Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Získavanie štruktúrovaných dát o pacientoch s ochorením COVID-19 z prepúšťacích správ a krvných výsledkov

Bakalárska práca

2023
Marián Kravec

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Získavanie štruktúrovaných dát o pacientoch s ochorením COVID-19 z prepúšťacích správ a krvných výsledkov

Bakalárska práca

Študijný program:	Dátová veda
Študijný odbor:	Informatika a Matematika
Školiace pracovisko:	Katedra aplikovanej informatiky
Školiteľ:	Mgr. Vladimír Boža, PhD.

Bratislava, 2023
Marián Kravec

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Marián Kravec
Študijný program: dátová veda (Medziodborové štúdium, bakalársky I. st., denná forma)
Študijné odbory: informatika
matematika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Získavanie štruktúrovaných dát o pacientoch s ochorením COVID-19 z prepúšťacích správ a krvných výsledkov
Extracting structured data about patients with COVID-19 from discharge reports and blood results

Anotácia: Nemocničný informačný systém v Univerzitnej nemocnici v Bratislava obsahuje iba neštrukturované textové dáta o pacientoch (či už prepúšťaciu správu alebo výsledky krvných testov).

Takýto formát dát neumožňuje efektívnu analýzu dát a hľadanie faktorov, ktoré ovplyvňujú prognózu liečby Covid-19.

Cieľom práce je vytvoriť softvér, ktorý tieto neštruktúrované dáta premení do tabuľkovej podoby, ktoré sa dajú následne jednoducho analyzovať.

Vedúci: Mgr. Vladimír Boža, PhD.
Katedra: FMFI.KAI - Katedra aplikovanej informatiky
Vedúci katedry: prof. Ing. Igor Farkaš, Dr.

Dátum zadania: 20.10.2022

Dátum schválenia: 22.10.2022

doc. Mgr. Tomáš Vinař, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie: Touto cestou by som rád poďakoval svojmu školiteľovi Mgr. Vladimírovi Božovi, Phd. za ochotu a rady pri písaní práce a kódu. Zároveň by som rád poďakoval pánovi doc. MUDr. Petrovi Sabakovi, PhD. za umožnenie použitia kódu vyvinutého počas projektu na účely tejto práce.

Abstrakt

Nemocničný informačný systém v Univerzitnej nemocnici v Bratislava obsahuje iba neštrukturované textové dáta o pacientoch (či už prepúšťaciu správu alebo výsledky krvných testov). Takýto formát dát neumožňuje efektívnu analýzu dát a hľadanie faktorov, ktoré ovplyvňujú prognózu liečby COVID-19. Cieľom práce je vytvoriť softvér, ktorý tieto neštruktúrované dáta premení do tabuľkovej podoby, ktoré sa dajú následne jednoducho analyzovať.

Kľúčové slová: prepúšťacia správa, regulárny výraz, tretie

Abstract

Abstract in the English language (translation of the abstract in the Slovak language).

Keywords: dismissal report, regular expresion

Obsah

Úvod **1**

1 Podobné práce **3**

2 Štruktúra prepúšťacej správy **7**

2.1 Výzor vstupných dát 7

2.1.1 Blok A - Osobné údaje a obdobie hospitalizácie 7

2.1.2 Blok B - Anamnéza 7

2.1.3 Blok C - Vyšetrenia 8

2.1.4 Blok D - Terapia 8

2.1.5 Blok E - Epikríza 8

2.1.6 Blok F - Záver, odporúčania, špecifické nálezy 9

2.1.7 Blok G - Krvné výsledky 9

2.2 Problémy pri rozdelení dát na bloky 9

2.2.1 Problém nenájdených blokov 9

2.2.2 Problém nesprávne umiestnených dát 10

2.3 Riešenia problémov s rozdelením dát do blokov 10

2.3.1 Homogénny text 10

2.3.2 Menej väčších blokov 10

3 Získavanie dát **13**

3.1 Osobné údaje pacienta a obdobie hospitalizácie 13

3.2 Protilátky proti vírusu SARS-CoV-2 14

4 Analýza výsledkov a chýb **15**

5 Modifikácia a vylepšenia **17**

Záver **19**

Príloha A **23**

Príloha B **25**

Zoznam obrázkov

2.1 Rozloženie správy 8
2.2 Upravené rozloženie správy 11

3.1 Hlavička a) 13
3.2 Hlavička b) 14
3.3 Protilátky 14

Úvod

V nemocničnom informačnom systéme Univerzitnej nemocnice v Bratislave nie je možnosť jednoducho získať tabuľku obsahujúcu dáta jedného pacienta respektíve skupiny pacientov, tieto dát sa získavali ručne buď postupným kopírovaním jednotlivých dát nachádzajúcich sa v rôznych častiach informačného systému alebo ich hľadaním a následným prepisovaním z prepúšťacej správy.

Tento proces bolo potrebné opakovať pre každého jedného pacienta čo bolo časovo náročné a vyžadovalo si nezanedbateľné množstvo ľudskej práce.

Hlavným účelom tejto práce je vytvorenie softvéru, ktorý pomôže výrazne zrýchliť a zjednodušiť získavanie týchto dát.

Tento softvér na svojom vstupe dostane prepúšťaciu správu pacienta a jeho krvné výsledky v podobe klasického textu a následne z týchto ne-štrukturalizovaných dát získava jednotlivé požadované dáta pričom v prípade, že nejakú informáciu nenájde alebo zistí, že zistená hodnota nie je v očakávaných limitoch alebo, že sa v správe nachádza viacero hodnôt pre jednu informáciu tak užívateľovi oznámi o akú informáciu ide a aký je s ňou problém.

Práca je rozdelená do piatich kapitol. V prvá obsahuje informácie o podobných softvéroch. Druhá kapitola sa zameriava na štruktúru prepúšťacej správy a dáta ktoré sa z nej snažíme získať. Tretia kapitola obsahuje problémy ktoré sa objavili pre jednotlivé získavané informácie a aké spôsoby riešenia sme sa rozhodli použiť. V štvrtej je analýza chybovosti softvéru. Posledná kapitola je o modifikáciách softvéru aby bol jednoduchšie použiteľný a modifikovateľný pre iné podobné použitia.

Kapitola 1

Podobné práce

Aj napriek moderným nemocničným informačným systémom je stále veľké množstvo nemocničných záznamov v podobe čistého alebo čiastočne štrukturovaného textu z ktorého je ručné získavanie dát časovo náročné. Preto sa na to využívajú automatizované systémy ktoré zväčša fungujú na jednom z dvoch princípov, respektíve na kombinácii oboch. Tými prístupmi sú regulárne výrazy a metódy strojového učenia určené na spracovanie prirodzeného jazyka.

Metóda regulárnych výrazov využíva na hľadanie informácii v texte špeciálne kódované reťazce znakov ktoré slúžia ako vzor. V texte sa hľadajú miesta, ktoré sa zhodujú so vzorom a z nájdených miest sa následne vyberá získavaná informácia.

Metóda strojového učenia

Obe tieto prístupy majú svoje výhody a nevýhody [1]. Výhodou regulárnych výrazov je ich presnosť a transparentnosť čiže možnosť vidieť a upravovať vnútorné fungovanie programu, presnejšie možnosť upravovať jednotlivé regulárne výrazy hľadajúce konkrétne informácie. Medzi nevýhody tohto prístupu patrí napríklad fakt, že všetky regulárne výrazy ktoré softvér využíva treba ručne vytvárať a vylepšovať čo je často náročné, väčšinou sú špecifické pre určitú oblasť pre ktorú je softvér vytváraný čo komplikuje využívanie regulárnych výrazov ktorých správne fungovanie bolo už otestované v iných softvéroch. Ďalšou nevýhodou je, že v prípade komplikovaných výrazov je aj ich upravovanie a vylepšovanie komplikované. Na druhej strane v prípade použitia niektorej z metód strojového učenia je často možné využiť už existujúcu metódu spracúvajúcu prirodzený jazyk a modifikovať ju pre konkrétne použitie a natrénovať model na predpripravených dátach, avšak aj tento prístup má nevýhody ako napríklad, že natrénovaný model často nie sú až tak presný ako dobre nastavené regulárne výrazy, pre zvýšenie presnosti je často nutné zväčšiť množstvo ručne spracovať dát určených na trénovanie modelu a zároveň v prípade nájdenia častej chyby alebo nutnosti modifikácie hľadaných dát (pridanie alebo odobranie získavanej informácie) je nutné model upraviť a celý nanovo pretrénovať a validovať aj už skôr validované časti.

Medzi už existujúce systémy využívajúce regulárne výrazy patrí napríklad systém HEDEA, [2] čiže Healthcare Data Extraction and Analysis ktorého autormi sú Anshul Aggarwal, Sunita Garhwal a Ajay Kumar a bol vyvinutý na získavanie Indických medicínskych dát. Systém využitím regulárnych výrazov hľadá každú jednu získavanú informáciu tak, že hľadá v texte kľúčové slovo označujúce požadovanú informáciu, následne ak by mala k danej informácií existovať aj konkrétna hodnota (či už číselná alebo slovná) tak hľadá túto hodnotu v okolí kľúčového slova a následne túto informáciu aj s jej hodnotou zapíše do databázy k konkrétnemu pacientovi na základe jeho identifikačného čísla ktoré sa na každom spracovávanom texte musí nachádzať. Podobne ako v našom prípade je hlavnou úlohou tohto softvéru získavať medicínske dáta z čiastočne štrukturalizovaných vstupných dát čiže lekárskych správ a výsledkov testov. Hlavné rozdiely oproti nášmu systému sú, že systém HEDEA sa snaží získavať iba základné dáta o pacientovi ako sú osobné údaje, výška, váha, tlak, základné krvné výsledky a prekonané ochorenia inými slovami jeho účelom je vytvoriť databázu obsahujúcu anamnézy jednotlivých pacientov ktorú môže využiť lekár pri diagnostike daného pacienta zatiaľ čo my získavame okrem týchto dát aj dáta špecifické pre pacientov s ochorením COVID-19 ako napríklad typ oxygenoterapie alebo výsledky testov na protilátky proti vírusu SARS-CoV-2 a našou snahou je vytvoriť tabuľku s dátami o ochorení COVID-19 ktorá sa dá využiť na analýzu rizikových faktorov a liečby tohto ochorenia, zároveň systém HEDEA určený na spracovávanie Indických dát takže je vytvorený pre dáta písané v oficiálnom jazyku Indie v tomto prípade angličtinu zatiaľ čo náš model je vytvorený pre dáta v slovenčine.

Systémom využívajúcim metódu strojového učenia na spracovávanie prirodzeného jazyka je napríklad systém ktorý vytvorili Fette a kol. na Univerzite vo Würzburgu [3] ktorý využívajúci metódu učenia s učiteľom s názvom Conditional random field ktorej úlohou je označiť jednotlivé slová respektíve viacslovné pomenovania [4] a následne pomocou metódy Keyword Matching with Terminology based disambiguation prepojiť nájdené slová a viacslovné pomenovania s databázou odborných pojmov tak, že v prípade jednoznačného prepojenie považuje danú informáciu za klasifikovanú a v prípade nejednoznačného prepojenia (dané slovo môže byť časťou rôznych informácii napríklad v prípade číselnej hodnoty nevieme bez ďalšej informácie určiť k čomu patrí) hľadá v okolí označeného slova iné označené slovo s jednoznačným prepojením ktoré bližšie určí prepojenie nejednoznačného slova. Okrem samotného spôsobu získavanie dát je v porovnaní s našim systémom rozdiel aj v prioritách pri získavaní dát keďže náš systém je vytvorený na čo najväčšiu presnosť pri získavaní dát špecificky z prepúšťacích správ zatiaľ čo ich systém je vytvorený tak aby ho bolo možné byť natrénovaný na získavanie dát z rôznych typov medicínskych dokumentov či už lekárskych správ, výsledkov testov alebo klinických štúdii pričom jedinou podmienkou je aby sa všetky získavané údaje nachádzali v databáze odborných pojmov a takisto platí, že celý systém je vytvorený

pre iný jazyk ako náš systém v tomto prípade ide o nemčinu.

Prístup ktorý kombinuje metódu strojového učenia s regulárnymi výrazmi využili v svojom systéme Cui a kol. [5] ktorý využili metódu s názvom Constructive heuristic ktorej úlohou nebolo priamo hľadanie získavaných informácii v texte ale generovanie čo najlepších regulárnych výrazov na to určených. Tento algoritmus začína s prázdnou množinou regulárnych výrazov a následne iteratívne túto množinu rozširuje a upravuje kým nie je splnená ukončovacia podmienka [6]. Výhodou tohto prístupu oproti bežným metódam strojového učenia je, že na konci trénovania má užívateľ množinu regulárnych výrazov ktoré môže ďalej upravovať a nie "čiernu skrinkuäko v prípade bežnej metódy strojového učenia, ktorej vnútornému fungovaniu je pre človeka nepochopiteľné. Oproti len použitiu regulárnych výrazov má výhodu, že nie je nutné regulárne výrazy vymýšľať od začiatku ale stačí iba výstup mierne upraviť. Nevýhodou je, že pochopenie a upravenie regulárnych výrazov je síce možné ale môže to byť pomerne náročné keďže ide o počítačom generované regulárne výrazy ktoré aj napriek tomu, že fungujú rovnako dobre môžu sa výrazne líšiť od toho čo by napísal človek. Hlavným rozdielom oproti nášmu systému je to, že hlavnou úlohou ich systém nie je priame získavanie dát z medicínskej dokumentácie ale generovanie regulárnych výrazov ktoré je po na takýto problém možné použiť.

Kapitola 2

Štruktúra prepúšťacej správy

Táto kapitola sa zameriava na pochopenie vstupných dát a približnú lokalizáciu hľadaných informácii v nich.

2.1 Výzor vstupných dát

Dáta z ktorých sa snažíme získať informácie o pacientovi softvér dostáva v tvare textu obsahujúceho prepúšťaciu správu a krvné výsledky. Väčšina textu v prepúšťacej správe nie je generovaná automaticky nemocničným informačným systémom ale je písaná lekárom čo spôsobuje, že každá správa je do určitej miery originálna.

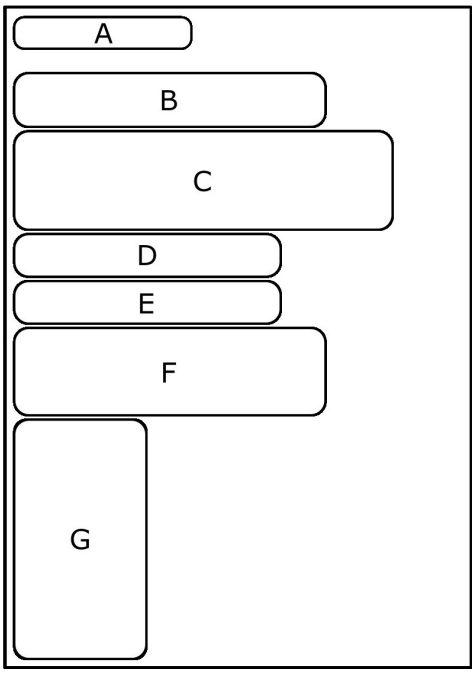
Napriek tomu existuje základná štruktúra ktorú majú spoločnú takmer všetky prepúšťacie správy vďaka ktorej je možné túto správu rozdeliť do blokov (viď obrázok 2.1) ktoré obsahujú určitý informácie. Teraz si prejdeme, čo obsahujú jednotlivé bloky a čo z nich sa mi snažíme získať.

2.1.1 Blok A - Osobné údaje a obdobie hospitalizácie

Bloku A je dvojriadková hlavička v ktorej sa nachádzajú osobné údaje pacienta, čiže jeho celé meno a rodné číslo, a zároveň sa tam nachádza dátum priatia a dátum prepustenia daného pacienta. Všetky tieto informácie sa snažíme získať.

2.1.2 Blok B - Anamnéza

Tento blok obsahuje informácie o anamnéze a stave pacienta pri prijatí do nemocnice, z tohto bloku sa snažíme získavať informácie ako sú výška, váha a saturácia krvi kyslíkom pri prijatí, a informácia o dlhodobých alebo v minulosti prekonaných chorobách a problémoch ako sú cukrovka, astma, demencia, infarkt myokardu, artériová hypertenzia, fibrilácia predsiení, srdcové zlyhanie a ďalšie.



Obr. 2.1: Rozdelenie správy do špecifických blokov podľa ich obsahu

2.1.3 Blok C - Vyšetrenia

Blok C obsahuje informácie o vykonaných vyšetreniach, pre nás sú podstatné výsledky testov na protilátky proti vírusu SARS-CoV-2 typu IgG a IgM pri prijatí, prípadne výsledok testu na ochorenie CDI (Infekcia spôsobená Clostridium difficile).

Zároveň sa tu nachádzajú aj výsledky krvných testov avšak tie sa tu nemusia nachádzať úplné. Preto softvér ktorým to spracovávame ich považuje za kontrolné a samotné informácie o krvných výsledkoch sa získavajú z posledného bloku (2.1.7).

2.1.4 Blok D - Terapia

V tomto bloku sú informácie o terapii odtiaľto získavame informáciu o liekoch ktoré boli pacientovi podané počas hospitalizácie a o tom či pacient potreboval aj oxygenoterapia a v prípade, že áno aj o aký typ oxygenoterapie išlo.

2.1.5 Blok E - Epikríza

Tento blok obsahuje časť správy s názvom epikríza čiže záverečná, súhrnná správa o pacientovi, priebehu jeho choroby a hospitalizácie. Jedinou získavanou informáciou je informácie o smrti pacienta. Táto časť sa však dá zároveň využiť aj na prípadnú kontrolu iných získavaných informácii keďže môže obsahovať informáciu o iných ochoreniach pacienta, jeho liečbe, prípadne o prítomnosti protilátok proti vírusu SARS-CoV-2.

2.1.6 Blok F - Záver, odporúčania, špecifické nálezy

Blok F obsahuje zvyšné časti správy ako sú záver, odporúčania a špecifické nálezy. Avšak obsah tohto bloku je výrazne nekonzistentný a jeho jednotlivé časti nemusia byť vôbec prítomné v správe. Našťastie tento blok by nemal neobsahovať žiadne konkrétne získavané informácie.

2.1.7 Blok G - Krvné výsledky

Záverečný blok už nie je priamo prepúšťacia správa ale ide o krvné výsledky pacienta ktoré narozdiel od výsledkov v bloku C (2.1.3) sa tu nachádzajú úplné a zároveň vďaka tomu, že nie sú napísané vedľa seba ako v bloku C ale pod sebou (každý výsledok na samostatnom riadku) je práca s nimi výrazne jednoduchšia.

2.2 Problémy pri rozdelení dát na bloky

Pri snahe o implementácia tohto delenia sme zistili, že aj napriek tomu, že sa pomerne veľký počet správ dal do takýchto blokov rozdeliť, tak sa objavilo niekoľko problémov či už pri samotnom delení správy ako aj pri informačnom obsahu jednotlivých častí kvôli ktorým sa ukazuje vhodnejšie takéto riešenie vôbec nepoužiť alebo použiť nejakú robustnejšiu verziu tohto delenie. Niektoré z nájdených problémov si teraz opíšeme.

2.2.1 Problém nenájdených blokov

Pri kontrole správ rozdelených do blokov sme zistili, že sa občas stávalo, že náš softvér nebol schopný nájsť niektorý z blokov, väčšinu z týchto problémov vieme rozdeliť do troch skupín: chýbajúci alebo nesprávne ohraničený blok F, blok E skrytý v bloku F, chýbajúci alebo nesprávne napísaný začiatok bloku.

Prvý problém pre nás nepredstavuje veľký problém keďže v bloku F by sa nemali nenachádzajú hľadané informácie.

Druhý problém je o niečo horší keďže blok E je pre nás dôležitý ale tento problém bolo jednoduché opraviť tým, že ak softvér nenájde blok E pri prvotnom delení správy ešte skontroluje či sa v bloku F náhodnou nenachádza.

Tretí problém sa ukazuje ako najproblematickejší sa ukázalo ako pomerne komplikované určiť začiatok a koniec bloku ak softvér nenájde kľúčové slová ktorými sa vo väčšine prípadov jednotlivé bloky začínajú a preto sa stávalo, že softvér niektoré bloky nenašiel a lebo ich pripojil k iným blokom. Tento problém sa stal jedným z hlavných zmeny prístupu k dátam.

2.2.2 Problém nesprávne umiestnených dát

Ako ďalší veľký problém sa ukazuje to, že niektoré informácie sa nenachádzajú sa očakávanom mieste. Zväčša išlo o informácie o anamnéze pacienta a výsledkoch jeho vyšetrení ktoré sme predpokladali, že nájdeme v blokoch B respektíve C avšak časť týchto informácii sa nachádzala až v bloku F.

Tento problém samotný sa dá riešiť tým, že informácie ktoré hľadáme v blokoch B a C budeme nakoniec hľadať avšak tento problém spôsobuje, že problém ktorý máme s nájdením a správnym ohraničením bloku F sa stáva relevantným a treba ho riešiť, nanešťastie tento problém je podobný problému ohraničeniu ostatných blokov a preto je jeho riešenie pomerne komplikované.

2.3 Riešenia problémov s rozdelením dát do blokov

Skúšaním rôznych spôsobov riešenia sa ukázalo, že najvhodnejšie je neriešiť okrajové prípady súčasného rozdelenia správy ale prerobiť samotné rozdeľovanie. Našli sme najlepšie spôsoby...

2.3.1 Homogénny text

Jednou možnosťou je považovať celý text ako jeden homogénny text v ktorom hľadáme všetky informácie.

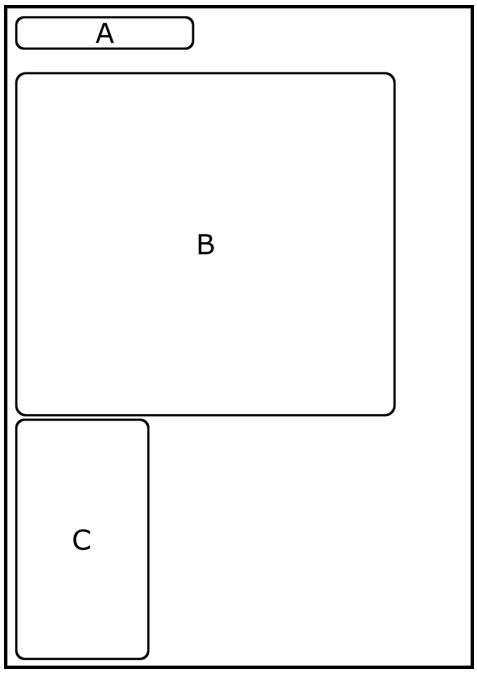
Tento prístup rieši všetky naše problémy avšak zbytočne hľadá niektoré informácie aj na miestach kde vieme, že sa nikdy nebudú nachádzať čo ho spomaľuje.

2.3.2 Menej väčších blokov

Tento prístup využíva rozdelenie do blokov ale namiesto pôvodných 7 blokov toto nové rozdelenie má iba 3 bloky, vďaka čomu nemusí hľadať všetky informácie v celom texte a zároveň pri správnom rozdelení vyriešime problémy ktoré sa v pôvodnom rozdelení objavili.

Výzor nového rozdelenia môžeme vidieť na obrázku 2.2. Vidíme, že jediná zmena ktorá nastala je taká, že sme bloky B až F spojili do jedného bloku s názvom B a pôvodný blok G sme premenovali na blok C.

Toto rozdelenie rieši všetky vyššie spomenuté problémy vďaka tomu, že o bloku A vieme, že vždy bude mať tvar dvojriadkovej hlavičky s pri skúšaní sme nenašli žiadnu výnimku, podobne pôvodný blok G čiže nový blok C nie je súčasťou samotnej prepúšťacej správy ale sú to krvné výsledky pacienta ktoré sú v našom vstupe vždy až za správou a majú tvar dvojíc testovaná veličina a výsledok testu (pozitivita alebo hod-



Obr. 2.2: Upravené rozdelenie správy do menšieho počtu väčších blokov

nota) vďaka čomu je jednoduché ich oddeliť od textu správy. Samotný text správy sa problematické pre ďalšie delenie preto ho nechávame pokope v bloku B.

Kapitola 3

Získavanie dát

V tejto kapitole si povieme niečo o jednotlivých získavaných dátach o problémoch ktorý sa pri ich získavaní vyskytli a ako sme tieto problémy riešili.

3.1 Osobné údaje pacienta a obdobie hospitalizácie

Ako prvé boli získavané dáta nachádzajúce sa v hlavičke dokumentu konkrétne išlo o meno pacienta, jeho rodné číslo dátumy prijatia a prepustenia. Zároveň sa v tejto časti dopočítavali 2 hodnoty a to vek pacienta a dĺžka hospitalizácie.

Hlavička má výzor tabuľky s dvomi riadkami a štyrmi alebo piatimi stĺpcami v závislosti od toho či bol pacient počas hospitalizácie preložený z infekčného oddelenia na jednotku intenzívnej starostlivosti (JIS) respektíve na iného oddelenie alebo nie. Konkrétne v prípade ak pacient nebol preložený na iné oddelenie počas hospitalizácie hlavička vyzerá ako na obrázku 3.1 a v prípade, že preložený bol tak vyzerá takto ako na obrázku 3.2.

Softvér z tejto tabuľky vyberie hodnoty z druhého riadka pričom v prípade, že v štvrtom stĺpci nájde informáciu o preklade na iné oddelenie tak dátum v tomto stĺpci ignoruje a za dátum prepustenia považuje dátum v piatom stĺpci.

Dátumy prijatia a prepustenia sú následne pre-typované do podoby časovej značky čím sa zároveň skontroluje platnosť dátumu (či taký dátum môže existovať) v prípade, že pri niektorom z dátumov vyskytne problém je táto informácia zapísaná do logovacieho súboru. V prípade, že sú obe dátumy v poriadku je následne spravený ich rozdiel čím sa vypočíta dĺžka hospitalizácie, čiže počet dní medzi prijatím a prepustením pri-

Meno	RČ	Dátum prijatia	Dátum prepustenia
Priezvisko Meno	XXXXXX/XXXX	DD.MM.YYYY	DD.MM.YY

Obr. 3.1: Výzor hlavička pacienta ktorý nebol preložený na JIS

Meno	RČ	Dátum prijatia	Dátum prepustenia	
Priezvisko Meno	XXXXXX/XXXX	DD.MM.YYYY	DD.MM.YY preklad	DD.MM.YY

Obr. 3.2: Výzor hlavička pacienta ktorý bol preložený na JIS

```
13.4.2020 rýchlotest SARS-CoV-2: protilátky IgM:  POZIT,  IgG:  POZIT

S-anti-SARS-CoV-2 IgM (ELISA)/1.12.2020/          55,5          NTU  pozitívny              (cut-off > 11)
S-anti-SARS-CoV-2 IgG (ELISA)/1.12.2020/          34,7          NTU  pozitívny              (cut-off > 11)

09.12.2020- PROTILÁTKY  SARS - CoV – 2:  IgM:  POZIT IgG:  POZIT

Rýchlotest na vyšetrenie protilátok proti SARS2-COV 12.4.2021:  IgM silno pozit, IgG slabo pozit

16.3.2021 Protilátky SARS-CoV-2: IgM:pozit., IgG: pozit

Rýchlotest na protilátky SARS CoV-12: IgM slabo pozit.       IgG negat.

25.4.2021 Protilátky COVID 19 rýchlotest z kapilárnej krvi: IgG: POZIT    IgM: NEGAT

26.1.2021 Rýchlotest na protilátky SARS-CoV2 (z kapilárnej krvi): IgM pozitívne, IgG pozitívne

PROTILÁTKY  SARS - CoV – 2:/5.1.21/.:IgM: POZIT,IgG: POZIT

Protilátky na SARS-Cov-2: 29.12. IgM, IgG negat., 5.1.21 IgM,IgG pozit.
```

Obr. 3.3: Rôzne spôsoby zápisu testov na protilátky v prepúšťacích správach

čom tento výsledok nám dáva ďalšiu kontrolu dátumov keďže očakávame, že dĺžka hospitalizácie je kladné číslo avšak nepredpokladáme, že to číslo bude veľmi veľké, konkrétne pri testovaní sa ukázalo, že priemerná dĺžka hospitalizácie je približne 12 dní pričom štandardná odchýlka je približne 9 dní avšak občas sa objavia aj prípady ktorých dĺžka hospitalizácie je cez 50 dní pričom nenastal žiaden preklep pri zapisovaní dátumom preto sme sa rozhodli použiť ako hornú hranicu hodnotu 100 ktorá pokrývala všetky naše doterajšie prípady.

Nakoniec v prípade, že dátum prijatia je v poriadku tak softvér určí z rodného čísla dátum narodenia a pomocou dátumu narodenia a dátumu prijatia vypočíta vek pacienta pri prijatí. Tento vek následne kontroluje či je v intervale 0 až 120, v prípade, že do tohto intervalu nepatrí indikuje to chyba buď v dátume prijatia alebo v rodnom čísle pacienta.

3.2 Protilátky proti vírusu SARS-CoV-2

V prípade hľadania výsledkov testov na protilátky sa ukázalo, že každý lekár ich zapisuje iným spôsobom pričom aj v správach jedného lekára sa nachádzajú rozdiel medzi jednotlivými zápismi. Niekoľko konkrétnych ukážok je na obrázku 3.3.

Preto sme pri hľadaní postupovali tak, že sme

Kapitola 4

Analýza výsledkov a chýb

Táto kapitola je zameraná na analýzu presnosti a chybovosti nášho systému.

Kapitola 5

Modifikácia a vylepšenia

Obsahom tejto kapitole sú úpravy a vylepšenia softvéru tak aby bol jednoduchší na používanie pre používateľa a aby ho bolo možné jednoducho modifikovať pre získavanie dát zo správ pacientov s iným ochorením ako je COVID-19 respektíve pre získavanie všeobecných informácii nachádzajúcich sa každej prepúšťacej správe.

Záver

Toto má ešte čas.

Literatúra

[1] Roni Romano, Lior Rokach, and Oded Maimon. Cascaded data mining methods for text understanding, with medical case study. In *Sixth IEEE International Conference on Data Mining-Workshops (ICDMW'06)*, pages 458–462. IEEE, 2006.

[2] Anshul Aggarwal, Sunita Garhwal, and Ajay Kumar. Hedea: a python tool for extracting and analysing semi-structured information from medical records. *Healthcare informatics research*, 24(2):148–153, 2018.

[3] Georg Fette, Maximilian Ertl, Anja Wörner, Peter Kluegl, Stefan Störk, and Frank Puppe. Information extraction from unstructured electronic health records and integration into a data warehouse. In *GI-Jahrestagung*, pages 1237–1251, 2012.

[4] Hanna M Wallach. Conditional random fields: An introduction. *Technical Reports (CIS)*, page 22, 2004.

[5] Menglin Cui, Ruibin Bai, Zheng Lu, Xiang Li, Uwe Aickelin, and Peiming Ge. Regular expression based medical text classification using constructive heuristic approach. *IEEE Access*, 7:147892–147904, 2019.

[6] Constructive heuristic. in wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Constructive%20heuristic&oldid=1032227019`, 2022. [Online; accessed 07-December-2022].

Príloha A: obsah elektronickej prílohy

Príloha B: Používateľská príručka