skenu, zväzujúcou šnúrkou a najskôr nálepkami prekrývajúce dôležité informácie tohto dokumentu.

Jedným z možných riešení by bola detekcia zvislých pozdĺžnych oblastí a tie vyhodnocovať za chybu. Pri implementácii sme premýšľali nad možnosťami filtrácie, no v tej dobe sme nemohli prísť s rozumným riešením, ktoré by nemalo fatálny dopad na iné analyzované dokumenty, ako si ukážeme na ďalšej strane, na obr. 5.3.

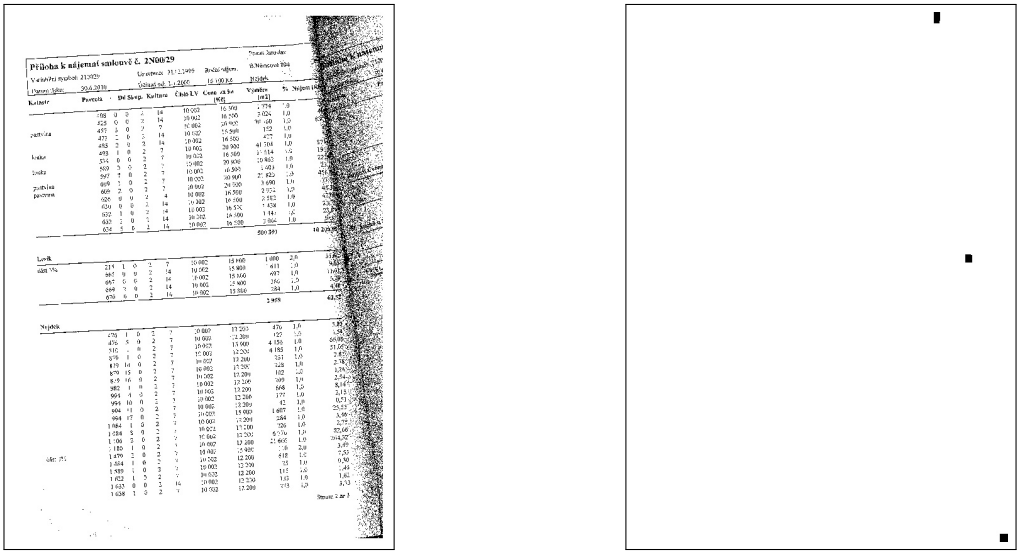


Obr. 5.3 Percento anonymizovania : 7,45 %

V tomto prípade sa jedná o prelepenie farebnými nálepkami, ktoré algoritmus rovnako dokázal detegovať a správne určiť za anonymizovanú oblasť.

Na predchádzajúcom príklade sme spomenuli možnú filtráciu pozdĺžnych nepravidelných objektov, čo by v tomto prípade znamenalo detekciu oblasti vpravo od tabuľky (všimnime si, že oblasť nie je zarovnaná a je šikmo) a po následnej filtárcii by došlo k odstráneniu oblasti, ktorá bola pôvodne detegovaná správne.

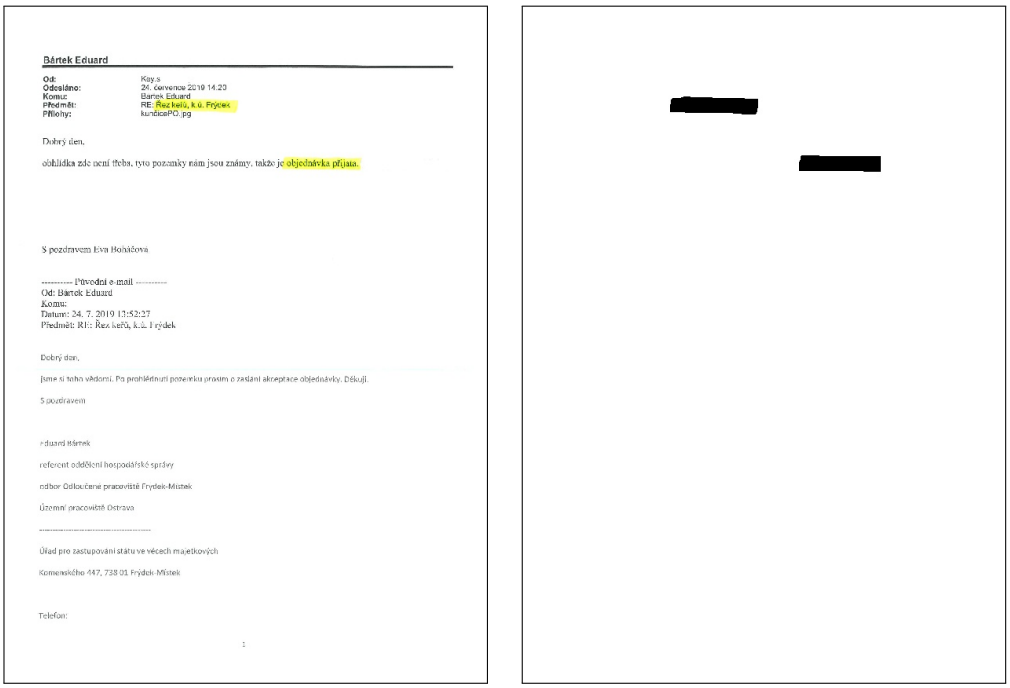
Znovu si môžeme všimnúť celkom výrazného trojuholníka v ľavom dolnom rohu, ktorý náš algoritmus dokázal odfiltrovať a nevyhodnotiť ako tmavú plochu, čo by spôsobilo nesprávne označenie oblasti za anonymizovanú.



Obr. 5.4 Percento anonymizovania : 0,08 %

Na obrázku 5.4 vidíme ďalšiu stranu z dokumentu, ktorý bol skenovaný veľmi nekvalitne. V origináli sa síce nevyskytuje anonymizovaná oblasť, no kvôli veľmi zlému skenu boli niektoré oblasti dostatočne tmavé na to, aby boli vyhodnotené naším algoritmom za anonymizované oblasti. Nejedná sa o veľké plochy a možné riešenie takýchto anomálií popisujeme v záverečnej kapitole 6.1.1.

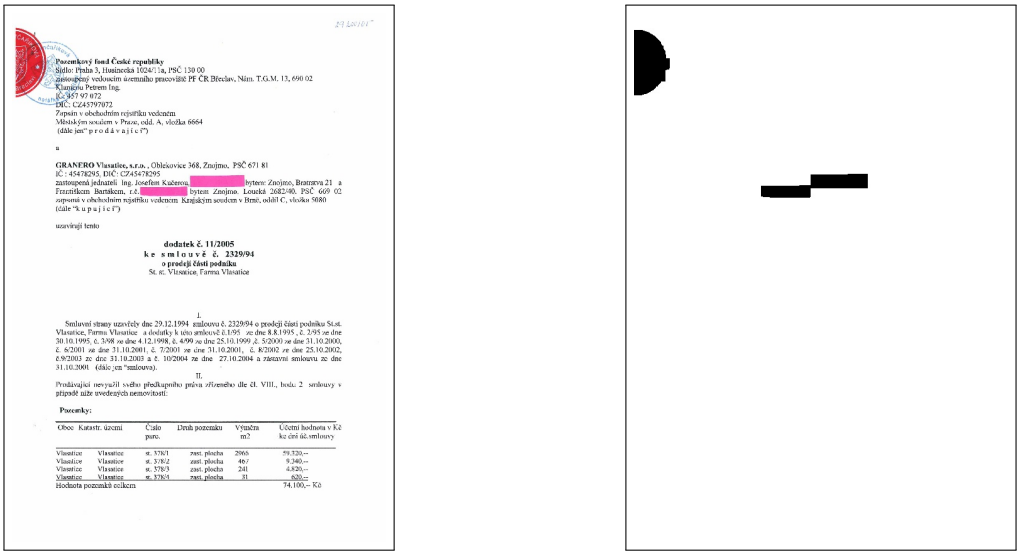
Na posledných príkladoch si ukážeme tzv. false positives, ktoré algoritmus nebol schopný rozoznať a správne odfiltrovať.



Obr. 5.5 Percento anonymizovania : 0,07 %

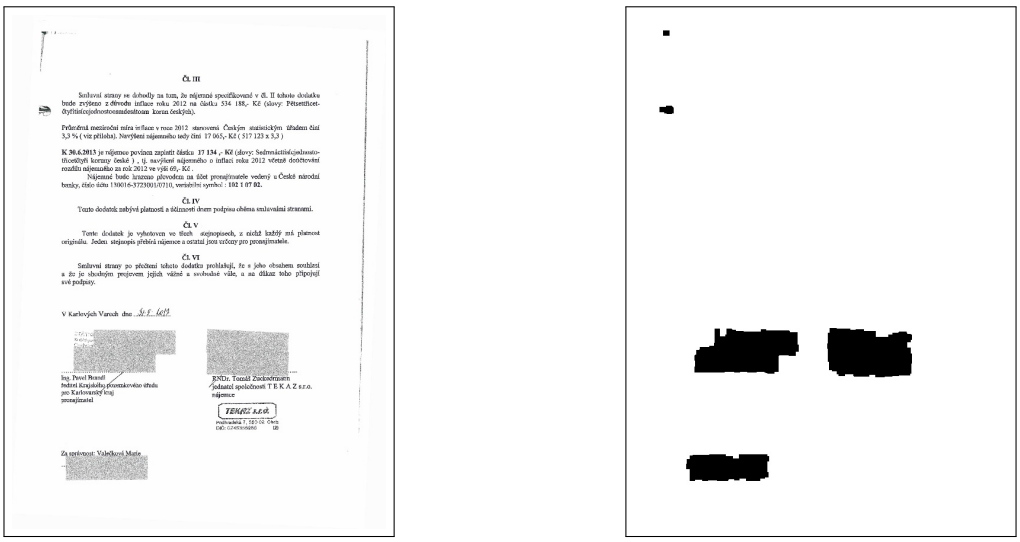
Ako bolo spomenuté v popise algoritmu v 4.1.2, v prípade detekcie farebných pixelov berieme tieto pixely ako možné anonymizované oblasti. V tomto prípade sa však jedná o zvýrazňovač, no algoritmus to vyhodnotil ako anonymizovanú plochu, a teda tieto oblasti sú false positive. Možné riešenia a prístupy sú spomenuté v záverečnej kapitole 6.1.1.

Na obrázku 5.6 vidíme obdobný problém, tentokrát týkajúci sa úradnej pečiatky, ktorá je výrazná červená.



Obr. 5.6 Percento anonymizovania : 1,4 %

Posledný príklad, ktorý uvedieme (obr. 5.7), je detekcia anonymizovaných oblastí šumom. Tento typ anonymizácie bol jednym z hlavných problémov, nakoľko bolo veľmi ťažké správne vyfiltrovať bežný šum, ktorý vznikal pri zlom skene od šumu, ktorý prekrýval nejaké údaje a teda sa jednalo o anonymizovanú oblasť. Detegované oblasti nie sú dokonalé, každopádne môžeme prehlásiť, že aj takýto typ anonymizácie sme schopní detegovať a relatívne presne označiť.



Obr. 5.7 Percento anonymizovania : 4,2 %

5.1.1 Dopady na ďalší vývoj

Na základe vykonaných pozorovaní a analýz detegovaných oblastí je zrejmé, že algoritmus má schopnosť identifikovať anonymizované oblasti s vysokou presnosťou, zároveň však sledujeme výzvy v špecifických scenároch, ktoré vyžadujú ďalšie vylepšenia. Niekoľko kľúčových poznatkov a návrhov pre ďalší význam, ktoré uvádzame, sú:

1. Zlepšenie filtrovacích metód.

   Ako sme si mohli všimnúť v príkladoch s nekvalitnými skenmi, algoritmus má tendenciu označovať tmavé zašumené oblasti za anonymizované. Pre budúci vývoj tak predpokladáme implementáciu pokročilých filtrovacích techník, ktoré by boli schopné dokázať lepšie rozlišovať medzi skutočne anonymizovanými oblasťami a nedokonalosťami skenu.

2. Rozpoznávanie kontextu.

   Ďalší vývoj algoritmu by mal cieliť na identifikáciu vzorov, kde dochádza k anonymizácii častejšie a naopak miesta, kde sa anonymizácia spravidla nevyskytuje (napr. okraje strán, nadpisy, určité formáty textov...).

3. Optimalizácia detekcie tvarov.

   Algoritmus často nesprávne označí zvýrazňovače či farebné pečiatky ako anonymizované oblasti. Budúci vývoj by mal zahrnúť detekciu takýchto tvarov a správne ich identifikovať a ignorovať.

Kapitola 6

Implementácia riešenia

6.1 Výber technológií a nástrojov

6.1.1 Kritériá výberu

Rozhodnutie pri výbere technológie padlo na `C#` a `.NET 8` kvôli robustnej podpore pre webové služby a pokročilým vedomostiam daného jazyka. Pre analýzu a spracovanie PDF dokumentov boli použité knižnice `ImageMagick` a `OpenCV`. Náš softvér cieli na operačný system Windows 10.

6.1.2 Použité technológie

- Jazyk: `C#`
- Framework: `.NET 8`
- API: Minimal API v `.NET 8`
- Použité knižnice (NuGety):
  - Coverlet [16]
    - Použitý pre štatistiku nad pokrytím kódu testami.
  - ErrorOr [17]
    - Menší NuGet, ktorý pomáha riešiť vyhadzovanie výnimiek a miesto nich poskytuje intuitívny prístup k návratovej hodnote.
  - FluentValidation [18]
    - Intuitívny spôsob validácie vstupov a validácie pri tvorbe objektov.

- Magick.NET [19]

  - Nadstavba nad ImageMagick, vhodný pri manipulácii a úprave obrázkov, v našom prípade použitý pri konverzii PDF dokumentov do formátu vhodného pre ďalšie procesy.

- Mapster [20]

  - Umožňuje jednoduché mapovanie objektov, primárne použitý pri mapovaní HTTP requestov na objekty, s ktorými ďalej pracuje aplikácia.

- MediatR [21]

  - Umožňuje implementáciu vzoru mediátora, čím zjednodušuje komunikáciu medzi komponentami aplikácie tým, že sprostredkováva požiadavky a notifikácie medzi odosielateľmi a prijímateľmi bez ich vzájomnej závislosti.

- Entity Framework Core [22]

  - ORM knižnica pre .NET, ktorá umožňuje pracovať s databázami, umožňuje mapovanie medzi databázovými tabuľkami a triedami v aplikácii.

- Microsoft DependencyInjection [23]

  - Poskytuje vstavanú podporu pre Dependency Injection, umožňuje automatickú správu závislostí medzi jednotlivými komponentami aplikácie.

- OpenCvSharp4 [15]

  - Nadstavba nad OpenCV pre .NET.

- xUnit [24]

  - Testovací framework slúžiaci na jednoduché testovanie aplikácie.

Aplikácia je vybavená aj SQLite[25] databázou, ktorá poskytuje ľahkú a kompaktnú relačnú databázu a umožňuje lokálne ukladanie a spracovanie dát bez potreby samostatného servera.

6.2 Architektúra a dizajn systému

6.2.1 Architektúra riešenia

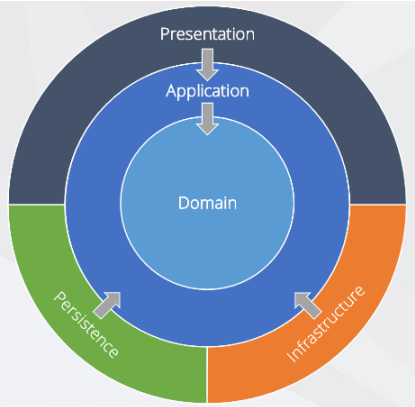
Architektúra našej aplikácie je založená na tzv. Domain-Driven Design, ktorá je popísana v knihe od Scotta Millett-a a Nicka Tune-a, *Patterns, Principles, and Practices of Domain-Driven Design*[26]. Domain-Driven Design (DDD), resp.

Domain-Driven architecture je často označovaný aj ako vrstvová architektúra, cibuľová architektúra či "čistá" architektúra.

Vrstvová architektúra je rozdelená do viacerých vrstiev, ktoré sa vzťahujú na rozličné aspekty daného systému. Každá vrstva má jasnú a špecifickú úlohu a vrstvy sa navzájom ovplyvňujú len jedným smerom.

Hlavnými prvkami vrstvovej architektúry sú:

1. Prezenčná vrstva: Zodpovedá za interakciu s uživateľom,
2. Aplikačná vrstva: Zodpovedá za implementáciu biznisovej logiky, ktorá sa vzťahuje na uživateľské požiadavky,
3. Servisná vrstva: Slúži na poskytovanie služieb, ktoré sú potrebné pre aplikáciu,
4. Infraštruktúrna vrstva: Zodpovedá za správu a komunikáciu s externými zdrojmi,
5. Perzistentná vrstva: Slúži na komunikáciu s perzistentne uloženými dátami, databázou.



Obr. 6.1 Grafické znázornenie cibuľovej architektúry[27]

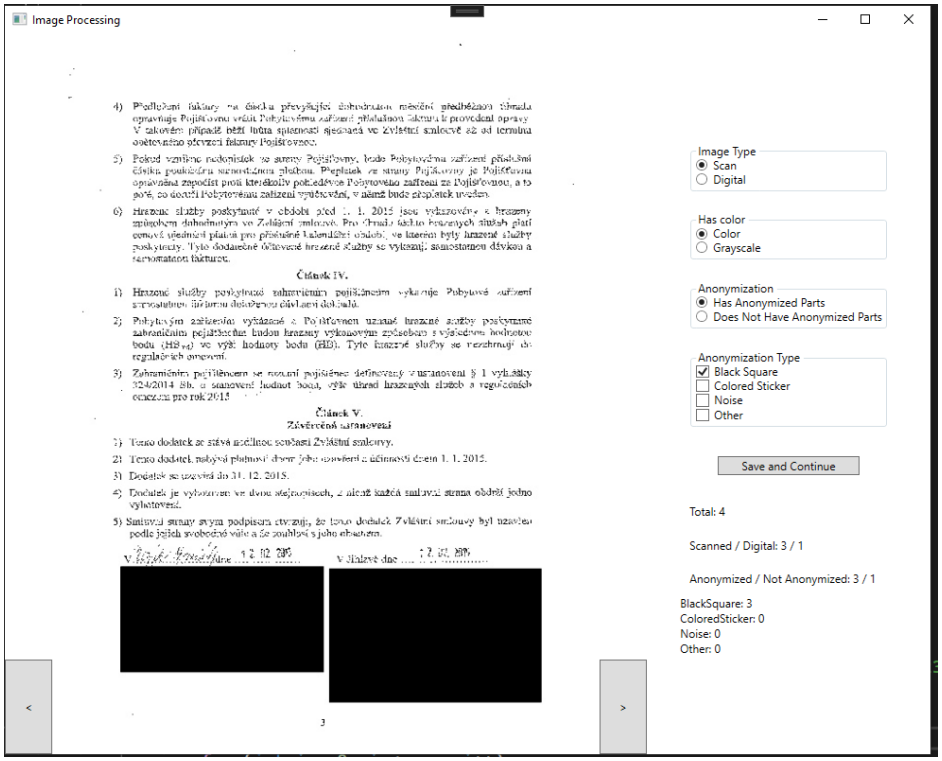
Každá vrstva je navrhnutá tak, aby bola izolovaná od ostatných vrstiev. Vrstvy medzi sebou komunikujú pomocou rozhraní a preto zmeny v jednej vrstve nevytvárajú nepredvídateľné zmeny v iných vrstvách. Pomocou rozhraní definujeme, čo dané vrstvy môžu a nemôžu robiť.

Najväčšími výhodami tejto architektúry sú modularita, odolnosť voči zmenám a jednoduchosť pri návrhu samotnej aplikácie.

6.3 Detaily implementácie

6.3.1 Pomocné aplikácie a testovacie projekty

V rámci projektu DAPP[28] sme implementovali pomocný projekt s názvom `ManualCheckerUtility`. Pri spustení automaticky stiahne dokumenty, ktoré sú následne po jednotlivých stránkach zobrazované uživateľovi. V pravej strane sa nachádza niekoľko možností, ktoré užívateľ môže zvoliť a tie sa následne postupne ukladajú do štatistiky.



Obr. 6.2 Ukážka grafického rozhrania `ManualCheckerUtility`

Súčasťou riešenia okrem testov je aj projekt s názvom `GenerateTestResults`. Jedná sa o jednoduchý skript, ktorý umožňuje rýchlo generovať výsledky s aktuálnym kódom a ukladať tieto výsledky sekvenčne s prebiehajúcimi zmenami. Skript si na začiatku vytvorí WebClient hlavného programu nesúceho názov API2. API2 je druhá verzia projektu API, ktorá bola vzhľadom na použitú architektúru zastaraná a preto sme sa rozhodli prejsť na tzv. minimal api [29]. Následne po inicializácii WebClient-a prevoláva toto API s requestami obsahujúcimi jednotlivé dokumenty z našej testovacej sady. Po vrátení response sa tieto dáta dekomponujú z formátu JSON a uložia na disk, kde ich je potom možné prehliadať. Ukladajú sa

originálne snímky, analyzované snímky a percentá anonymizácie pre jednotlivé strany.

6.3.2 Realizácia algoritmu

Hlavným projektom, ktorý je nosnou časťou našej aplikácie okrem samotného API a jeho častí, je projekt s názvom `DAPPAnalyzer`, ktorý obsahuje algoritmus na detekciu anonymizovaných častí.

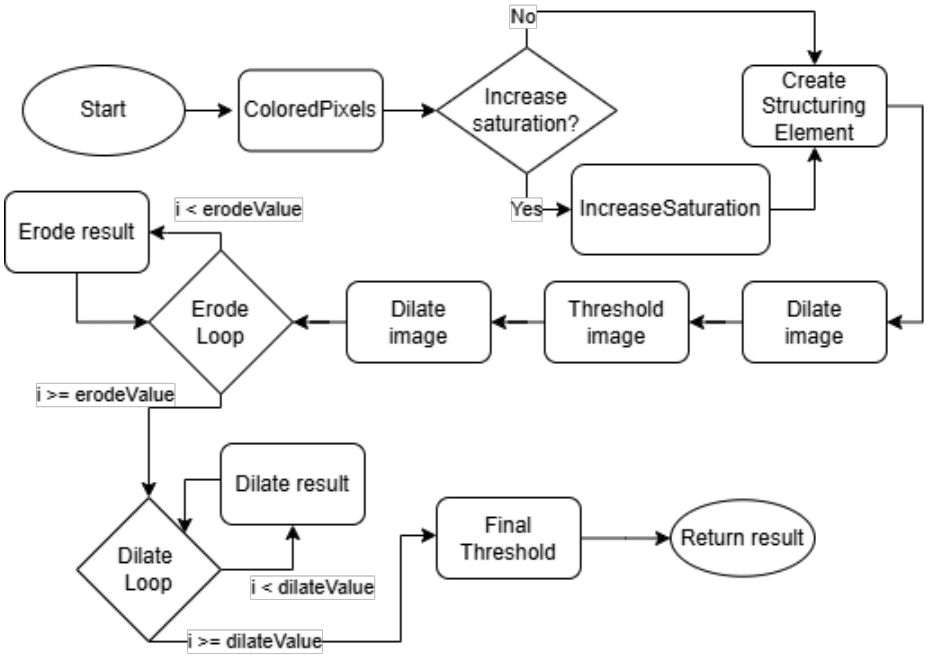
Súbor `PDFAnalyzer.cs` obsahuje implementáciu triedy `PDFAnalyzer`, ktorá implementuje niekoľko metód. Hlavnou a vstupnou metódou je metóda `Task<AnalyzedResult> AnalyzeAsync(DappPDF pdf);`, ktorá je jedinou verejnou metódou tejto triedy. Jej funkciou je paralelne pre každú stranu z parametru pustiť privátnu funkciu `AnalyzePage` a následne vrátené výsledky uložiť do modelu `AnalyzedResult`, ktorý obsahuje okrem identifikačných znakov, akými sú názov dokumentu, url či počet strán dokumentu, samotné výsledky analýzy. Tými sú:

- `containsAnonymizedData` : boolean, ktorý určuje, či boli detegované anonymizované oblasti,
- `anonymizedPercentagePerPage` : slovník, kde kľúč je index strany a hodnota je percento anonymizácie pre danú stranu,
- `originalImages` : slovník, kde kľúč je index strany a hodnota je `byte[]`, teda snímok danej strany uložený ako `byte array`,
- `anonymizedImages` : rovnako ako pri `originalImages`, až na to, že `byte[]` obsahuje výslednú "masku", teda obraz, kde sú znázornené len anonymizované oblasti.

Na ďalšej strane prikladáme zdrojový kód tejto funkcie. Pre ešte bližšie detaily prikladáme v prílohe zdrojový kód, kde je dostupná celá implementácia so všetkými pomocnými skriptami a testovacími súbormi.

```
internal static Mat GetAnonymizedParts(
    Mat img,
    int erodeValue = 8,
    int dilateValue = 4)
{

    var coloredPixels = ColoredPixels(img);
    // Increase their saturation
    var imgSaturatedColors = img;
    if (coloredPixels.Count != 0)
    {
       imgSaturatedColors =
            IncreaseSaturation(img, coloredPixels, 100);
    }
    // Create structuring element
    Mat se =
        Cv2.GetStructuringElement(
            MorphShapes.Rect,
            new Size(3, 3));
    var dilated = Dilate(imgSaturatedColors, se);

    var dilated_threshold = Threshold(dilated, 20);
    var dilated2 = Dilate(dilated_threshold, se);


    var result = dilated2;
    for (int i = 0; i < erodeValue; i++)
    {
        result = Erode(result, se);
    }
    for (int i = 0; i < dilateValue; i++)
    {
        result = Dilate(result, se);
    }

    return Threshold(result,127);
}
```

Pre jednoduchšiu čitateľnosť je na ďalšej strane znázornený stavový diagram.



Obr. 6.3 Diagram funkcie `GetAnonymizedParts`

6.3.3 Testovanie

Vývoj prebiehal na verzovacom systéme git a je uložený na školskej inštancii gitlab [28], v repozitári je pripravená CI/CD pipeline na automatický build a automatické spustenie testov. Na obrázku 6.4 vidíme pokrytie testami sprostredkované frameworkom coverlet[16].

Files	Tracked lines	Covered	Partial	Missed	Coverage %
..					
API2	146	106	16	24	72.60%
Application	121	121	0	0	100.00%
Contracts	20	16	0	4	80.00%
Domain	89	82	3	4	92.13%
Infrastructure	446	360	0	86	80.72%
PDFAnalyzer	131	125	2	4	95.42%
Subtotal	953	810	21	122	

Obr. 6.4 Pokrytie testov jednotlivých častí softvéru[16]

Kapitola 7

Záver

7.1 Zhrnutie dosiahnutých výsledkov

V práci sme sa zamerali na vývoj a testovanie programu s algoritmom na detekciu anonymizovaných oblastí v dokumentoch. Výsledky, ktoré sme dosiahli, zahŕňajú:

- Vývoj algoritmu na detekciu anonymizovaných oblastí

  Implementovali sme algoritmus schopný detegovať anonymizované oblasti rôznych typov v skenovaných a digitálnych dokumentoch. Algoritmus sme navrhli tak, aby bol schopný detegovať rôzne anonymizované oblasti, vrátane nepravidelných a farebných či šumových anonymizácií.

- Testovanie a validácia

  Algoritmus sme testovali na mnohých dokumentoch, ktoré zahŕňali rôzne typy anonymizácie. Výsledky ukázali vysokú presnosť detekcie v prípade bežných typov anonymizovania, ako sú čierne obdĺžniky a farebné nálepky.

- Špecifické výzvy

  Podrobnou analýzou sme identifikovali problémy spojené s detekciou anonymizovaných oblastí v prípade nekvalitných skenov a dokumentov s rôznymi druhmi rušivých elementov. Tieto výzvy boli čiastočne vyriešené, no stále je tu priestor na ďalší výskum a zlepšenie.

- Možnosti integrácie a nasadenia

  V súčasnej dobe nie je aplikácia integrovaná so žiadnym iným systémom ani nasadená ako samostatná aplikácia. Aplikácia je však vhodná k použitiu ako referenčná implementácia. V budúcnosti je možnosť integrácie tejto aplikácie v rámci Hlídače státu[1].

7.2 Doporučenia pre ďalší výskum

Vychádzajúc z našich zistení a výsledkov máme niekoľko doporučení pre budúci výskum v oblasti detekcie anonymizovaných oblastí:

- Vyššia odolnosť voči rušivým elementom

  Významným problémom, ktorému je možné venovať sa bližšie, je redukcia rušivých elementov v dokumentoch, obzvlášť v prípade nekvalitných skenov, ale aj digitálných dokumentov. Jedná sa o logá firiem, úradné pečiatky, záhlavia tabuliek či iné rušivé elementy, ktoré sú problémom pre náš algoritmus pri správnom vyhodnocovaní a vyriešením tohoto problému by sa prispelo k redukcii falošne pozitívnych detekcií.

- Využitie metód strojového učenia

  Implementácia konvolučných sietí alebo iných metód strojového učenia môže zvýšiť presnosť a správnosť detekcie. Problémom je tu však absencia datasetu, na základe ktorého by mohla byť takáto konvolučná sieť namodelovaná.

7.3 Osobné zistenia a závery

Pri práci na tejto téme sme si uvedomili viacero kľúčových bodov, ktoré sú dôležité pre správne pochopenie z pohľadu technického vývoja. Medzi najdôležitejšie body uvádzame komplexnosť a rôznorodosť reálnych dát. Rôzne formáty dokumentov, kvalita skenov, typy anonymizácií predstavovali a predstavujú výzvy, ktoré si vyžadujú komplexné a flexibilné riešenia.

Ďalším bodom, ktorý spomenieme, je spoľahlivosť a presnosť. Falošne pozitívne výsledky môžu mať závažné dopady, obzvlášť v prípade, že by sa na základe týchto výsledkov hodnotila transparentnosť či dôvernosť inštitúcií, ktoré takéto dokumenty, resp. zmluvy zverejňujú.

Dúfame, že tieto zistenia a závery poslúžia ako základ pre ďalší vývoj a výskum v oblasti detekcie anonymizovaných oblastí v dokumentoch.

Použitá literatúra a iné zdroje

[1] Hlídač státu z.ú. https://hlidacstatu.cz/. Online: 2024-07-11.

[2] Webpomoc s.r.o. https://www.webpomoc.sk/aky-je-rozdiel-medzi-pseudonymizaciou-a-anonymizaciou. Online: 2024-07-11.

[3] AION CS s.r.o. https://www.zakonyprolidi.cz/cs/2000-101. Online: 2024-07-11.

[4] Úrad pre vydávanie publikácií Európskej únie. https://eur-lex.europa.eu/legal-content/en/TXT/?uri=CELEX%3A32016R0679. Online: 2024-07-11.

[5] AION CS s.r.o. https://www.zakonyprolidi.cz/cs/2005-412. Online: 2024-07-11.

[6] Software602 a.s. https://man.602.cz/site/signer/obsluha/anonymizace.html. Online: 2024-07-11.

[7] Syntho B.V. https://www.syntho.ai/sk/the-best-data-anonymization-tools-next-gen-techniques/. Online: 2024-07-11.

[8] Ministerstvo vnitra ČR. https://anonymizace.gov.cz/crossroad/. Online: 2024-07-11.

[9] AION CS s.r.o. https://www.zakonyprolidi.cz/cs/2015-340. Online: 2024-07-11.

[10] Ministerstvo vnitra ČR. https://smlouvy.gov.cz. Online: 2024-07-11.

[11] Laurens Leurs. https://prepressure.com/library/paper-size. Online: 2024-07-11.

[12] Šikudová Elena; Černeková Zuzana; Benešová Wanda; Haladová Zuzana; Kučerová Júlia. *Počítačové videnie Detekcia a rozpoznávanie objektov*. Wikina Praha, 2014. ISBN: 978-80-87925-06-5. URL: http://www.sccg.sk/~sikudova/strukturovana_kniha_CD.pdf.

[13] Otsu Nobuyuki. "A threshold selection method from gray-level histograms". In: *IEEE Transactions on Systems, Man, and Cybernetics* (1979). DOI: 10.1109/TSMC.1979.4310076.

[14] Malia Gehan; Noah Fahlgren. https://plantcv.readthedocs.io/en/stable/nonuniform_illumination/. Online: 2024-07-11.

[15] shimat. https://github.com/shimat/opencvsharp. Online: 2024-07-11.

[16] Toni Solarin-Sodara. https://github.com/coverlet-coverage/coverlet. Online: 2024-07-11.

[17] Amichai Mantinband. https://github.com/amantinband/error-or. Online: 2024-07-11.

[18] Jeremy Skinner. https://docs.fluentvalidation.net/en/latest/. Online: 2024-07-11.

[19] Dirk Lemstra. https://github.com/dlemstra/Magick.NET/. Online: 2024-07-11.

[20] Chaowlert Chaisrichalermpol; Eric Swann. https://github.com/MapsterMapper/Mapster. Online: 2024-07-11.

[21] Jimmy Bogard. https://github.com/jbogard/MediatR. Online: 2024-07-11.

[22] Microsoft Corporation. https://github.com/dotnet/efcore. Online: 2024-07-11.

[23] Microsoft Corporation. https://learn.microsoft.com/en-us/dotnet/core/extensions/dependency-injection. Online: 2024-07-11.

[24] .NET Foundation. https://xunit.net/. Online: 2024-07-11.

[25] SQLite Org. https://www.sqlite.org/index.html. Online: 2024-07-11.

[26] Scott Millet; Nick Tune. *Patterns, Principles and Practices of Domain-Driven Design*. 2015. ISBN: 978-1118714706.

[27] Daniel Rusnok. https://dev.to/danielrusnok/onion-architecture-or-how-to-not-make-spaghetti-244b. Online: 2024-07-11.

[28] Lukáš Salak. https://gitlab.mff.cuni.cz/salakl/detection-anonymized-parts-in-pdfs-bachelor-thesis. Online: 2024-07-11.

[29] Microsoft Corporation. https://learn.microsoft.com/en-us/aspnet/core/fundamentals/minimal-apis/overview?view=aspnetcore-8.0. Online: 2024-07-11.

Zoznam obrázkov

Zoznam tabuliek

Prílohy

Zdrojový kód, postup pri inštalácii, užívateľská a technická dokumentácia spoločne s pomocnými skriptami a testovacím datasetom dokumentov, ktoré boli v tejto práci použité, je možné nájsť v priloženom súbore `attachments.zip`. Tento komprimovaný súbor obsahuje taktiež tabuľky a obrázky, ktoré boli použité v tejto práci.