Slovenská technická univerzita

Fakulta informatiky a informačných technológii

Ilkovičova 3, 812 19 Bratislava

Projektová dokumentácia

Tím 08: WIM

Akademický rok: 2019/2020
Predmet: Tímový projekt

Študenti: Bc. Lenka Beňová
Bc. Martin Chreno
Bc. Štefan Lazový
Bc. Marek Sninčák
Bc. Christian Ščípa
Bc. Martin Tonhauzer
Bc. Viktor Valaštín

Vedúci tímu: doc. Ing. Tibor Krajčovič, PhD.

Obsah

Inžinierske dielo

Úvod

Tento dokument predstavuje dokumentáciu k inžinierskemu dielu tímu Neural Plates (tím číslo 8). Opisuje štruktúru existujúceho systému a moduly, ktoré boli v rámci tímového projektu vytvorené. Cieľom projektu je overiť použiteľnosť neurónových sieti pri výpočte VDR záznamov - informácií o vozidlách, ktorý je v súčasnom riešení počítaný zložitými algoritmami.

Technologický kontext:

Weight-in-motion je zariadenie určené na meranie a kategorizáciu vozidiel na cestách – počíta rýchlosť prechádzajúcich vozidiel, prípadne nerovnomerné rozloženie váhy na pravej a ľavej strane vozidla. Získané dáta sú následne uložené v internej databáze. K zariadeniu je taktiež možné pristupovať pomocou webového rozhrania, v ktorom môžeme vidieť zozbierané dáta, alebo aktuálnu konfiguráciu zariadenia.

1. Globálne ciele projektu

Cieľom projektu je navrhnúť, overiť a nasadiť produkt, ktorý by miesto deterministických algoritmov na výpočet hmotnosti vozidiel využíval na výpočet neurónovú sieť.

Globálne ciele projektu na zimný semester

Celkovým cieľom za zimný semester je vytvorenie čiastkového prototypu a príprava na ďalšie úspešné smerovanie projektu v letnom semestri. Keďže nevyvíjame systém od začiatku, ale pracujeme s existujúcim riešením, časť úsilia bude vynaložená na jeho prispôsobenie, aby vyhovovalo naším potrebám.

V zimnom semestri sme si stanovili nasledovné ciele (Plánovaný stav produktu k januáru 2020):

- Bude vykonaná analýza aktuálne nasadeného riešenia
- Bude vykonaná dátová analýza súborov a objektov používaných v produkcii zameraná na kľúčové atribúty pre trénovanie modelov
- Bude implementovaný spôsob zbierania dát
- Bude vytvorený prototyp časti riešenia

Globálne ciele projektu na letný semester

V zimnom semestri sa nám podarilo zoznámiť sa s existujúcim systémom a dátami potrebnými na vytvorenie riešenia, vytvoriť základné nástroje na zber dát a vytvoriť čiastočný prototyp riešenia, schopný vypočítať hmotnosť vozidla. Hlavným cieľom v letnom semestri je vytvoriť z tohto prototypu ucelený produkt, ktorý bude možné nasadiť do prevádzky. Tento cieľ sa počas semestra rozšíril o rozšírenie vytvoreného produktu o rozpoznávanie kategórie vozidla. Zároveň sme v letnom semestri začali pracovať na rozšírení webového rozhrania slúžiaceho na zobrazenie informácii zo zariadenia. Tu bolo cieľom zlepšiť interaktivitu a vylepšiť funkcionalitu súvisiacu s údajmi počítanými naším hlavným produktom.

2. Celkový pohľad na systém

V tejto časti je uvedený celkový pohľad na náš systém. Keďže rozširujeme existujúci systém, je tu priblížený aj ten. Aktuálny systém získava informácie o vozidle získané zo senzorov ukrytých vo vozovke. Dáta sa následne spracovávajú algoritmicky. Cieľom projektu bolo vytvoriť algoritmus, ktorý tieto dáta (primárne váhu vozidla) vypočíta pomocou neurónovej siete. Na uskutočnenie nášho riešenia bolo okrem vytvorenia samotnej neurónovej siete tiež potrebné upraviť existujúci systém tak, aby bol schopný posielať údaje prúdom (anglicky stream) cez rozhranie WebSocket a poskytoval údaje potrebné na výpočet a porovnanie výsledkov. Počas riešenia tiež vzniklo viacero pomocných nástrojov na analýzu, či spracovanie dát.

Pre zvýšenie používateľského zážitku sme rozšírili aj funkcionalitu existujúceho webového rozhrania, kde sme implementovali viacero funkčných požiadaviek ako je stránkovanie záznamov, drag & drop funkcionalita na nastavovanie pozície senzorov na ceste, alebo vizualizáciu kategórie vozidla.

2.1 Pôvodný systém

Táto časť poskytuje celkový pohľad na architektúru systému, ktorý rozširujeme. Z dôvodu, že obsahuje citlivé dáta, sa v tejto verzii nenachádza.

2.2 Rozšírenie existujúceho systému

Táto časť popisuje zmeny, ktoré boli v pôvodnom systéme vykonané. Z dôvodu, že obsahuje citlivé dáta, sa v tejto verzii nenachádza.

2.3 Výpočet parametrov vozidiel neurónovou sieťou

Táto časť popisuje výpočet parametrov vozidiel vykonávaný neurónovou sieťou. Z dôvodu, že obsahuje citlivé dáta, sa časť z nej v tejto verzii nenachádza.

Vstupom neurónovej siete sú sekvencie záznamov fixnej dĺžky. Pre dodržanie fixnej dĺžky boli sekvencie dopĺňané potrebným počtom nulových hodnôt.

Natrénovaný model je zabalený do samostatnej služby, ktorá prijíma správy o nových vozidlách na senzoroch a tie následne spracuje, aplikuje na nich modely neurónových sietí a výsledky uloží do súboru formátu sqlite3 na lokálnom systéme.

2.4 Rozšírenie frontendu

Drag & drop funkcionalita

Jednou z funkcionalít, ktorú poskytuje frontend je možnosť nastavenia rozloženia jednotlivých senzorov, pričom toto rozloženie je na obrazovke vizualizované. V rámci pôvodného riešenia bolo možné toto rozloženie nastaviť iba zadaním parametrov do príslušných polí. Pre zjednodušenie používania a dosiahnutie lepšej interaktivity aplikácie sme túto funkcionalitu rozšírili o možnosť pohybovať s force a presence senzormi ťahom myšou. V závislosti od typu senzora je možné vykonávať vertikálny a horizontálny pohyb, prípadne nastaviť uhol náklonu senzoru. Zároveň je možné senzor presunúť medzi jednotlivými jazdnými pruhmi alebo koordinačnými systémami.

Vizuálna reprezentácia kategórii vozidiel

Nasledujúcou pridanou funkcionalitou je zobrazovanie piktogramov vozidiel na základe mapovania interných ID vozidiel voči oficiálnej európskej smernici na klasifikáciu vozidiel EUR

13. Takto je možné aby užívateľ, ktorý nemá znalosti v automobilovej doméne, bol schopný na zaklade piktogramu identifikovať o aké vozidlo sa jedná.

História vozidiel - stránkovanie

Ďalšou zmenou bolo pridanie stránkovania na webovú stránku obsahujúcu históriu prechodov vozidiel senzormi. Pôvodne táto stránka zobrazovala maximálne 200 záznamov. Ak by používateľ chcel vyhľadať ďalšie staršie záznamy, musel by podľa toho upraviť filtrovanie. Pridaním stránkovania sme umožnili jednoduché prechádzanie ďalšími výsledkami tlačidlami vpred a späť alebo priamo načítať ľubovoľnú stránku podľa čísla strany.

2.5 Pomocné nástroje

Na realizovanie našej úlohy bolo nutné vytvoriť viacero pomocných nástrojov slúžiacich na zber a prípravu dát. Tieto nástroje sú bližšie opísané v časti Moduly systému.

2.6 Ohraničenia

Systém predikcie je nutné prevádzkovať na oddelenom zariadení, narozdiel od algoritmického výpočtu, pretože Keras framework nepodporuje architektúru procesora zberného zariadenia.

Neurónová sieť bola trénovaná na dátach, ktoré vypočítal algoritmus a nie na reálnych hmotnostiach vozidiel. Toto obmedzilo naše riešenie tým, že maximálna dosiahnutá presnosť sa mohla rovnať iba presnosti dosiahnutej algoritmickým riešením, pričom pri výpočte hmotnosti algoritmus zaokrúhľoval hmotnosti na desiatky kg. Presnosť kategorizácie vozidiel bola negatívne ovplyvnená nízkym počtom nameraných vozidiel špeciálnych kategórií a teda malým počtom trénovacích dát pre tieto kategórie.

3. Moduly systému

3.1 Analýza

V tejto časti je opísaná analýza existujúceho riešenia, analýza vytváranej neurónovej siete a analýza časti funkcionality webového rozhrania.

Analýza existujúceho riešenia

Pre vytvorenie nášho riešenia bolo nutné oboznámiť sa s aktuálnym systémom, pričom výstup tejto analýzy je opísaný v časti 2 tohto dokumentu. Zároveň bolo nutné vykonať analýzu dát, s ktorými môžeme pracovať. Ide konkrétne o FPT a VDR záznamy. Analýzy týchto záznamov sú dostupné v prílohách.

Analýza neurónovej siete

Neurónová sieť má nahradiť aktuálny algoritmický výpočet VDR záznamu z FPT záznamov. Vzor VDR a FPT záznamov vo formáte JSON sa nachádza v prílohe

[Príloha B-3: FTP záznamy]

[Príloha B-4: VDR záznamy]

Zoznámili sme sa a analyzovali sme bežne používané typy neurónových sietí:

- **Dopredná neurónová sieť - MLP**
  - Výhody:
    - jednoduchosť
    - rýchlosť
  - Nevýhody:
    - nutnosť vstupu rovnakej dĺžky

- **Rekurentná neurónová sieť - RNN**
  - Výhody:
    - Dynamická dĺžka vstupu
  - Nevýhody:
    - zložitosť siete - malá rýchlosť v porovnaní s MLP

Zároveň sme analyzovali dostupné knižnice. Preferovanou voľbou, v ktorej bude vytvorený prototyp sú knižnice TensorFlow a Keras, ktoré sú najrozšírenejšie a niektorí členovia s nimi majú skúsenosti.

Zariadenie, na ktorom bude riešenie nasadené, je postavené na architektúre ARM a využíva vlastnú distribúciu založenú na operačnom systéme Linux. Momentálne sa nám na tomto zariadení podarilo spustiť knižnicu OpenNN, ktorej vhodnosť bola tiež analyzovaná. Jej použitie sa ukázalo ako nie veľmi vhodné a hlavnou voľbou ostáva spomínaná knižnica TensorFlow.

Analýza frontendu - drag & drop

Analýza vzťahujúca sa k drag & drop funkcionalite zahŕňala najmä vybratie vhodnej knižnice umožňujúcej implementáciu pohybu komponentov myšou do pôvodného systému a neskôr analýzu výkonu.

Pri výbere vhodnej knižnice sme sa rozhodli využiť podobné knižnice, aké boli v rámci pôvodného projektu využité. Na základe tohto kritéria sa ako vhodná knižnica ukázala knižnica d3-drag.

Po implementácii prvotnej funkcionality sme sa zamerali na zlepšenie výkonu, keďže v určitých prípadoch nebol pohyb komponentu plynulý. Na základe tejto analýzy sa ako možné riešenia ukázali:

- využitie React.PureComponent namiesto React.Component na zníženie počtu zbytočných vykreslení komponentu
- využitie shouldComponentUpdate na zníženie počtu zbytočných vykreslení komponentu
- využitie funkcie requestAnimationFrame na vynútenie aktualizácie animácie pri vykreslení zmenenej stránky

3.2 Návrh

V tejto časti je opísaný návrh vytváranej neurónovej siete. Je tu uvedených viac možností implementácie, pričom možnosťou, ktorú sme vo finálnom riešení využili je možnosť 2.

Vstup neurónovej siete

Táto časť opisuje dáta, ktoré budú vstupovať do neurónovej siete. Z dôvodu, že obsahuje citlivé dáta, sa v tejto verzii nenachádza

Výstup neurónovej siete

Výstupom by mal byť záznam, obsahujúci hodnoty vypočítané navrhovanou neurónovou sieťou.

Možnosť 1 - MLP + RNN:

Táto časť opisuje možnosť riešenia problému použitím kombinácie MLP a RNN sietí. Z dôvodu, že obsahuje citlivé dáta, sa v tejto verzii nenachádza.

Možnosť 2 – RNN:

Táto časť opisuje možnosť riešenia problému použitím samotnej RNN siete. Z dôvodu, že

obsahuje citlivé dáta, sa v tejto verzii nenachádza.

Možnosť 3 – MLP:

Táto časť opisuje možnosť riešenia problému použitím samotnej MLP siete. Z dôvodu, že obsahuje citlivé dáta, sa v tejto verzii nenachádza.

Alternatívne formy vstupov:

Táto časť opisuje alternatívy, ktoré môžu byť vstupom neurónovej siete. Z dôvodu, že obsahuje citlivé dáta, sa v tejto verzii nenachádza.

3.3 Prepojenie s existujúcim systémom

Táto časť popisuje, ako bolo naše riešenie napojené na existujúci systém. Z dôvodu, že obsahuje citlivé dáta, sa v tejto verzii nenachádza.

3.4 Implementácia

Implementačná časť nášho riešenia pozostávala z viacerých častí – bolo nutné rozšíriť existujúci systém, implementovať podporné nástroje a implementovať prototyp samotnej neurónovej siete.

Rozšírenie existujúceho systému

Táto čať opisuje spôsob rozšírenia existujúceho systému. Z dôvodu, že obsahuje citlivé dáta, sa v tejto verzii nenachádza.

Modul výpočtu hmotnosti - MLP prototyp

Modul výpočtu hmotnosti bude modulom, ktorý nahradí algoritmický výpočet údajov o vozidle v pôvodnom riešení. V zimnom semestri sa nám podarilo vytvoriť jeho návrh a základný prototyp.

Tento prototyp neurónovej siete bol vytvorený pomocou frameworkov Tensorflow a Keras. Neurónová sieť brala na vstupe upravené Force záznamy a ako výstupnú hodnotu brala vypočítané hmotnosti pre príslušné Force záznamy.
Implementáciu sme sa rozhodli riešiť pomocou doprednej neurónovej siete Multi-layer perceptron. Samotná architektúra pozostáva z N Dense vrstiev rovnakého typu a jednej Dense vrstvy ktorá produkuje výstup siete.

Na trénovanie siete a určenie optimálnej architektúry sme použili trénovanie pomocou zmeny hyperparametrov. Generátor hyperparametrov používal nasledovné nastavenia:

Batch size = 2^3 - 2^3

Dense size = 2^3 - 2^9

Learning rate = 10^-2 – 10^-4

Activation function = relu / linear / tanh / sigmoid

Number of layers = 1 – 10

Samotný proces trénovania bol logovaný a analyzovaný pomocou nástroja tensorboard. Výsledná Dense vrstva nemenila konfiguráciu a zostávala nastavená na veľkosti 1 s akvitačnou funkciou linear. Toto nastavenie sme zvolili, nakoľko neurónová sieť predpovedala hmotnosť a teda jedno číslo. Táto aktivačná funkcia neobmedzovala neurónovú sieť žiadnym skreslením výsledku a dovolila jej predpovedať akékoľvek hodnoty.

Z tohto prototypu sme plánovali vychádzať aj v letnom semestri, kde MLP sieť mala počítať hmotnosti jednotlivých kolies a tieto mali byť následne spracovávané sekvenčne rekurentnou neurónovou sieťou. Ukázalo sa však, že samotná rekurentná neurónová sieť dokázala pracovať aj s dátami, ktoré boli využité ako vstup do MLP siete, pričom dosiahnuté výsledky neboli horšie a tak sme v letnom semestri vytvorili riešenie využívajúce samotnú rekurentnú neurónovú sieť.

Modul výpočtu hmotnosti – RNN

Tento modul nahrádza algoritmický výpočet hmotnosti vozidla rekurentnou neurónovou sieťou.

Pri implementácií neurónovej siete sme použili frameworky Tensorflow a Keras. Neurónová sieť brala ako vstup sekvencie Force záznamov, ktoré patrili k jednodlivým vozidlám. Pri trénovaní sme predikovanú hodnotu hmotnosti validovali voči hmotnosti získanej z VDR záznamov pre toto vozidlo. Výstupom neurónovej siete bola už celková hmotnosť vozidla.

Implementovali sme rekurentnú neurónovú sieť – RNN. Návrh siete obsahoval niekoľko LSTM vrstiev, s výstupnou Dense vrstvou.

Na stanovenie výslednej architektúry siete a jej parametrov, sme použili trénovanie pomocou generátora hyperparametrov. Najoptimálnejšou sa ukázala byť kombinácia jednej LSTM vrstvy s 32 neurónmi a aktivačnou funkciou ReLU a výstupnej Dense vrstvy s veľkosťou 1 a aktivačnou funckiou ReLU.

Neurónová sieť používala ako vstup sekvencie fixnej dĺžky, ktoré boli rozšírené nulami v prípade kratších vstupných sekvencií.

Pri trénovaní dosahovala neurónová sieť odchýlku približne 5% z celkovej váhy vozidla.

Modul kategorizácie vozidiel

Tento modul nahrádza algoritmický výpočet kategórie vozidla použitím rekurentnej neurónovej siete.

Pri implementácií neurónovej siete sme opäť použili frameworky Tensorflow a Keras. Neurónová sieť brala ako vstup sekvencie Force záznamov, ktoré patrili k jednotlivým vozidlám rozšírené o vypočítaný parameter s počtom náprav vozidla. Pri trénovaní sme predikovanú hodnotu kategórie validovali voči kategórií získanej z VDR záznamov pre toto vozidlo. Výstupom neurónovej siete bol vektor pravdepodobností, s ktorými patrí vozidlo do jednotlivých kategórií.

Implementovali sme rekurentnú neurónovú sieť – RNN. Návrh siete obsahoval niekoľko LSTM vrstiev, s výstupnou Dense vrstvou.

Na stanovenie výslednej architektúry siete a jej parametrov, sme použili trénovanie pomocou generátora hyperparametrov. Najoptimálnejšou sa ukázala byť kombinácia jednej LSTM vrstvy s 32 neurónmi a aktivačnou funkciou ReLU a výstupnej Dense vrstvy s veľkosťou rovnou počtu kategórií a aktivačnou funckiou Softmax.

Neurónová sieť používala ako vstup sekvencie fixnej dĺžky, ktoré boli rozšírené nulami v prípade kratších vstupných sekvencií. Kontrolný výstup bol transformovaný z jedného číselného atribútu do vektora binárnych hodnôt s rovnakou dĺžkou ako bol počet kategórií, kde jednotka bola v prípade, že vozidlo patrilo do danej kategórie. Výstup neurónovej siete bol v rovnakom formáte vektora pravdepodobností. Ako výsledná kategória sa brala tá s najvyššou pravdepodobnosťou.

Pri trénovaní dosahovala neurónová sieť predikovala správnu kategóriu pre 99% vozidiel.

Modul na zbieranie prúdu dát VDR

Na natrénovanie neurónovej siete bolo nutné zozbierať reálne dáta. Za týmto účelom bol vytvorený modul na zbieranie prúdu dát VDR, ktorý zabezpečuje zber VDR záznamov pomocou protokolu websocket. Aktuálna štruktúra VDR obsahuje všetky potrebné údaje pre modelovanie prvého prototypu neurónovej siete.

Tento modul ukladá na disk jednotlivé záznamy VDR, ktoré budú v ďalších krokoch spracované na formát vhodný pre trénovanie modelu. Technická dokumentácia je priložená v prílohe [Príloha B-2: Zbieranie dát VDR]

Modul na parsovanie a filtrovanie nazbieraných VDR dát

Príprava trénovacích dát pre neurónovú sieť vyžadovala vytvorenie nástroja na parsovanie všetkých .json VDR záznamov do jedného .csv súboru, ktorý bol vstupom do neurónovej siete. Tento súbor obsahoval polia potrebné pre výpočet váhy auta, ktoré sa nachádzali vo force záznamoch samotného VDR záznamu. Okrem parsovania bolo potrebné vyfiltrovať chybné dáta, ktoré obsahovali záporné hodnoty, prázdne polia alebo nesprávny prejazd vozidla cez jednotlivé senzory (napr. prejazd viacerých kolies tým istým senzorom). Vyfiltrované boli aj vozidlá prechádzajúce senzormi rýchlosťou menšou ako 20 km/h.

Táto časť opsiuje spôsob filtrovania VDR záznamov. Z dôvodu, že obsahuje citlivé dáta, sa v tejto verzii nenachádza.

Technická dokumentácia k nástroju je priložená v prílohe [Príloha B-5: VDR parser].

Modul spracovania dát

Modul spracovania zozbieraných dát funguje na rovnakom princípe ako vyššie opísaný modul na parsovanie a filtrovanie nazbieraných VDR dát s tým rozdielom, že pracuje už so samotnými FTP záznamami, ktoré sú získavané z rozšírenia systému o stream FTP záznamov z websocketu. Modul sa pripája na websocket zariadenia, z ktorého zbiera dáta. Tieto dáta (FTP záznamy obohatené o ID vozidla) sú ukladané do výstupného CSV súboru obsahujúceho polia potrebné pre výpočet hmotnosti vozidla, ako aj jeho kategorizácie do váhovej skupiny. Filtrovanie FTP záznamov bolo upravené, pričom obsahovalo všetky typy vozidiel, vrátane vozidiel nad 3.5 tony. Modul bol neskôr integrovaný priamo na zariadenie, pričom spracovával práve prichádzajúce FTP záznamy, ktoré z modulu prúdili priamo do neurónovej siete vykonávajúcej kategorizáciu a váženie.

Táto časť opsiuje polia nachádzajúce sa v FTP záznamoch. Z dôvodu, že obsahuje citlivé dáta, sa v tejto verzii nenachádza.

Frontend - drag & drop

Na implementáciu drag & drop funkcionality opísanej v predchádzajúcej kapitole sme využili knižnicu d3-drag, ktorá bola kompatibilná s existujúcim riešením. Keďže komponent

zabezpečujúci vizualizáciu sa nachádzal v internej knižnici, bola táto zmena implementovaná tam. V našom prípade sme nemali prístup na nahranie novej verzie do interného registra knižníc a tak sme zvolili riešenie využitia tohto modulu ako lokálneho modulu.

Na dosiahnutie plynulých animácii sa ako vhodná voľba ukázalo využitie funkcie requestAnimationFrame, ktorá slúži na vynútenie aktualizácie animácie pri vykreslení zmenenej stránky. Implementácia ostatných spôsobov zlepšenia výkonu v našom prípade neviedla k zlepšeniu výkonu.

3.5 Nasadzovanie

Proces nasadzovania sa skladá z dvoch častí:

1) Nasadzovanie našej verzie softvéru na senzory.
    a) Je potreba prekompilovať softvér pomocou skriptu dodaného externou firmou a nahradiť súbory na vzdialenom zariadení.
2) Spojazdnenie servera s neurónovou sieťou
    a. Treba mať nainštalovaný Python3.7
    b. Stiahnuť závislosti pre skript neurónovej siete
    c. Upravenie konfiguračných súborov
    d. Prevádzka
    e. Výsledky sa uložia do sqlite súboru.

3.6 Testovanie

Testovanie bolo vykonané najmä pre vytváraný modul neurónovej siete. Vyplynulo, že najlepšia konfigurácia siete boli 3 Dense vrstvy, s aktivačnou funkciou relu, veľkosti 32 a rýchlosťou učenia 0,001. Použitý bol optimizer Adam. Najlepší dosiahnutý model mal priemernú odchýlku 0,2% od originálnej hodnoty hmotnosti.

Nasadili sme naše riešenie na zariadenie poskytnuté externou firmou Kistler a porovnali sme databázu nášho a pôvodného riešenia.

Finálne testovanie sme spravili v prevádzke. Firma Kistler nám poskytla zariadenie vo Švajčiarsku, kam sme nahrali tam náš vlastný softvér, tak ako je opísané v časti Nasadzovanie.

Vyhodnotenie prebehlo na dátach zozbieraných v priebehu 24 hodín porovnaním databáz pôvodného a nami vytvoreného riešenia. Tieto výsledky boli aj zaznamenané do tabuľky, ktorá vyzerala nasledovne:

id	weight_nn	cat_nn	is_success	weight_orig	cat_orig	weight_difference	weight_acc
72	2818.398193	1	1	2790	1	28.39819336	0.010178564
73	1573.595459	1	1	1520	1	53.59545898	0.03526017
74	1870.903198	1	1	1900	1	29.09680176	0.015314106
75	1318.065796	1	1	1300	1	18.0657959	0.013896766
76	1370.268677	1	1	1310	1	60.26867676	0.046006623
77	1858.712891	1	1	1890	1	31.28710938	0.016554026
78	1718.157959	1	1	1670	1	48.15795898	0.028837101
79	2245.224854	1	1	2170	1	75.22485352	0.034665831
80	2047.474609	1	1	2070	1	22.52539063	0.010881831
81	1393.495605	1	1	1350	1	43.49560547	0.032218967
82	1379.317993	1	1	1310	1	69.31799316	0.052914499
83	2630.94043	1	1	2630	1	0.940429688	0.000357578
84	1696.302368	1	1	1690	1	6.302368164	0.003729212
85	1953.873535	1	1	1910	1	43.87353516	0.022970437
86	1798.038452	1	1	1820	1	21.96154785	0.012066785
87	1244.981689	1	1	1260	1	15.01831055	0.011919294
88	1554.184937	1	1	1540	1	14.18493652	0.009210998
89	2741.719482	1	1	2790	1	48.28051758	0.017304845
90	2111.957031	1	1	2050	1	61.95703125	0.030222942
91	2569.137695	1	1	2540	1	29.13769531	0.011471534
92	6223.010254	1	1	6370	1	146.9897461	0.023075313
93	1715.171265	1	1	1700	1	15.17126465	0.008924273
94	1973.331177	1	1	1930	1	43.33117676	0.022451387

V spomenutej tabuľke môžeme vidieť stĺpce s váhou a kategóriou určenou pôvodným (weight_orig, cat_orig) a naším riešením (weight_nn, cat_nn). Zároveň sú tu stĺpce zobrazujúce rozdiel medzi vypočítanými váhami v kilogramoch aj v percentách premietnutých do intervalu <0, 1> (weight_difference, weight_acc).

Štatistika nad percentuálnou odchýlkou (premietnutá do intervalu <0, 1>) vyzerá nasledovne:

- count 12372.000000
- mean 0.037375
- std 0.063819
- min 0.000000
- 10% 0.003095
- 25% 0.008157
- 50% 0.018821

- 75% 0.035504
- 90% 0.067576
- 95% 0.186061
- 99% 0.319825
- max 0.833702

Môžme vidieť, že priemerná chyba na množine 12372 vozidiel bola 3.7%.

V prípade frontendu sa využívali jednotkové testy, pričom v tomto prípade išlo hlavne o prispôsobenie existujúcich testov. Na vytváranie týchto testov bola využitá knižnica Jest.

Príloha B-1: Inštalačná a používateľská príručka pre ftp-processing

Jedná sa o službu zaobalujúcu spracovanie záznamov áut a ich vyhodnotenie pomocou modelu neurónovej siete.

Požiadavky:

- python vo verzii 3.7 alebo vyššie
- supervisor (pre prevádzku ako služby)
- virtualenv (odporúčané)

Inštalácia:

1. Uistite sa pri klonovaní zdrojového repozitára, že rekurzívne sklonujete aj závislosti projektu - git submodules. Príklad git príkazu, ktorý zabezpečí potrebné závislosti:
   `git clone --recurse-submodules <REPO URL>`
2. vytvorte nové virtualenv, napríklad `virtualenv tfp`
3. aktivujte virtualenv a spustite príkaz `pip install -r requirements.txt`
4. nainštalujte závislosti aj pre `models` submodul:
   `pip install -r models/requirements.txt`
5. upravte súbor `` `ftp_data_processing.conf.json `` podľa nasadenia
6. vytvorte súbor `secret` obsahujúci heslo.
7. Vytvorte supervisor konfiguráciu:
   a. pre pokyny je potrebné sa riadiť oficialnou dokumentáciou nástroja.
   b. nasledovné sú odporúčane parametre konfigurácie. Hodnoty medzi `<>` znakmi je nutné upraviť podľa nasadenia.

```
[program:ftp-processing]
autorestart = true
autostart = true
startretries = 3
startsecs = 10
user = nobody
directory = <DIRECTORY_WITH_THIS_SCRIPT>
```

```
command = <PATH_TO_VIRTUALENV_PYTHON_BINARY> ftp_data_processing.py
```

Použitie:

Spusti súbor skriptu `ftp_data_processing.py` buď pomocou supervisor nástroja alebo ručne. Skript je potrebné spúšťať z adresára, v ktorom sa nachádza. Skript hľadá po spustení súbor `ftp_data_processing.conf.json` v adresári spustenia. Jednotlivé parametre konfiguračného objektu znamenajú:

- `ftp_source_ws` : lokácia zdrojového websocketu, v socket notácií (IP:PORT)
- `ftp_source_login`: prihlasovacie meno pre websocket
- `ftp_source_secret_file`: súbor s heslom pre prihlásenie k websocket
- `tf_weight`: konfig pre neurónovú sieť pre počítanie váhy
- `tf_category`: konfig pre neurónovú sieť pre počítanie kategórie vozidiel
- `output_db_file`: súbor s sqlite3 databázov, pre ukladanie výsledkov

`tf_weight` a `tf_category` sú objekty ktoré obsahujú následujúce atribúty:

- `model`: súbor s predtrénovaným modelom
- `parsers`: python skript v ktorom sa hľadáju metódy `input_preprocess` a `output_postprocess` pre fungovanie modelu

Niektoré z vyššie uvedených parametrov majú predvolené hodnoty, pre použitie s dodanými modelmi neurónovej siete.

Získanie výsledkov

Vypočítané údaje sa nachádzajú v súbore formátu sqlite3, ktorý bol špecifikovaný pri nastaveniach opísaných vyššie. K údajom sa dá pristupovať štandardnými SQL nástrojmi podporujúcimi sqlite3. Všetky výsledky sa nachádzajú v tabuľke opísanej príkazom:

```
CREATE TABLE IF NOT EXISTS weights(
  id integer PRIMARY KEY,
  weight real,
  MappedClassCategoryID integer,
  is_success integer NOT NULL
)
```

Príloha B-2: Zbieranie dát VDR

VDR-collector

Used to collect VDR from device

dependencies

- NodeJS 12

usage

```
cd VDR-collector
npm install
node src/collector.js ADDRESS -a AUTH -p WS PORT [-o OUTPUT FOLDER] [-r
NUMBER OF RETRIES]
```

parameters

- ADDRESS device IP address
- -a AUTH authentication string in format USERNAME:PASSWORD
- -p WS PORT VDR stream port
- -o OUTPUT FOLDER (optional) folder to store output files
- -r NUMBER OF RETRIES (optional) number of retries in case of failed network operations

examples

```
node src/collector.js 182.108.142.232 -a user123:mypassword -p 12345
node src/collector.js 182.108.142.232 -a user123:mypassword -p 12345 -o
C:\Users\user123\Desktop\out
node src/collector.js 182.108.142.232 -a user123:mypassword -p 12345 -o
C:\Users\user123\Desktop\out -r 5
```

output

Data output is saved to */data* folder by default. One file corresponds to one VDR record.

Filename convention: *vdr-%timestamp in ms%.json* .

Príloha B-3: FTP záznamy

V tejto prílohe je opísaná štruktúra FTP záznamu. Z dôvodu, že sa jedná o citlivé dáta, sa v tejto verzii nenachádza

Príloha B-4: VDR záznamy

Táto príloha ukazuje štruktúru VDR záznamu. Z dôvodu, že sa jedná o citlivé dáta, sa v tejto verzii nenachádza.

Príloha B-5: VDR parser

Used to parse VDR .json files to .csv

usage

```
usage: data_processing.py [-h] [--count COUNT] input_path output_path
```

arguments

```
positional arguments:
  input_path     Specify the input directory path from which the json data
                 will be parsed
  output_path    Specify the output path for the csv file

optional arguments:
  -h, --help     show this help message and exit
  --count COUNT  Specify the count of json data to be parsed
```

examples

```
python data_processing.py /my/path/to/dir /my/path/to/file.csv
python data_processing.py /my/path/to/dir /my/path/to/file.csv --count mynum
python data_processing.py -h
```

Príloha B-6: Neurónová sieť

NN-prototype

Folder structure:

- /data - training datasets
- /model_category - categorical NN models
- /model_category/saved_models - saved trained categorical models
- /model_weight - weight NN models
- /model_weight/saved_models - saved trained weight models

Usage:

Categorical NN

Run python file for NN model.

```
python -i category.py
```

After initialization model can be fitted using function fit_cat() inserting number of epochs as parameter.

After fitting is done function check_cat() can be called which lists all mispredicted values.

Weight NN

Run python file for NN model.

```
python -i weight.py
```

After initialization model can be fitted using function fit_weight() inserting number of epochs as parameter.

After fitting is done function check() can be called which lists all predicted and real values and average absolute error per dataset.

Riadenie projektu

Úvod

Tento dokument popisuje priebeh tímového projektu tímu Neural Plates (tím číslo 8) a poskytuje prehľad procesov, ktoré boli vykonávané. Obsahuje priblíženie práce jednotlivých členov tímu, opis aplikovaných manažérskych procesov, priebeh jednotlivých šprintov aj celkovú retrospektívu.

1. Role členov tímu a podiel práce

Táto sekcia opisuje role členov v tíme a podiely práce jednotlivých členov v uplynulých obdobiach.

Role členov tímu

V našom tíme používame dva druhy rolí - všeobecné a manažérske role. Všeobecná rola vyjadruje, čomu sa v tíme člen primárne venuje. Manažérska rola vyjadruje, aké manažérske procesy rieši. Nasledujúca tabuľka opisuje rozdelenie rolí medzi členov tímu:

Meno	**Všeobecná rola**	**Manažérska rola**
Lenka Beňová	backend developer	manažér testovania
Martin Chreno	špecialista na neurónové siete	manažér verziovania
Štefan Lazový	C++ developer	manažér kvality C++ kódu
Marek Sninčák	scrum master, full stack developer	manažér komunikácie
Christián Ščípa	frontend developer	manažér úloh
Martin Tonhauzer	devops	manažér dokumentácie
Viktor Valaštín	návrhár UX/UI	manažér UX/UI

Podiel práce

V jednotlivých šprintoch boli nasledovné podiely práce:

Šprint 1:

Meno	**Podiel práce**
Lenka Beňová	15%
Martin Chreno	10%
Štefan Lazový	14%
Marek Sninčák	16%
Christián Ščípa	16%
Martin Tonhauzer	12%
Viktor Valaštín	17%

Šprint 2:

Meno	**Podiel práce**
Lenka Beňová	10%
Martin Chreno	16%
Štefan Lazový	8%
Marek Sninčák	20%
Christián Ščípa	16%
Martin Tonhauzer	18%
Viktor Valaštín	12%

Šprint 3

Meno	**Podiel práce**
Lenka Beňová	17%
Martin Chreno	16%
Štefan Lazový	20%
Marek Sninčák	10%
Christián Ščípa	10%
Martin Tonhauzer	12%
Viktor Valaštín	13%

Šprint 4

Meno	**Podiel práce**
Lenka Beňová	16%
Martin Chreno	18%
Štefan Lazový	14%
Marek Sninčák	10%
Christián Ščípa	10%
Martin Tonhauzer	16%
Viktor Valaštín	16%

Šprint 5

Meno	**Podiel práce**
Lenka Beňová	12%
Martin Chreno	14%
Štefan Lazový	14%
Marek Sninčák	18%
Christián Ščípa	18%
Martin Tonhauzer	12%
Viktor Valaštín	12%

Šprint 6

Meno	**Podiel práce**
Lenka Beňová	15%
Martin Chreno	10%
Štefan Lazový	14%
Marek Sninčák	16%
Christián Ščípa	16%
Martin Tonhauzer	12%
Viktor Valaštín	17%

Šprint 7

Meno	**Podiel práce**
Lenka Beňová	10%
Martin Chreno	16%
Štefan Lazový	8%
Marek Sninčák	20%
Christián Ščípa	16%
Martin Tonhauzer	18%
Viktor Valaštín	12%

Šprint 8:

Meno	**Podiel práce**
Lenka Beňová	17%
Martin Chreno	16%
Štefan Lazový	20%
Marek Sninčák	10%
Christián Ščípa	10%
Martin Tonhauzer	12%
Viktor Valaštín	13%

Šprint 9

Meno	**Podiel práce**
Lenka Beňová	16%
Martin Chreno	18%
Štefan Lazový	14%
Marek Sninčák	10%
Christián Ščípa	10%
Martin Tonhauzer	16%
Viktor Valaštín	16%

Šprint 10

Meno	**Podiel práce**
Lenka Beňová	12%
Martin Chreno	14%
Štefan Lazový	14%
Marek Sninčák	18%
Christián Ščípa	18%
Martin Tonhauzer	12%
Viktor Valaštín	12%

2. Aplikácie manažmentov

Táto sekcia opisuje jednotlivé procesy realizované v našom tíme.

2.1 Komunikácia

Na komunikáciu v rámci tímu a komunikáciu s product ownermi využívame mimo osobných stretnutí nástroj Slack. Máme vytvorených viacero kanálov, ktoré zabezpečujú prehľadnosť a zvyšujú efektivitu komunikácie. Kanály a ich účel je podrobne opísaný v metodike komunikácie [Prílohy A-4: Metodika komunikácia]

Ďalej máme v tomto nástroji naplánované aj pravidelné stand-upy, okrem tých vykonávaných osobne, čo nám zabezpečí lepší prehľad o aktuálnom stave úloh a umožní lepšie plánovať ďalšie kroky.

2.2 Plánovanie

Na plánovanie, zadávanie a kontrolu priebehu práce využívame nástroj Jira. V tomto nástroji udržiavame produktový backlog, v ktorom máme ohodnotené príbehy (stories, na ich ohodnotenie využívame metodiku planning poker) a z ktorého vyberáme príbehy do nasledujúceho šprintu.

Na začiatku každého šprintu prebieha plánovanie, kde vyberieme príbehy, ktoré budeme v tomto šprinte riešiť, z produktového backlog-u a ak to je potrebné, tak sa vytvoria podúlohy, ktoré budú viesť k ich splneniu (pri menších príbehoch vytvárame len jednu podúlohu, aby nebolo nutné venovať priveľa času samotnému manažmentu). Takisto sú automaticky pridané prípadné nesplnené úlohy z predchádzajúceho šprintu.

Najdôležitejšie úlohy si potom rozdelíme medzi členov tímu. Niektoré menej kritické úlohy tak môžu ostať na začiatku šprintu nepridelené, pričom pridelené budú počas trvania šprintu. Tento krok nám zabezpečí lepšiu flexibilitu.

Pomocou tohto nástroja aj zaznamenávame stav jednotlivých úloh. Úloha môže mať nasledovný stav:

- To Do – úloha, ktorá bola vytvorená a ešte sa na nej nezačalo pracovať
- In Progress – úloha, na ktorej sa pracuje a ešte nie je dokončená
- Review – úloha, ktorá bola dokončená a čaká na vykonanie prehliadky kódu

- Done – úloha, ktorá bola dokončená a spĺňa Definition of Done

Po každom stretnutí exportujeme stav úloh. Tento proces opisuje metodika exportu úloh [Prílohy A-3: Metodika exporty úloh].

Pri uzatváraní šprintu prebieha predvedenie vytvorených častí product ownerom (demo) a v prípade splnenia aj ostatných bodov uvedených v metodike Definition of Done [Príloha A-9: Metodika definition of done] je daná story uzavretá. V prípade, že sa nejakú story nepodarí uzavrieť, je prenesená do nasledujúceho šprintu.

2.3 Retrospektíva

Po ukončení šprintu prebiehala retrospektíva, kde rozoberáme pozitívne aj negatívne veci z predchádzajúceho šprintu. Z negatívnych vecí vyberieme tie najdôležitejšie, ktoré sa v ďalšom šprinte pokúsime zlepšiť, a pokúsime sa nájsť riešenie ako to dosiahnuť (vytvorenie metodiky, naplánovanie pravidelných standupov...). Taktiež sa spätne vrátime k negatívam z predošlého obdobia a vyhodnotíme, či podniknuté akcie boli dostatočné na elimináciu vybraných nedostatkov. Z tohto procesu vznikne aj zápisnica, ktorej vzor sa nachádza v metodike tvorby zápisu zo stretnutia a retrospektívy. [Príloha A-8: Metodika zápisov a retrospektív]

2.4 Verziovanie

Na správu verzii kódu využívame nástroj Git, pričom ako vzdialené úložisko sme využili:

- GitLab poskytnutý firmou Kistler pre zdrojové kódy vytváraného riešenia. Keďže išlo o komerčne zakúpenú licenciu, bolo možné toto riešenie využívať bez obmedzení počtu prispievateľov. Toto riešenie malo nevýhodu, že na pripojenie nutné využívať VPN, avšak to bolo nutné aj na prístup k samotnému zariadeniu, na ktorom sa kód spúšťal a tak nám vyšlo toto riešenie ako najlepšie.
- GitHub pre zdrojové kódy webovej stránky. Keďže na webovej stránke nepracuje celý tím, bolo možné využiť GitHub. Tým sa odstránila nutnosť využitia VPN a bolo tak možné využiť kontinuálnu integráciu, či notifikácie v aplikácii Slack.

Git využívame nasledovne:

Pri dopĺňaní funkcionality si člen vykonávajúci implementáciu vytvorí vlastnú vetvu (branch), v ktorej pracuje. Po vytvorení príslušnej funkcionality vytvorí pull request (zároveň aktualizuje stav úlohy v nástroji Jira). Následne je vykonaná prehliadka kódu, prípadné nedostatky sa opravia a

kód je mergenutý do hlavnej vetvy. V súčasnosti zvažujeme použitie vývojárskej vetvy, pričom kód by bol mergeovaný do nej a až po schválení product ownerom by bol presunutý do hlavnej vetvy.

Na vykonanie prehliadky kódu vznikla potreba vytvorenia metodiky. Táto metodika opisuje presný postup pri jej vykonaní. [Príloha A-6: Metodika spravovania zdrojového kódu]

2.5 Dokumentácia

Počas riešenia problémov vytvárame rôznu dokumentáciu či iné podporné dokumenty. Na ich zdieľanie využívame Google Drive. Dokumenty, ktoré sem nahrávame zahŕňajú:

- metodiky
- všeobecné dokumentácie
- návody
- exporty stavu úloh
- retrospektívy
- zápisnice zo stretnutí

Nachádzajú sa tu aj upraviteľné verzie (t.j. nie len PDF, ale aj napríklad DOCX), pri súboroch kde to je možné, čo nám umožňuje ich v budúcnosti jednoducho upraviť.

Pre dokumenty, pri ktorých vznikla potreba aj formálnej úpravy – v súčasnej dobe dokumentácia zápisníc a retrospektív, vznikla metodika určujúca ich štruktúru [Príloha A-8: Metodika zápisov a retrospektív].

Vzhľadom na použitie viacerých podporných nástrojov umožňujúcich ukladanie súborov (dokumentov, zdrojových kódov, či iných informácii) bolo potrebné jasne definovať, kde sa jednotlivé súbory majú nachádzať. Z tohto dôvodu vznikla metodika ukladania súborov [Príloha A-7: Metodika ukladania dokumentov], ktorá tento problém rieši.

3. Sumarizácie šprintov

Nasledujúca časť opisuje jednotlivé šprinty, ktoré boli ukončené. V prvom kontrolnom bode sa jedná o tri šprinty:

Šprint 1 (8.10.2019 - 22.10.2019)

Úvodný šprint bol venovaný základnej analýze problematiky a dát, ktoré bolo možné získať (VDR – Vehicle Data Record, FPT record – Force Presence Time record). Zároveň bolo súčasťou tohto šprintu stanovenie základných princípov fungovania tímu – výber a zavedenie nástrojov, ktoré využívame a napísanie potrebných prvotných metodík. Taktiež bolo potrebné vytvoriť prezentačnú webovú stránku tímu.

V tomto šprinte sa nám podarilo vytvoriť princípy fungovania tímu, webové sídlo a prototyp nástroja na získavanie VDR. Analýzy sa nám podarilo dokončiť len čiastkovo. V tomto šprinte sa nám tak nepodarilo uzavrieť všetky user story.

Šprint 2 (22.10.2019 – 5.11.2019)

V druhom šprinte sme sa venovali dokončeniu rozpracovaných analýz, vytvoreniu nástroja na zbieranie VDR, analýze či dostupné dáta sú vhodné na natrénovanie neurónovej siete a doplneniu dát o dáta, ktoré boli potrebné na natrénovanie neurónovej siete (rozšírenie funkcionality existujúceho systému).

V tomto šprinte sa nám podarilo dokončiť analýzy z predchádzajúceho šprintu a vytvoriť nástroj na zbieranie VDR. Analýza vhodnosti sa ukázala záberom širšia ako bolo očakávané a z dôvodu potreby externých konzultácii jej časť ostala rozpracovaná. Rozšírenie funkcionality existujúceho systému sa nám v tomto šprinte nepodarilo dosiahnuť – nepodarilo sa nám pripraviť prostredie a príslušné príbehy sme presunuli do nasledujúceho šprintu.

Šprint 3 (5.11.2019 - 19.11.2019)

V treťom šprinte sme sa venovali dokončovaniu príbehov z predchádzajúceho šprintu, rozšíreniu funkcionality existujúceho systému a vykonanie analýzy použitia neurónovej siete. Zároveň sme riešili manažérske úlohy, súvisiace s fungovaním tímu a písaním dokumentácie. V tomto šprinte bolo hlavným cieľom úspešne uzavrieť šprint, čo sa nám v prvých dvoch prípadoch nepodarilo. Podarilo sa nám úspešne dokončiť všetky vývojárske úlohy, čo bolo našou prioritou. Úlohy týkajúce sa odovzdania dokumentácie po 3. šprinte ostali na dokončenie, keďže bolo potrebné doplniť údaje aj po uzavretí šprintu.

Šprint 4 (19.11.2019 - 3.12.2019)

Vo štvrtom šprinte sme sa zamerali na zozbieranie a prípravu prvotných vstupných dát pre neurónovú sieť. Prebehla analýza nazbieraných dát a identifikovanie kľúčových hodnôt, dáta sme pre lepšiu prehľadnosť vizualizovali. Po vyfiltrovaní chybných dáta, ktoré obsahovali záporné hodnoty, prázdne polia a podobne sme začali s vytvárať prvý prototyp neurónovej siete.

Zároveň sme analyzovali súčasný systém zariadenia, v ktorých moduloch prebiehajú konkrétne operácie, zbieranie, transformovanie dát, výpočet. A analyzovali sme vhodné miesto na umiestnenie neurónovej siete do systému ako náhradu súčasného riešenia výpočtu.

Šprint 5 (3.12.2019 - 10.12.2019)

Tento šprint bol kratší a trval týždeň, kvôli koncu semestra. V tomto šprinte sme pracovali na zlepšení presnosti prototypu neurónovej siete. Ďašou úlohou bolo aktulizácia tímového repozitáru na základe originálneho firemného repozitáru a vyriešiť prípadné chyby a kolízie s SDK. Zvyšok šprintu sme sa venovali finalizácii dokumentácie po piatich šprintoch a ukončenie zimného semestra.

Šprint 6 (25.2.2020 - 10.3.2020)

V šiestom šprinte sme pripravovali jednotlivé moduly na odosielanie a prijímanie dát zo zariadenia na spracovanie neurónovou sieťou. Tvorba prototypu rekurentnej neurónovej siete. Implementovanie dodatočnej funkcionality vo webovej aplikácii zariadenia, konkrétne drag and drop funkcionalita pri nastavení senzorov. Analyzovali sme takisto rozšírenie webovej aplikácii o vizualizovanie piktogramov pri jednotlivých druhoch áut.

Šprint 7 (10.3.2020 - 24.3.2020)

V siedmom šprinte sme integrovali neurónovú sieť a moduly na server. Analyzovali možné vylepšenia a počítanie ďalších parametrov. Vo webovej aplikácii sme implementovali možnosť meniť pozíciu senzorov v rámci rôznych tratí.

Šprint 8 (24.3.2020 - 7.4.2020)

Ôsmy šprint sme sa venovali vylepšeniu datasetu a klasifikácie. Integrovali sme riešenie na zariadení a porovnávali výsledky v meraniach. Vo webovej aplikácii sme vylepšili výkon a zaviedli stránkovanie záznamov zo zariadenia.

Šprint 9 (7.4.2020 - 21.4.2020)

V deviatom šprinte sme sa zaoberali k príprave odovzdania finálneho produktu. Finalizovali sme integráciu jednotlivých častí riešenia. Implementovali sme vo webovej aplikácii zobrazenie piktogramov pri jednotlivých druhoch vozidiel. Zjednotili sme GIT repozitáre, aktualizovali metodiky, vytvorili draft finálnej dokumentácie.

Šprint 10 (21.4.2020 - 4.5.2020)

V desiatom šprinte sme vytvorený produkt odovzdali a nasadili do prevádzky. Zároveň sme pracovali aj na projektovej dokumentácii.

4. Globálna retrospektíva – zimný semester

V prvých piatich šprintoch sa nám podarilo stanoviť procesy potrebné na úspešné fungovanie v tíme. Z pohľadu práce na riešení sa nám podarilo vykonať analýzu architektúry existujúceho projektu, analyzovať dáta potrebné na vykonanie úlohy a pripraviť ich zbieranie, navrhnúť základnú štruktúru riešenia a rozšíriť dáta poskytované existujúcim systémom o dáta potrebné na vyriešenie úlohy. Výsledkom po prvých piatich šprintoch je prvotný prototyp neurónovej siete, ktorý sme sa snažili vylepšiť a priblížiť výsledky k originálnym výsledkom.

V globálnej retrospektíve sme identifikovali nasledovné pozitíva:

- celkom dobre fungujeme
- máme definované potrebné procesy
- vieme sa dohodnúť
- napredujeme v komunikácii problémov a ich riešení
- dobrá komunikácia v rámci závislých úloh
- vieme splniť zadané úlohy
- prvotný prototyp

Identifikovali sme aj negatíva, ktoré boli počas prvého obdobia najvýraznejšie a na ktorých sme sa snažili pracovať:

- efektivita plánovania
- celková komunikácia tímu niekedy zlyháva
- procesy fungovania ešte nie sú uhladené - objavujú sa nové problémy
- nezaznamenávame všetky detaily z riešenia úloh do Jiry
- dokumentácia nie je ucelená na jednom mieste
- občasná nízka motivácia, angažovanosť alebo tímová zodpovednosť

5. Globálna retrospektíva – letný semester

Počas ďaľších piatich šprintov sme rozšírili moduly na prijímanie dát zo zariadenia za účelom následného spracovania neurónovou sieťou. Vytvorili sme model rekurentnej neurónovej siete, ktorý sme spolu s modulmi prijímania a spracovania dát integrovali na server. Analyzovali sme možné vylepšenia a počítanie ďalších parametrov, ktoré by spresnili náš model, ako aj zväčšili a rozšírili náš pôvodný dataset. Naše výsledky sme porovnávali s pôvodným algoritmickým riešením. Ruku k dielu sme priložili aj na frontende implementáciou drag and drop funkcionality pri nastavení senzorov, takisto sme pridali aj zobrazovanie piktogramov pre ľahšiu vizuálnu klasifikáciu vozidiel a stránkovanie záznamov. Celkovo sa nám ľahko kooperovalo a tvorili sme plnohodnotný tím. Vo výsledku sme odovzdali stanovený produkt a nasadili ho do prevádzky.

Identifikovali sme nasledovné pozitíva:

- zlepšenie komunikácie oproti zimnému semestru
- vytvorenie uceleného produktu – splnili sme zadanie
- lepšie rozdelenie častí projektu – každý si našiel oblasť, v ktorej sa mohol realizovať
- zlepšenie v Jire – vždy stanovené akceptačné kritéria
- lepšia kooperácia
- vyššia motivácia oproti zimnému semestru
- ustálené procesy
- autonómnejšie fungovanie
- väčší progres vo vývoji produktu oproti zimnému semestru

Identifikovali sme nasledovné negatíva, na ktorých sme sa snažili pracovať:

- stále priestor na zlepšenie komunikácie

- neskorý začiatok práce na úlohách – často sme si ich nechávali na poslednú chvíľu
- ohýbanie metodík
- flegmatickejší prístup
- menej diskutovania v dôsledku remote meetingov
- niekedy slabšie demá – chýbala dôslednejšia prezentácia aj menších častí

Príloha A-1: Exporty

Šprint 1 (odkaz)

Issue Type	Summary	Assignee	Status	Created
Story	Prepare web page	*Unassigned*	Closed	8.10.2019
Sub-task	Provide WIM graphics and texts	Bacik Zdenko	Closed	8.10.2019
Sub-task	Deploy	*Unassigned*	Closed	8.10.2019
Sub-task	Research and collect requirements	*Unassigned*	Closed	8.10.2019
Sub-task	Fill with content	*Unassigned*	Closed	15.10.2019
Sub-task	Create team members sections	Snincak Marek	Closed	15.10.2019
Sub-task	Create timeline section	Snincak Marek	Closed	15.10.2019
Sub-task	Pick template for webpage	Valastin Viktor	Closed	15.10.2019
Sub-task	Create documents section	Scipa Christian	Closed	15.10.2019
Sub-task	Implement Continuous Integration of webpage	Lazovy Stefan	Closed	15.10.2019
Sub-task	give access to pista	Tonhauzer Martin	Closed	16.10.2019
Story	GitHub	Valastin Viktor	Closed	8.10.2019
Sub-task	GitHub Slack notifications	Lazovy Stefan	Closed	20.10.2019
Story	Analyze FPT record	*Unassigned*	Closed	27.9.2019
Sub-task	Provide documentation	Bacik Zdenko	Closed	8.10.2019
Sub-task	Create "how to pull fpt records from device"	Chreno Martin	Closed	15.10.2019
Sub-task	Analyze meaning of fpt record fields	Chreno Martin	Closed	16.10.2019
Story	Analyze VehicleData Record	*Unassigned*	Closed	27.9.2019
Sub-task	Provide documentation	Bacik Zdenko	Closed	8.10.2019
Sub-task	Explore listening for data on websocket	Snincak Marek	Closed	15.10.2019
Sub-task	Analyze meaning of VDR record fields	Benova Lenka	Closed	16.10.2019
Sub-task	Document VDR retrieval	Benova Lenka	Closed	20.10.2019
Story	Grant access to necessary tools	Bacik Zdenko	Closed	27.9.2019
Story	Prepare WIM device access	Bacik Zdenko	Closed	7.10.2019
Story	Establish comunication tools	Tonhauzer Martin	Closed	27.9.2019

Story	Generic tasks container	*Unassigned*	Closed	8.10.2019
Sub-task	Notes from second meetup	Scipa Christian	Closed	8.10.2019
Sub-task	Submit TP cup registration	Valastin Viktor	Closed	15.10.2019
Sub-task	Create template for meetup documentation	Scipa Christian	Closed	16.10.2019
Sub-task	Find a way to export jira board	Scipa Christian	Closed	16.10.2019
Sub-task	Notes from third meetup	Scipa Christian	Closed	16.10.2019
Sub-task	Create methodology for document management	Valastin Viktor	Closed	16.10.2019
Sub-task	request more space at VPS	Tonhauzer Martin	Closed	22.10.2019
Sub-task	create group at gitlab	Bacik Zdenko	Closed	22.10.2019

Šprint 2 (odkaz)

Issue Type	Summary	Assignee	Status	Created
Story	Analyze FPT record	*Unassigned*	Closed	27.9.2019
Sub-task	Provide documentation	Bacik Zdenko	Closed	8.10.2019
Sub-task	Create "how to pull fpt records from device"	Chreno Martin	Closed	15.10.2019
Sub-task	Analyze meaning of fpt record fields	Chreno Martin	Closed	16.10.2019
Story	Analyze VehicleData Record	*Unassigned*	Closed	27.9.2019
Sub-task	Provide documentation	Bacik Zdenko	Closed	8.10.2019
Sub-task	Explore listening for data on websocket	Snincak Marek	Closed	15.10.2019
Sub-task	Analyze meaning of VDR record fields	Benova Lenka	Closed	16.10.2019
Sub-task	Document VDR retrieval	Benova Lenka	Closed	20.10.2019
Story	Extend VDR development record to have ID in force record section	*Unassigned*	Closed	18.10.2019
Sub-task	Implement	Benova Lenka	Closed	29.10.2019
Story	Prepare enviroment for development	*Unassigned*	Closed	22.10.2019
Story	Implement VDR collecting	Snincak Marek	Closed	27.9.2019
Sub-task	Implement	Snincak Marek	Closed	29.10.2019
Story	Analyze if current FPT record is suitable for neural netowrk	*Unassigned*	To Do	24.10.2019

Sub-task	Analyze	Chreno Martin	In Progress	29.10.2019
Story	Generic tasks container	*Unassigned*	Closed	8.10.2019
Sub-task	Notes from second meetup	Scipa Christian	Closed	8.10.2019
Sub-task	Submit TP cup registration	Valastin Viktor	Closed	15.10.2019
Sub-task	Create template for meetup documentation	Scipa Christian	Closed	16.10.2019
Sub-task	Find a way to export jira board	Scipa Christian	Closed	16.10.2019
Sub-task	Notes from third meetup	Scipa Christian	Closed	16.10.2019
Sub-task	Create methodology for document management	Valastin Viktor	Closed	16.10.2019
Sub-task	request more space at VPS	Tonhauzer Martin	Closed	22.10.2019
Sub-task	create group at gitlab	Bacik Zdenko	Closed	22.10.2019
Sub-task	Create methodology for tasks export	Scipa Christian	Closed	29.10.2019
Sub-task	Create methodology for meeting notes and sprint retrospective	Scipa Christian	Closed	29.10.2019
Sub-task	Create methodology for communication	Scipa Christian	Closed	30.10.2019
Story	Follow TP Webpage requirements	Valastin Viktor	Closed	22.10.2019
Story	Implement FTP record stream	*Unassigned*	Closed	18.10.2019
Sub-task	Implement	Lazovy Stefan	Closed	29.10.2019

Šprint 3 (odkaz)

Issue Type	Summary	Assignee	Status	Created
Story	Documentation Submission	*Unassigned*	To Do	5.11.2019
Sub-task	Create methodology for code review	Valastin Viktor	Closed	6.11.2019
Sub-task	Update Webpage with second retrospective and third sprint description	Valastin Viktor	Closed	6.11.2019
Sub-task	Create Management Documentation	Snincak Marek	Closed	6.11.2019
Sub-task	Finalize formatting of final document	Scipa Christian	In Progress	6.11.2019
Sub-task	Create global "definition of done" methodology	Snincak Marek	Closed	10.11.2019
Sub-task	Publish "document storage methodology" on projects webpage	Snincak Marek	Closed	10.11.2019
Sub-task	Create "document storage" methodology	Tonhauzer Martin	Closed	12.11.2019
Sub-task	Add steps taken to each retrospective	Scipa Christian	Closed	12.11.2019
Sub-task	Add product owners to the webpage	Valastin Viktor	Closed	12.11.2019
Sub-task	Create Engineering Project Documentation	Tonhauzer Martin	In Progress	12.11.2019
Story	Extend VDR development record to have ID in force record section	*Unassigned*	Closed	18.10.2019
Sub-task	Implement	Benova Lenka	Closed	29.10.2019
Story	Prepare enviroment for development	*Unassigned*	Closed	22.10.2019
Sub-task	Document	Lazovy Stefan	Closed	6.11.2019
Story	Analyze if current FPT record is suitable for neural netowrk	*Unassigned*	Closed	24.10.2019
Sub-task	Analyze	Chreno Martin	Closed	29.10.2019
Story	Implement FTP record stream	*Unassigned*	Closed	18.10.2019
Sub-task	Implement	Lazovy Stefan	Closed	29.10.2019

Šprint 4 (odkaz)

Issue Type	Summary	Assignee	Status	Created
Story	Documentation Submission	*Unassigned*	Closed	5.11.2019
Sub-task	Create methodology for code review	Valastin Viktor	Closed	6.11.2019
Sub-task	Update Webpage with second retrospective and third sprint description	Valastin Viktor	Closed	6.11.2019
Sub-task	Create Management Documentation	Snincak Marek	Closed	6.11.2019
Sub-task	Finalize formatting of final document	Scipa Christian	Closed	6.11.2019
Sub-task	Create global "definition of done" methodology	Snincak Marek	Closed	10.11.2019
Sub-task	Publish "document storage methodology" on projects webpage	Snincak Marek	Closed	10.11.2019
Sub-task	Create "document storage" methodology	Tonhauzer Martin	Closed	12.11.2019
Sub-task	Add steps taken to each retrospective	Scipa Christian	Closed	12.11.2019
Sub-task	Add product owners to the webpage	Valastin Viktor	Closed	12.11.2019
Sub-task	Create Engineering Project Documentation	Tonhauzer Martin	Closed	12.11.2019
Sub-task	Create NDA compliant version of documentation and publish it on webpage	Snincak Marek	Closed	21.11.2019
Story	Generic task container	*Unassigned*	Closed	24.11.2019
Sub-task	Analyze if OpenNN is suitable for our usecase	Snincak Marek	Closed	24.11.2019
Sub-task	Update webpage	Scipa Christian	Closed	28.11.2019
Story	Create NN MLP prototype	*Unassigned*	Closed	19.11.2019
Sub-task	Process input created in task WIMNN-84	Tonhauzer Martin	Closed	20.11.2019
Sub-task	Create model	Chreno Martin	Closed	20.11.2019
Story	Collect test VDR data from Rumlang	*Unassigned*	Closed	15.11.2019
Sub-task	Collect data	Valastin Viktor	Closed	20.11.2019
Story	Prepare dataset for MLP RNN	*Unassigned*	Closed	15.11.2019
Sub-task	Create dataset suitable as input for NN	Benova Lenka	Closed	20.11.2019
Sub-task	Analyse dataset and create visualisations	Lazovy Stefan	Closed	20.11.2019
Story	Find where the NN should be placed	*Unassigned*	Closed	7.10.2019
Sub-task	Find where NN should be placed	Scipa Christian	Closed	27.11.2019

Šprint 5 (odkaz)

Issue Type	Summary	Assignee	Status	Created
Story	Adapt source to new image	*Unassigned*	Closed	25.11.2019
Sub-task	Adapt	Snincak Marek	Closed	3.12.2019
Story	Build Yocto locally	*Unassigned*	To Do	3.12.2019
Sub-task	Build Yocto	Lazovy Stefan	To Do	3.12.2019
Story	Adjust NN prototype to achieve better accuracy	*Unassigned*	Closed	3.12.2019
Sub-task	Improve NN accuracy	Chreno Martin	Closed	3.12.2019
Story	Filter dataset	*Unassigned*	Closed	3.12.2019
Sub-task	Remove faulty values	Benova Lenka	Closed	3.12.2019
Story	Documentation submission	*Unassigned*	To Do	3.12.2019
Sub-task	Update current documents	Scipa Christian	In Progress	3.12.2019
Sub-task	Create NDA version of documentation	Scipa Christian	In Progress	3.12.2019
Sub-task	Add NN prototype to documentation	Chreno Martin	Closed	10.12.2019
Sub-task	Add VDR parser to documentation	Benova Lenka	Closed	10.12.2019
Story	Prepare enviroment for UI development	*Unassigned*	To Do	3.12.2019
Sub-task	Prepare enviroment	Snincak Marek	In Progress	3.12.2019

Šprint 6 (odkaz)

Issue Type	Summary	Assignee	Status	Created
Story	Prepare dataset for Recurrent NN		Closed	27.9.2019
Sub-task	Prepare dataset	G0381	Closed	25.2.2020
Story	Create Recurent NN prototype		Closed	21.11.2019
Sub-task	Create prototype	G0382	Closed	25.2.2020
Story	Stream FPT data from device		To Do	19.11.2019
Sub-task	Implement	G0376	In Review	25.2.2020
Story	React: Drag'n'drop functionality on site setup		Closed	25.11.2019
Sub-task	Implement (cooperative task)	G0377	Closed	25.2.2020
Sub-task	Implement (cooperative task)	G0378	Closed	25.2.2020
Story	Process FPT data on a server		Closed	25.2.2020
Sub-task	Implement	G0380	Closed	25.2.2020
Story	React: Analyze possibility of dynamic pictogram generation	G0379	Closed	25.2.2020
Sub-task	Analyze	G0379	Closed	25.2.2020

Šprint 7 (odkaz)

Issue Type	Summary	Assignee	Status	Created
Story	Extend output of VS module with vehicle id		Closed	25.2.2020
Sub-task	Implement	G0376	Closed	10.3.2020
Story	Integrate Neural network on the server		Closed	21.11.2019
Sub-task	Integrate NN on server (cooperative task)	G0381	Closed	10.3.2020
Sub-task	Integrate NN on server (cooperative task)	G0380	Closed	10.3.2020
Story	React: Pictogram for vehicle classes		To Do	25.11.2019
Sub-task	Implement	G0379	In Progress	10.3.2020
Story	Analyze NN possibilities to evalutate different parameters		Closed	10.1.2020
Sub-task	Analyze	G0382	Closed	10.3.2020
Story	React: Drag'n'drop functionality on-site setup - force sensor		Closed	10.3.2020
Sub-task	Implement	G0377	Closed	10.3.2020
Story	React: React: Drag'n'drop functionality between lanes - presence sensor		Closed	10.3.2020
Sub-task	Implement	G0378	Closed	10.3.2020

Šprint 8 (odkaz)

Issue Type	Summary	Assignee	Status	Created
Story	React: Pictogram for vehicle classes		To Do	25.11.2019
Sub-task	Implement	G0379	In Progress	10.3.2020
Story	Store vehicle data on a Server		Closed	17.2.2020
Sub-task	Implement	G0381	Closed	24.3.2020
Sub-task	Integrate categorical model into service	G0381	Closed	6.4.2020
Story	React: Analyse possibility to implement paging on currently broken history page		Closed	24.3.2020
Sub-task	Debug history page	G0377	Closed	24.3.2020
Sub-task	Analyse pagination implementation difficulty	G0377	Closed	24.3.2020
Story	Integrate solution on real device and compare results		To Do	7.10.2019
Sub-task	Compare solutions	G0376	In Progress	24.3.2020
Story	Parse new collected vehicles dataset suitable for classification via NN		Closed	24.3.2020
Sub-task	Prepare new dataset	G0380	Closed	24.3.2020
Story	Improve vehicle classification		Closed	24.3.2020
Sub-task	Improve results	G0382	Closed	24.3.2020
Story	React: improve drag&drop performance		Closed	24.3.2020
Sub-task	Improve performance	G0378	Closed	24.3.2020
Story	Add page possibilities in History Page	G0377	Closed	23.3.2020
Sub-task	Implement	G0377	Closed	31.3.2020

Šprint 9 (odkaz)

Issue Type	Summary	Assignee	Status	Created
Story	React: Pictogram for vehicle classes		Closed	25.11.2019
Sub-task	Implement	G0379	Closed	10.3.2020
Story	Integrate solution on real device and compare results		To Do	7.10.2019
Sub-task	Compare solutions	G0376	In Progress	24.3.2020
Story	Update methodologies		Closed	7.4.2020
Sub-task	Update methodologies	G0377	Closed	7.4.2020
Story	Finalize NN and FTP processing integration		Closed	7.4.2020
Sub-task	connect ftp-processing to nn repository	G0381	Closed	20.4.2020
Sub-task	Prepare NN models for integration with server	G0382	Closed	20.4.2020
Story	Clean GIT repo		Closed	7.4.2020
Sub-task	move vdr collector	G0378	Closed	19.4.2020
Sub-task	polish NN related repositories	G0380	Closed	19.4.2020
Sub-task	clean backend repo	G0376	Closed	19.4.2020
Story	Create final document draft		Closed	7.4.2020
Sub-task	Create draft	G0378	Closed	7.4.2020

Šprint 10 (odkaz)

Issue Type	Summary	Assignee	Status	Created
Story	Integrate solution on real device and compare results		Closed	7.10.2019
Sub-task	Compare solutions	G0376	Closed	24.3.2020
Story	Integrate frontend on real device		Closed	21.4.2020
Sub-task	integrate	G0379	Closed	21.4.2020
Story	Create final documentation		To Do	21.4.2020
Sub-task	write part of the documentation	G0380	Closed	27.4.2020
Sub-task	write part of the documentation	G0377	Closed	27.4.2020
Sub-task	write part of the documentation	G0379	Closed	27.4.2020
Sub-task	write part of the documentation	G0378	Closed	27.4.2020
Sub-task	create presentation for final product demo	G0381	Closed	28.4.2020
Sub-task	create presentation for final product demo	G0378	Closed	28.4.2020
Sub-task	write part of the documentation	G0382	Closed	28.4.2020
Sub-task	write part of the documentation	G0381	In Progress	28.4.2020
Sub-task	write part of the documentation	G0376	In Progress	28.4.2020

Prílohy A-2: Motivačný dokument

tim08-2019-fiit@googlegroups.com

Tímový projekt: Motivačný list

Tím:

Lenka Beňová
Martin Chreno
Štefan Lazový
Marek Sninčák
Christian Ščípa
Martin Tonhauzer
Viktor Valaštín

O nás:

Náš tím sa skladá z absolventov STU FIIT zo všetkých odborov. Aj mimo školy sa venujeme rôznym oblastiam IT. Viacerí sa venujú backendu a low-level programovaniu. Taktiež sa časť nášho tímu venuje vývoju frontendu. Všetci popri škole pracujeme v oblasti IT. Niektorí naši členovia sa zúčastnili viacerých súťaží - napr. Guardians v spolupráci s NBU CERT a Binary confidence. Niekoľko našich bakalárskych prác bolo ocenených vyznamenaním dekanky. Jeden z našich členov dokonca na základe svojej bakalárskej práce publikoval vedecký článok, ktorý odprezentoval na medzinárodnom sympóziu Elmar.

Dynamické váženie vozidiel počas premávky

Táto téma nás zaujala kvôli jej orientácií na low-level programovanie a využitie technológií ako C++ a neurónové siete, ktoré si prevažná väčšina nášho tímu dobrovoľne zvolila ako voliteľný predmet zimného semestra.

Traja z našich členov napísali svoju bakalársku prácu v jazyku C++. Jeden z členov sa venoval v rámci práce sieti unikernelov pre prostredie IoT a cloudu. V ďalších našich prácach sa členovia nášho tímu venovali low-level čítaniu dát z disku a operačnej pamäte v prostredí Linuxu. Dvaja členovia nášho tímu pracovali v rámci svojich prác s Pythonom.

Jeden z našich členov sa profesionálne venuje vývoju aplikácií pre Linux v C++. Dvaja z našich členov sa profesionálne venujú správe databáz. Náš tím má skúsenosti s MySQL, PostgreSQL, Redis a MongoDB. Jeden z našich členov absolvoval predmet Mikropočítače.

Mimo školy a práce sa taktiež zaujímame o inovatívne technológie v automobilovom priemysle ako aj v oblasti vývoja hardvéru

Multibanková mobilná aplikácia pre manažment financií

Myslíme si, že v tejto téme si nájde každý člen niečo svoje. Náš tím pozostáva z ľudí so širokou oblasťou schopností získaných nielen na predmetoch bakalárskeho štúdia. Prevažná väčšina členov tímu má praktické skúsenosti s JavaScriptom vo frameworkoch ako VueJS a Express, ako aj s vývojom mobilných aplikácií v React Native a Jave v prostredí Android Studio. Náš tím pozostáva aj z členov so skúsenosťami v dátovej analýze.

Oblasť financií považujeme za atraktívne a dynamické odvetvie. Vidíme reálny prínos projektu pre koncových používateľov. Existuje mnoho ľudí, ktorí majú účet vo viac ako jednej banke. Vtedy môže byť výzvou mať komplexný prehľad nad finančnými pohybmi na všetkých účtoch. Taktiež vidíme možný potenciál v rozšírení sledovania výdavkov aj o pohyby hotovostných transakcií využitím overovania QR kódov Virtuálnej Registračnej Pokladne.

Pomoc zvieratám v núdzi

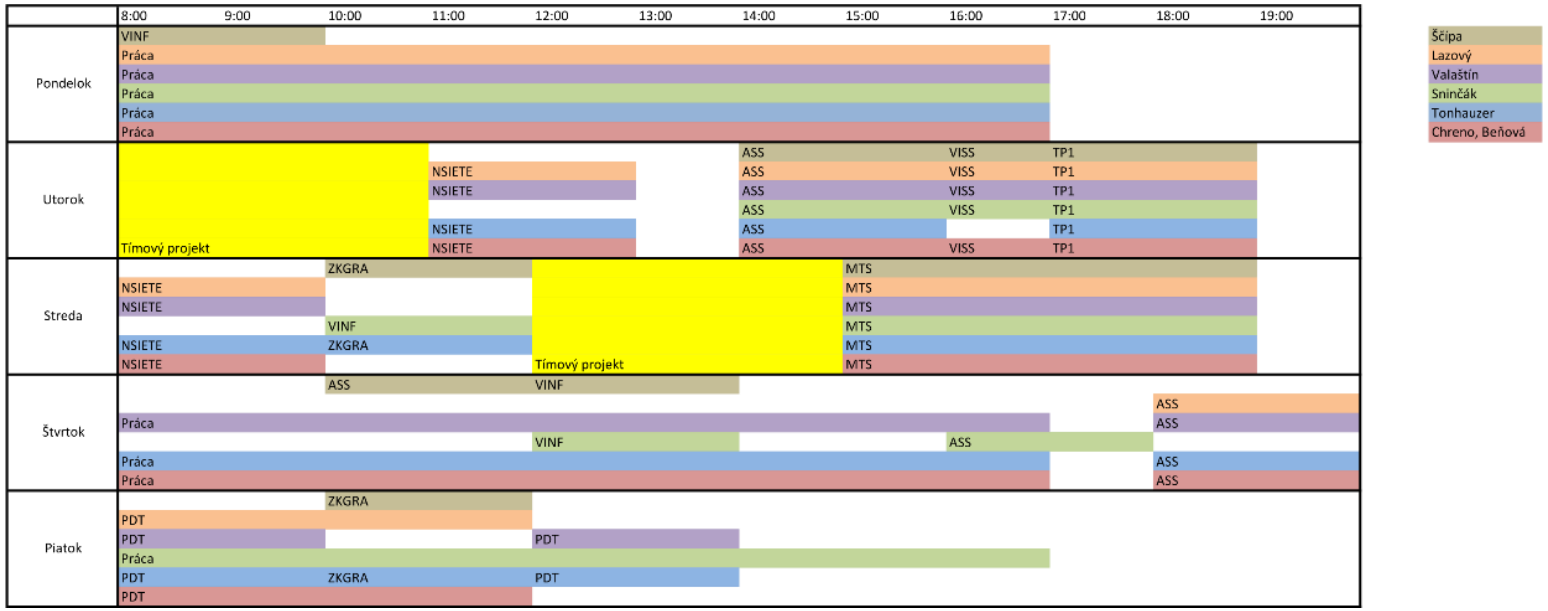
Téma nás zaujala tým, že je jej riešením systém, ktorý bude používaný nielen internými zamestnancami, ale aj širokou verejnosťou a bude voľne dostupný pre kohokoľvek. Tento systém nielen že pomôže dobrej veci, ale je to aj správny krok, akým by sa malo uberať viacero občianskych združení, poprípade štátnych inštitúcií, pri využívaní moderných technológií a styku s verejnosťou. Systém uľahčí a sprehľadní nahlasovanie a sledovanie rôznych prípadov zlého zaobchádzania so zvieratami. V neposlednom prípade môže pomôcť s propagáciou občianskeho združenia, zvýšením záujmu verejnosti a takisto priniesť nových prispievateľov na financovanie združenia.

Okrem samotného riešeného problému nás téma zaujala aj použitými technológiami, ktoré nám umožnia posunúť sa ďalej v oblastiach, o ktoré sa zaujímame aj mimo tohto projektu.

Všetci členovia tímu máme skúsenosti s návrhom zložitejších softvérových systémov z bakalárskeho štúdia na fakulte. Takisto väčšina z nás pracovala na tvorbe webových aplikácií, či už z pohľadu frontendu alebo backendu, vrátane spomenutých povinných technológií v popise témy. Pravidelne pracujeme s databázami či už v škole alebo v praxi. Máme zapísané aj odporúčané predmety ako Objektovo orientovaná analýza a návrh softvéru alebo Pokročilé databázové technológie.

Zoradenie všetkých tém podľa priority:

18. Dynamické váženie vozidiel počas premávky [WIM]
9. Multibanková mobilná aplikácia pre manažment financií [MultiBank]
5. Pomoc zvieratám v núdzi [AnimalRescue]
4. Vizualizácia softvéru vo virtuálnej a rozšírenej realite 2.0 [VizReal]
16. Virtuálna identita [Virtual ID]
6. Predikcia ceny vozidla na základe inzerátu [CarAd]
8. Vyhľadávanie pomocou obrázkov [ImageSearch]
3. Prostredie na vizualizáciu mikrogridu [GridBox]
11. We AR city - Inteligentná komunikácia občana s mestom [city4us]
14. Bezpečné manažovanie siete v prostredí Internetu vecí [SecIoT]
21. Kolaboratívne virtuálne prostredie [Coven]
1. Event Navigation [EventNav]
22. Smart pivovar [SmartBrew]
10. Databanka otázok a úloh [FIIT-DU]
15. Inteligentný tvorca príbehov z dát [TellStoryAI]
7. Animované architektúry [AnimArch]
2. Prehľadaváníe a vizualizácia textov [TextTool]
12. Inovatívny portál pre boj s antisociálnym správaním s využitím umelej inteligencie [FireAnt]
19. Automatizácia procesu spracovania a klasifikácie textov [TextProcessing]
13. Podpora kvality služieb pre budúci Internet [QoSbySDN]
20. Použiteľnosť mobilných aplikácií [MobeUX]
17. Blockchain platobné brány [BlockPay]


8:00
9:00
10:00
11:00
12:00
13:00
14:00
15:00
16:00
17:00
18:00
19:00
Pondelok
VINF
Práca
Práca
Práca
Práca
Práca
Utorok
Tímový projekt
NSIETE
NSIETE
NSIETE
NSIETE
ASS
ASS
ASS
ASS
ASS
ASS
VISS
VISS
VISS
VISS
VISS
TP1
TP1
TP1
TP1
TP1
Streda
NSIETE
NSIETE
NSIETE
NSIETE
ZKGRA
VINF
ZKGRA
Tímový projekt
MTS
MTS
MTS
MTS
MTS
MTS
Štvrtok
Práca
Práca
Práca
ASS
VINF
VINF
ASS
ASS
ASS
ASS
ASS
Piatok
PDT
PDT
Práca
PDT
ZKGRA
PDT
ZKGRA
PDT
PDT
Ščipa
Lazový
Valaštín
Snínčák
Tonhauzer
Chreno, Beňová

Prílohy A-3: Metodika exporty úloh

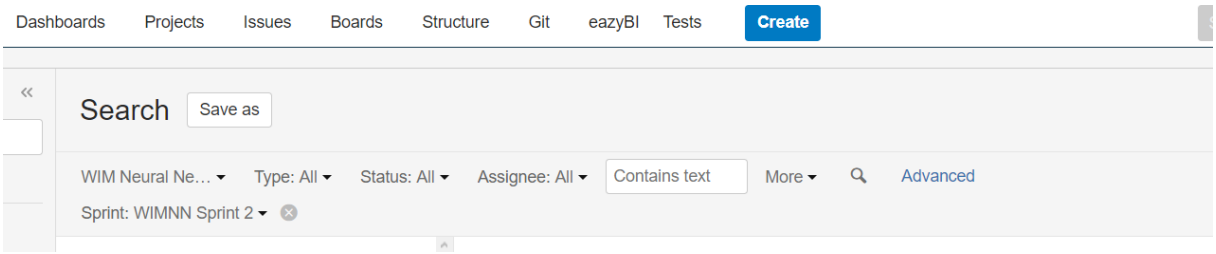
Metodika exportu úloh v JIRA

Tím 08| Téma: WIM

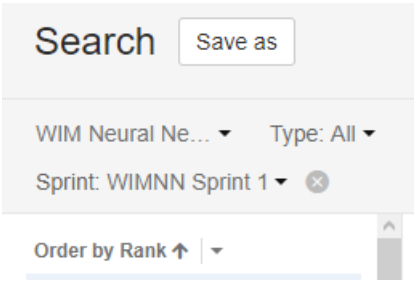
Súčasťou dokumentácie je aj pravidelný export úloh aktuálneho šprintu. Export sa vykonáva po stretnutí tímu, poprípade po vytvorení a zadelení úloh členom tímu.

Postup pri tvorbe exportu po stretnutí:

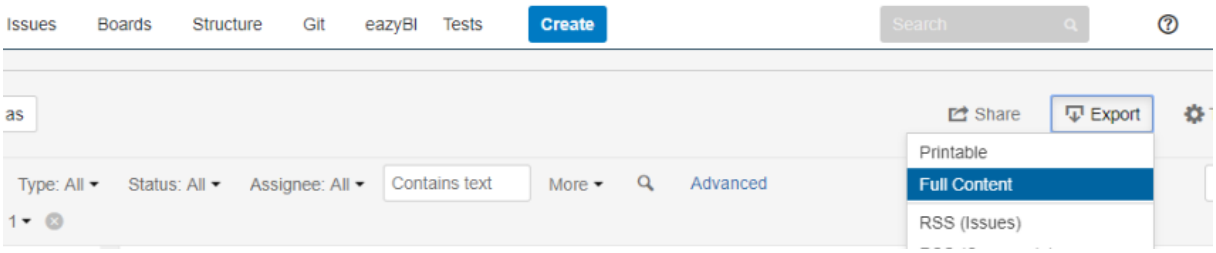
1. V JIRE na karte *Issues* na hlavnom paneli nájdeme vyhľadávanie „*Searchforissues*“
2. Použitím filtru pre aktuálny projekt a šprint vyhľadáme všetky *stories* a *subtasky* pre daný šprint



3. Výsledky si zoradíme podľa Ranku, tým dosiahneme že výsledky budú zoradené podľa stories a ich taskov, ktoré k nim patria.



4. Po vyfiltrovaní výsledkov, kliknutím na tlačidlo Export ->Fullcontent sa nám otvorí stránka s prehľadným zoznamom.

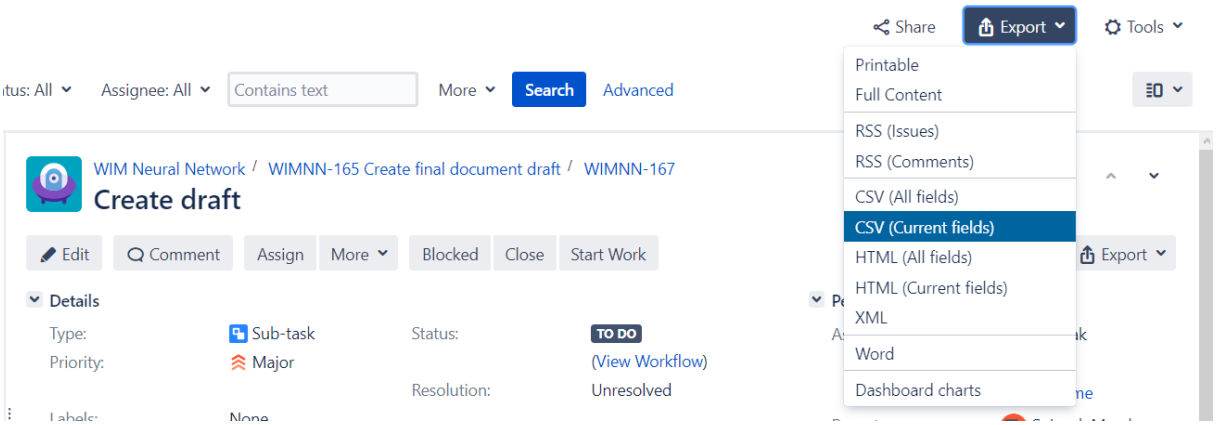


5. Túto stránku si uložíme pomocou prehliadača vo formáte pdf.
6. Ak potrebujeme pdf následne ešte upraviť (zmazať citlivé údaje, linky). Využijeme na to tool na úpravu pdf. (naprhttps://www.sejda.com/pdf-editor)

Postup pri tvorbe export na konci šprintu:

Súčasťou dokumentácie je aj prehľadný zoznam úloh na konci šprintu.

Postup je rovnaký ako pri tvorbe klasického exportu s tým rozdielom že po vyfiltrovaní výsledkov pri kroku 4. využijeme funkciu CSV currentfield.



Vyexportuje sa nám stručná tabuľka s prehľadom všetkých stories a subtaskov daného šprintu.

Prílohy A-4: Metodika komunikácia

Metodika komunikácie

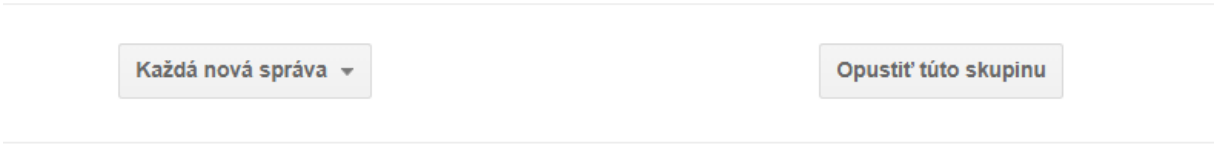
Tím 08 | Téma: WIM

Komunikácia v rámci tímu a zadávateľmi prebieha v nástroji Slack. V nástroji využívame niekoľko rôznych kanálov na posielanie hromadných správ, rozdelených podľa témy/funkcionality. Tieto kanály sú

- **#general** – hlavný kanál určený na komunikáciu tímu so zadávateľmi
- **#links** – kanál kde sa zdieľajú potrebné linky na git repozitáre, zariadenie, dokumenty atď.
- **#notifications** – kanál na automatické notifikácie Github repozitáru pre web stránku
- **#private** – kanál určený na súkromnú komunikáciu členov tímu
- **#random** – kanál v ktorom sa komunikuje všetko čo nie je určené do iných kanálov
- **#standup** - kanál na pravidelné aktualizovanie stavu aktuálnych stories a subtaskov
- **#webpage** - kanál určený na pripomienky a návrhy k zmenám na web stránke tímu

Kontaktovať tím môže ktokoľvek na e-mailovej stránke tim08-2019-fiit@googlegroups.com ,

Je potrebné aby každý člen tímu túto e-mailovú schránku pravidelne kontroloval, poprípade si nastavil preposielanie správ na svoj osobný e-mail.



Príloha A-5: Metodika kontroly kódu

Metodika kontroly zdrojového kódu

Tím 08 | Téma: WIM

Dokument obsahuje postup kontroly zdrojového kódu. Cieľom metodiky je udržať vysokú kvalitu zdrojového kódu.

Riešiteľ úlohy ktorý postupoval podľa **metodiky spravovania zdrojového kódu.** Po dokončení úlohy vytvorí riešiteľ požiadavku na merge s branchoumaster a požiada iného člena tímu o kontrolu kódu. Člen tímu, ktorý bol poverený kontrolou pozorne prejde všetky zmeny v súboroch so zdrojovým kódom a v prípade nezrovnalostí okomentuje s riadok s kódom, ktorý sa mu nezdá v poriadku, s tým že uvedie stručný popis s chybou.

Takto okomentovaný kód môže byť zamergovaný až po vyriešení všetkých problémov.

Príloha A-6: Metodika spravovania zdrojového kódu

Metodika spravovania zdrojového kódu

Tím 08 | Téma: WIM

Informácie potrebné v rámci tímu pre správnu správu so zdrojovým kódom projektu. Každý člen tímu, ktorého úloha je implementácia do zdrojového projektu si pred začatím práce stiahne do svojho pracovného stroja najnovšiu verziu repozitára uloženého na branchimaster (použitím príkazu git pullorigin/master).

Následne si riešiteľ vytvára vlastnú branchu s menom *číslоÚlohy-názov* (napr 62-ftp-id) použitím príkazu *git checkout -b číslоÚlohy-názov*. Checkout je potrebné vykonať z branchemaster.

Člen tímu, ktorý na zadaní pracuje je povinný svoju prácu priebežne commitovať a pushovať aby ostatní členovia tímu mohli v prípade potreby na úlohe pokračovať.

Po dokončení úlohy vytvorí riešiteľ požiadavku na merge s branchoumaster a požiada iného člena tímu o kontrolu kódu. Skontrolovaný a schválený kód môže byť následne zamergovaný do branchemaster.

Príloha A-7: Metodika ukladania dokumentov

Metodika ukladania dokumentov a informácii

Tím 08 | Téma: WIM

Informácie potrebné v rámci tímu a projektu sú uložené na rôznych miestach

- **Kód k webovej stránke** – Uložený v repozitári
- **Kód k ostatným projektom** – Uložené na GitlabeKistleru (potrebná VPN)
- **Project Management** – Prebieha v JireKistleru (potrebná VPN)
- **Dokumenty** – Ostatné dokumenty ako sú metodiky, šprinty, export taskov sú uložené na Google Drive

Dokumenty, ktoré musia byť na uložené na webovej stránke tímového projektu sa ukladajú do static/files následne sa k nim vytvorí link v index.html.

Príloha A-8: Metodika zápisov a retrospektív

Metodika tvorby zápisu zo stretnutia a retrospektívy

Tím 08| Téma: WIM

Z pravidelných stretnutí sa tvorí zápisnica, ideálne počas stretnutia. Zápisnica obsahuje:

- Hlavičku – s hlavnými informáciami dátum, miesto, účasť, meno zapisovateľa
- Obsah stretnutia – krátky opis v jednotlivých bodov, čo všetko sa riešilo na stretnutí
- Ukončené úlohy – zoznam úloh, ktoré boli od posledného stretnutia označené ako ukončené
- Nové identifikované úlohy – zoznam nových úloh alebo aktualizovanie starých úloh, ktoré vznikli alebo sa zmenili od posledného stretnutia
- Iné – voliteľné sekcia s ďalšími poznatkami zo stretnutia (napr. dohodnutie novej metodiky, identifikovanie problému)

Vzor zápisu :

Zápis zo stretnutia č. X

Tím 08 | Téma: WIM

Dátum: xx.xx.2019

Miesto: 1.36 (FIIT)

Prítomní: Všetci

Zapisovateľ: xxxxxxxxxxx

`

Obsah stretnutia:

1. Xxxx
2. xxxxx

Ukončené úlohy:

1. xxxxx
2. xxxxx

Nové identifikované úlohy:

1. xxxxx
2. xxxxxx

Retrospektíva šprintu spočíva vo vypísaní kladov a záporov, ktoré sa v danom šprinte objavili. Členovia tímu by sa mali snažiť dodržiavať kladné vlastnosti šprintu a snažiť sa vyriešiť problémy, ktoré sa vyskytli. V poslednej časti „čo začneme robiť“ sú spísané konkrétne úlohy / akcie na zlepšenie nasledujúcich šprintov.

Vzor retrospektívy:

Šprint č.X - retrospektíva

Tím 08 | Téma: WIM

Dátum: XX.XX.2019 – XX.XX.2019

`

POZITÍVA	ČO TREBA ZLEPŠIŤ
• xxxxxxx • xxxxxxx	• xxxxx • xxxxx

ČO ZAČNEME ROBIŤ
• xxxxxx

Príloha A-9: Metodika definition of done

Metodika Definition of Done

Tím 08 | Téma: WIM

Tento dokument obsahuje zoznam kritérií, ktoré musia byť splnené, aby bolo možné príslušnú úlohu (user story, task) považovať za splnenú.

user story:

- sú splnené všetky jej podúlohy
- sú splnené akceptačné kritéria (ak nie sú explicitne definované, tak vyplývajúce z názvu a popisu)
- je akceptovaná productownerom

implementačná úloha:

- vytvorený kód je možné spustiť
- sú splnené akceptačné kritéria (ak nie sú explicitne definované, tak vyplývajúce z názvu a popisu)
- kód je primerane otestovaný (ak je to vhodné / možné, sú vytvorené automatizované testy)
- ak je to potrebné, je vytvorená primeraná dokumentácia (používateľská príručka ...)
- je vykonaná prehliadka kódu

analytická úloha:

- je vytvorený dokument obsahujúci výsledky analýzy
- sú splnené akceptačné kritéria (ak nie sú explicitne definované, tak vyplývajúce z názvu a popisu)