

UNIVERZITA PARDUBICE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

DIPLOMOVÁ PRÁCE

2021

Michal Struna

Univerzita Pardubice
Fakulta elektrotechniky a informatiky

Detekce a analýza exoplanet s využitím
distribuovaných výpočtů a umělé inteligence

Michal Struna

Diplomová práce

2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Michal Struna**
Osobní číslo: **I16144**
Studijní program: **B2646 Informační technologie**
Studijní obor: **Informační technologie**
Název tématu: **Webový 3D simulátor těles ve vesmíru**
Zadávající katedra: **Katedra informačních technologií**

Z á s a d y p r o v y p r a c o v á n í :

Práce se zabývá tvorbou webové typescriptové aplikace pro 3D vizualizaci těles ve vesmíru. V rámci práce je kladen důraz na dynamický obsah, na kterém se mohou všichni uživatelé po úspěšné autentifikaci podílet. Data jsou ukládána do databáze na serveru. Pro práci s databází je využito REST API. Cílem bakalářské práce je vytvořit webovou aplikaci v jazyce TypeScript, jejímž obsahem je 3D simulátor těles ve vesmíru v reálném čase. Aplikace se skládá z klientské a serverové části. Klientská část zahrnuje: - Uživatelské rozhraní v Reactu a ostylované v SASS umožňující uživatelům autentifikaci, zobrazení a úpravu obsahu a písemnou komunikaci s ostatními uživateli. - 3D simulátor využívající knihovnu THREE.js zobrazující tělesa v reálném čase. Průběh času je možné zrychlovat, zpomalovat či vracet. Serverová část obsahuje: - Serverová aplikace napsaná v Node.js poskytující REST API pro práci s daty. - Data budou uložena v MongoDB databázi, se kterou se bude pracovat za využití knihovny Mongoose. - Dokumentace REST API pomocí nástroje Swagger. Výstupem práce je aplikace, kterou je po nainstalování závislostí pomocí balíčkovacího systému npm a transpilaci TypeScriptu do JavaScriptu možné okamžitě spustit.

Rozsah grafických prací:

Rozsah pracovní zprávy: **min. 30 s., dop. rozsah 40 s.**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

KLECZEK, Josip. Velká encyklopedie vesmíru. Praha: Academia, 2002s., 48s. barev. obr. příl. ISBN 80-200-0906-x

REES, Martin J, ed. Vesmír: [obrazová encyklopedie]. Přeložil Pavel

PŘÍHODA. V Praze: Knižní klub, 2006. ISBN 80-242-1668-x

JPL Solar System Dynamics. JPL Solar System Dynamics [online]. Dostupné z: <https://ssd.jpl.nasa.gov>

MARDAN, Azat. Practical Node.js: building real-world scalable web apps.

Berkeley, California: Apress, [2014]. Expert's voice in Web development. ISBN 978-1-4302-6595-5

MARDAN, Azat. Practical Node.js: building real-world scalable web apps.

Berkeley, California: Apress, [2014]. Expert's voice in Web development. ISBN 978-1-4302-6595-5

Vedoucí bakalářské práce:

Ing. Monika Borkovcová, Ph.D.

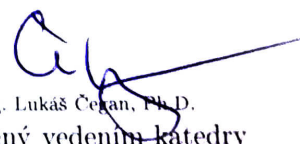
Katedra informačních technologií

Datum zadání bakalářské práce: **31. října 2018**

Termín odevzdání bakalářské práce: **12. května 2019**



Ing. Zdeněk Němec, Ph.D.
děkan



Ing. Lukáš Čechan, Ph.D.
pověřený vedením katedry

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 9. 5. 2021

Michal Struna

Poděkování

...

ANOTACE

...

KLÍČOVÁ SLOVA

exoplanety, extrasolární planety, kepler, umělá inteligence, python

TITLE

Artificial intelligence for exoplanet detection from transit data

ANNOTATION

...

KEYWORDS

exoplanet, extrasolar planets, kepler, artificial intelligence, python

OBSAH

Seznam obrázků	11
Seznam tabulek	13
Seznam zdrojových kódů	14
Seznam vzorců	15
Seznam obrázků	16
Úvod	17
1 Hledání exoplanet	18
1.1 Tranzitní metoda	20
1.1.1 Target pixel file	21
1.1.2 Světelná křivka hvězdy	21
1.1.3 Vyřazení false positives	23
1.1.4 Výpočet vlastností planety	24
1.1.5 Výpočet hmotnosti planety	25
1.2 Metoda radiálních rychlostí	26
1.2.1 Měření vlnové délky záření hvězdy	27
1.2.2 Výpočet radiální rychlosti hvězdy	27
1.2.3 Výpočet hmotnosti planety	29
1.3 Astrometrická metoda	30
1.4 Gravitační mikročočky	31
1.5 Přímé zobrazení	31
1.6 Transit timing	31
1.7 Pulsar timing	31
2 Umělá inteligence	32
2.1 Umělé neuronové sítě	32
2.2 Evoluční algoritmy	32
2.2.1 Genetický algoritmus	32

2.2.2	Algoritmus diferenciální evoluce	32
2.3	Fuzzy systémy	32
2.4	Expertní systémy	32
3	Umělé neuronové sítě	33
3.1	Neuron	33
3.1.1	Vstupy neuronu	34
3.1.2	Váhy spojení	34
3.1.3	Práh neuronu	35
3.1.4	Agregační funkce	35
3.1.5	Aktivační funkce	35
3.2	Typy neuronových sítí	36
3.2.1	Jednoduchý perceptron	36
3.2.2	Dopředná vícevrstvá umělá neuronová síť	37
3.2.3	Konvoluční neuronová síť	38
3.3	Typy vrstev neuronových sítí	40
3.3.1	Plně propojená vrstva	40
3.3.2	Konvoluční vrstva	40
3.3.3	Vrstva Max-Pooling	42
3.3.4	Vrstva Flatten	43
3.3.5	Vrstva Dropout	43
3.4	Učení umělé neuronové sítě	44
3.4.1	Algoritmus zpětného šíření chyby	44
4	Projekty pro hledání exoplanet	46
4.1	Planet Hunters	46
4.2	Astronet	46
5	Použité technologie	47
5.1	TypeScript	47
5.1.1	React	47
5.1.2	Styled Components	47
5.1.3	NPM	47
5.2	Python	47

5.2.1	Flask	47
5.2.2	Astropy	47
5.2.3	LightKurve	47
5.2.4	TensorFlow	47
5.2.5	Pip	47
5.3	MongoDB	47
5.4	Socket.io	47
5.5	Git	47
6	Návrh a vývoj aplikace	48
6.1	Server	48
6.1.1	REST API	49
6.1.2	Socket.io API	50
6.2	Webová aplikace	51
6.3	Klientská aplikace	51
6.4	Neuronová síť	52
6.5	Databáze	52
6.6	Datasety	52
6.6.1	Target pixel files	53
6.6.2	Světelné křivky hvězd	53
6.6.3	Radiální rychlosti hvězd	53
6.6.4	Hvězdy	53
6.6.5	Planety	54
6.6.6	Názvy	54
7	Rozvržení aplikace	56
7.1	Přehled	56
7.2	Databáze	56
7.3	Detail systému	56
7.4	Objevování	56
7.5	Nápověda	56
7.6	Autentizace	56
	Závěr	57

Použitá literatura	57
Seznam příloh	60

SEZNAM OBRÁZKŮ

Obrázek 1	Četnost objevů exoplanet v jednotlivých letech	18
Obrázek 2	Počty objevených exoplanet jednotlivými metodami	19
Obrázek 3	Znamé exoplanety dle jejich vlastností a metody objevení	19
Obrázek 4	Přechod planety přes kotouč hvězdy	20
Obrázek 5	Target pixel file hvězdy Kepler-10	21
Obrázek 6	Světelná křivka soustavy Kepler-13	22
Obrázek 7	Dvojhvězda KIC 8262223	23
Obrázek 8	Cefeida KIC 3733346	23
Obrázek 9	Proměnná hvězda KIC 9832227	23
Obrázek 10	Kataklyzmická proměnná hvězda KIC 9406652	23
Obrázek 11	Světelná křivka Kepler-10	23
Obrázek 12	Složená světelná křivka	23
Obrázek 13	Globální pohled na tranzit Kepler-10 c	24
Obrázek 14	Lokální pohled na tranzit Kepler-10 c	24
Obrázek 15	Metoda radiálních vzdáleností	26
Obrázek 16	Spektrum slunečního záření	27
Obrázek 17	Radiální rychlost hvězdy 51 Pegasi	28
Obrázek 18	Astrometrická metoda	30
Obrázek 19	Kolísání hvězdy Gliese 876 s planetou	30
Obrázek 20	Model formálního neuronu	34
Obrázek 21	Příklady aktivačních funkcí neuronu	36
Obrázek 22	Lineárně separovatelná (vlevo) a neseparovatelná (vpravo) úloha . . .	36
Obrázek 23	Zapojení neuronů	37
Obrázek 24	Architektura dopředné vícevrstvé umělé neuronové sítě	37
Obrázek 25	Ručně psané číslice 9.	38
Obrázek 26	Vztah mezi počítačovým viděním, strojovým učením a konvoluční sítí	38
Obrázek 27	Příklad konvoluční sítě pro rozpoznávání ručně psaných číslic	39
Obrázek 28	Postupné posouvání konvolučního filtru přes vstup	40
Obrázek 29	Příklad aplikace konvolučního filtru na vstup	41
Obrázek 30	Příklad operace Max-Pooling	42

Obrázek 31	Příklad fungování vrstvy Flatten	43
Obrázek 32	Příklad fungování vrstvy Dropout	43
Obrázek 33	Příklad algoritmu zpětného šíření chyby	45
Obrázek 34	Komponenty projektu a jejich komunikace	48
Obrázek 35	Architektura serveru	48
Obrázek 36	Architektura webové aplikace	51
Obrázek 37	Stavy procesů a přechody mezi nimi	51
Obrázek 38	Trénovací množina	52
Obrázek 39	Část datasetu s vlastnostmi hvězd	53
Obrázek 40	Část datasetu s vlastnostmi planet	54

SEZNAM TABULEK

Tabulka 1	Fáze oběhu tranzitující exoplanety	22
Tabulka 2	Veličiny tranzitu	22
Tabulka 3	Příklady výpočtu hmotnosti planet	29
Tabulka 4	Komponenty formálního neuronu	33
Tabulka 5	Parametry plně propojené vrstvy	40
Tabulka 6	Parametry konvoluční vrstvy	41
Tabulka 7	Parametry vrstvy Max-Pooling	42
Tabulka 8	Parametry vrstvy Dropout	43
Tabulka 9	Údaje o hvězdách ukládané do databáze	53
Tabulka 10	Údaje o planetách ukládané do databáze	54
Tabulka 11	Pojmenování soustavy Kepler-10 v různých katalozích	55

SEZNAM ZDROJOVÝCH KÓDŮ

Zdrojový kód 1	Vytvoření modelů v REST API.	49
Zdrojový kód 2	Vytvoření koncového bodu v REST API.	49
Zdrojový kód 3	Ukázka komunikace pomocí socket.io.	50

SEZNAM VZORCŮ

1	Pravděpodobnost zpozorování tranzitu planety přes hvězdu	21
2	Výpočet velké poloosy dráhy planety	24
3	Výpočet poloměru planety	25
4	Odhad průměrné rychlosti oběhu planety	25
5	Výpočet inklinace dráhy	25
6	Radiální rychlost na základě změny vlnové délky	28
7	Radiální rychlost	29
8	Obecný vztah vstupu neuronu a jeho váhy	34
9	Lineární vztah vstupu neuronu a jeho váhy	35
10	Agregační funkce neuronu	35
11	Hyperbolicko-tangenciální aktivační funkce	35
12	Sigmoidální aktivační funkce	35
13	Gaussova aktivační funkce	35
14	Provádění konvoluce pro výřez vstupu a filtr	41
15	Výpočet chyby pro jeden vzor u algoritmu zpětného šíření chyby	44
16	Přírůstek váhy u algoritmu zpětného šíření chyby	44
17	Výpočet lokálního gradientu neuronu	45

SEZNAM ZKRATEK

CNN	Convolutional neural network
csv	Comma-separated values
FC	Fully-connected layer
FFNN	Feed-forward neural network
fits	Flexible image transport system
ly	Light year
au	Astronomical unit

ÚVOD

V naší sluneční soustavě se nachází celkem 8 dosud objevených planet včetně Země. Mimo ni ale v pozorovatelném vesmíru existují odhadem stovky miliard galaxií a v každé z nich v průměru stovky miliard hvězd. Z toho, co o vzniku a fungování hvězdných soustav víme je pravděpodobné, že většinu těchto hvězd bude obíhat jedna nebo více planet, tzv. extrasolárních planet nebo také exoplanet. [8]

První potvrzená exoplaneta byla objevena již roku 1992, ale výzkum exoplanet se dostal do oblasti širokého zájmu až během posledního desetiletí. Stalo se tak především kvůli vesmírnému teleskopu Kepler, který má na svém kontě od roku 2009 přes 2 500 objevených exoplanet. [9, 12]

K dnešnímu dni je známo více jak 4 000 potvrzených exoplanet. Toto číslo se s nejvyšší pravděpodobností bude rychle zvyšovat, protože roku 2018 byl vypuštěn nástupce Kepleru – satelit TESS – od něhož je očekáván objev 20 000 exoplanet. [14]

Planety u jiných hvězd většinou nelze pozorovat přímo. Proto je nepřímými metodami zkoumáno jejich působení na své mateřské hvězdy, které už pozorovat lze. Výstupem z takovýchto pozorování jsou často stovky GiB fyzikálních a statistických dat, jež je následně nutno zpracovat. [8]

Cílem této diplomové práce je vytvořit aplikaci umožňující uživatelům poskytovat výpočetní výkon svých počítačů pro analýzu právě těchto dat. Projekt sestává z klientského programu, webové aplikace a serveru. Klientský program provádí potřebné distribuovatelné výpočty na počítači uživatele. Tento program je možné ovládat z rozhraní webové aplikace, jež zároveň poskytuje přehled o všech aktivitách, uživatelích a datech. Rozdělování výpočetních úloh mezi klienty a ukládání dat do databáze pak řeší server.

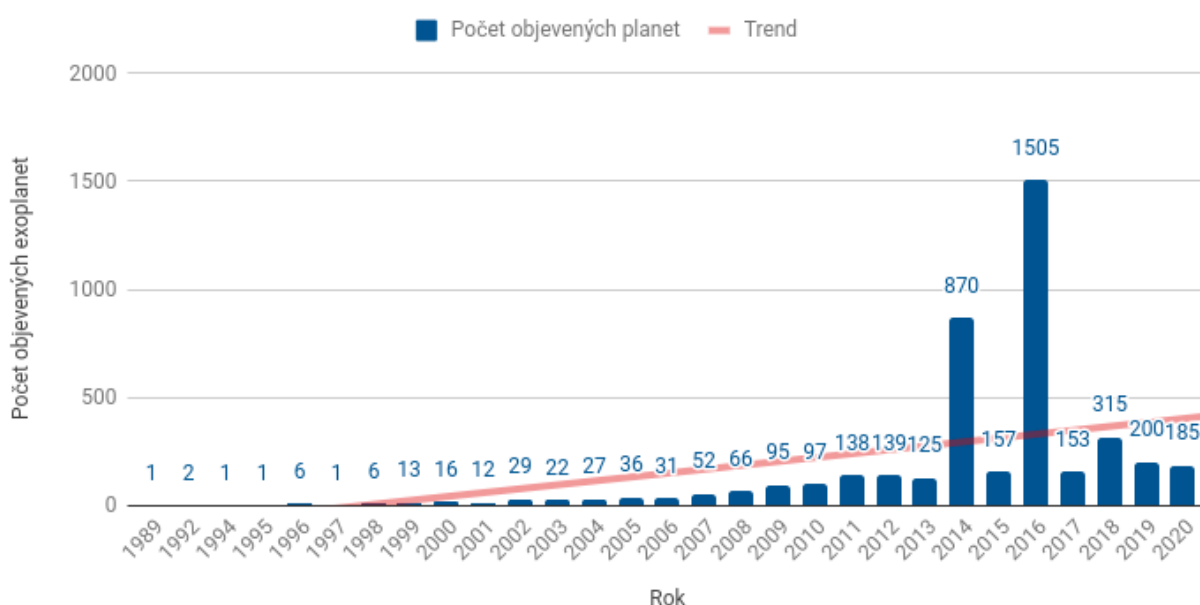
Díky distribuovaným výpočtům se do výzkumu exoplanet bude moci bez vysokého úsilí, znalosti či technického vybavení zapojit i široká veřejnost. To může urychlit vývoj a zároveň zvýšit povědomí o této vědní disciplíně.

V projektu jsou využity některé techniky spadající pod umělou inteligenci, v důsledku čehož je zpracovávání dat zcela automatizované. Platí však, že umělá inteligence je v současnosti stále intenzivně se rozvíjející oblastí, a proto výsledky nemusí být natolik vypovídající ve srovnání s tím, kdy by výzkum prováděli lidé manuálně, byť by to trvalo nesrovnatelně déle.

1 HLEDÁNÍ EXOPLANET

Pouhým zkoumáním planet v naší sluneční soustavě se omezujeme na velice specifické podmínky existující v okolí našeho Slunce. Pro hlubší pochopení fungování planetárních systémů je nutné rozšířit oblast zájmu i na planety v okolí jiných hvězd – tzv. exoplanety. Můžeme se tak přiblížit odpovědím na otázky jako „Jak vzácné jsou podmínky pro život ve vesmíru?“ nebo „Jak vznikla a jak se vyvíjela naše planeta?“ [8]

Na počátku 90. let minulého století byla objevena první exoplaneta v okolí pulsaru a v roce 1995 první exoplaneta v okolí hvězdy podobné Slunci. Od té doby frekvence objevů planet v průměru neustále stoupá. [8, 12]

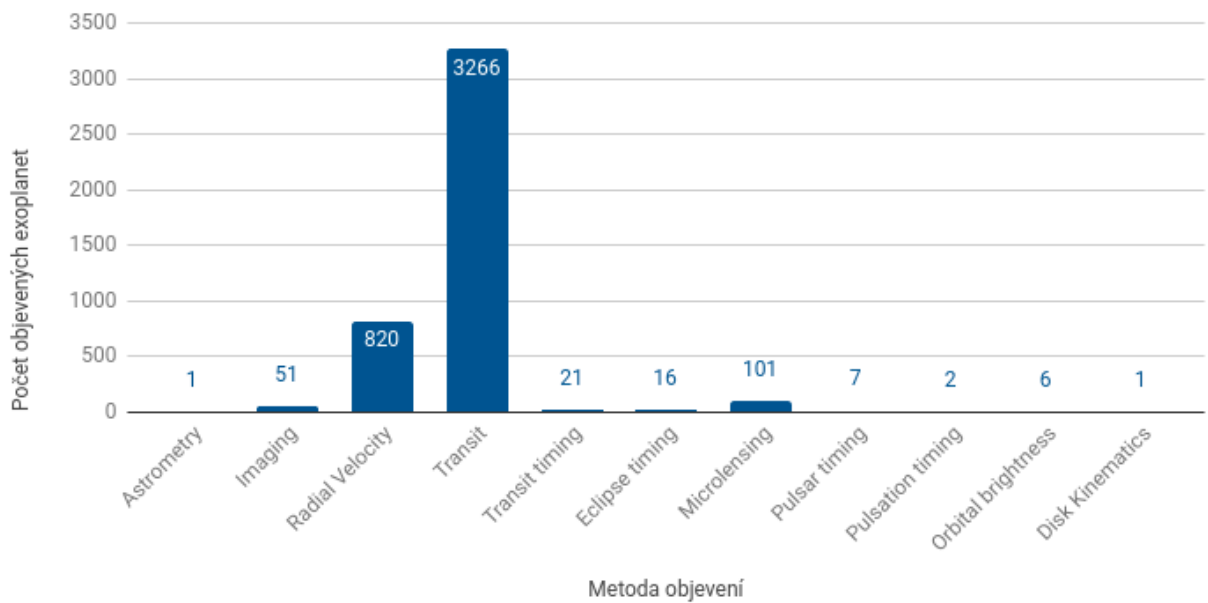


Obrázek 1: Četnost objevů exoplanet v jednotlivých letech ¹

Cestovat k jiným hvězdám a posílat sondy k exoplanetám je však naprosto mimo možnosti naší současné technologie. Dokonce i přímé pozorování exoplanet teleskopem je ve většině případů nemožné. Téměř veškerá pozorování se tak provádí skrze nepřímé metody. Ty využívají skutečnosti, že i když není možné spatřit exoplanetu samotnou, je možné detekovat její působení na své okolí (např. na mateřskou hvězdu).

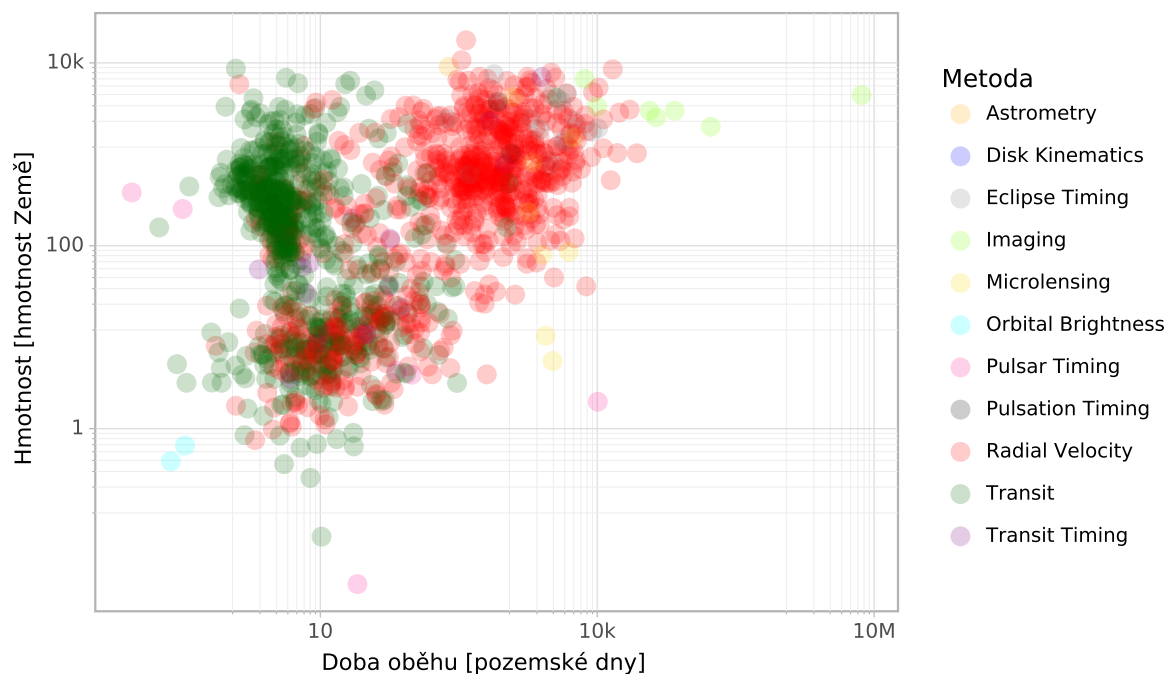
Jednotlivé metody budou popsány v následujících podkapitolách. Zdaleka nejvýznamnější je tranzitní metoda, kterou byla objevena většina exoplanet. Velké množství planet bylo objeveno taktéž metodou radiálních rychlostí. [12]

¹Vytvořeno autorem, zdroj dat: [12].



Obrázek 2: Počty objevených exoplanet jednotlivými metodami ¹

Dá se říci, že každá metoda je vhodnější pro objevování různých typů planet. Málo hmotné planety byly objevovány častěji tranzitní metodou, zatímco hmotnější planety spíše metodou radiálních rychlostí. Pro planety vzdálené od své mateřské hvězdy se nejlépe osvědčila metoda přímého zobrazení. [12]

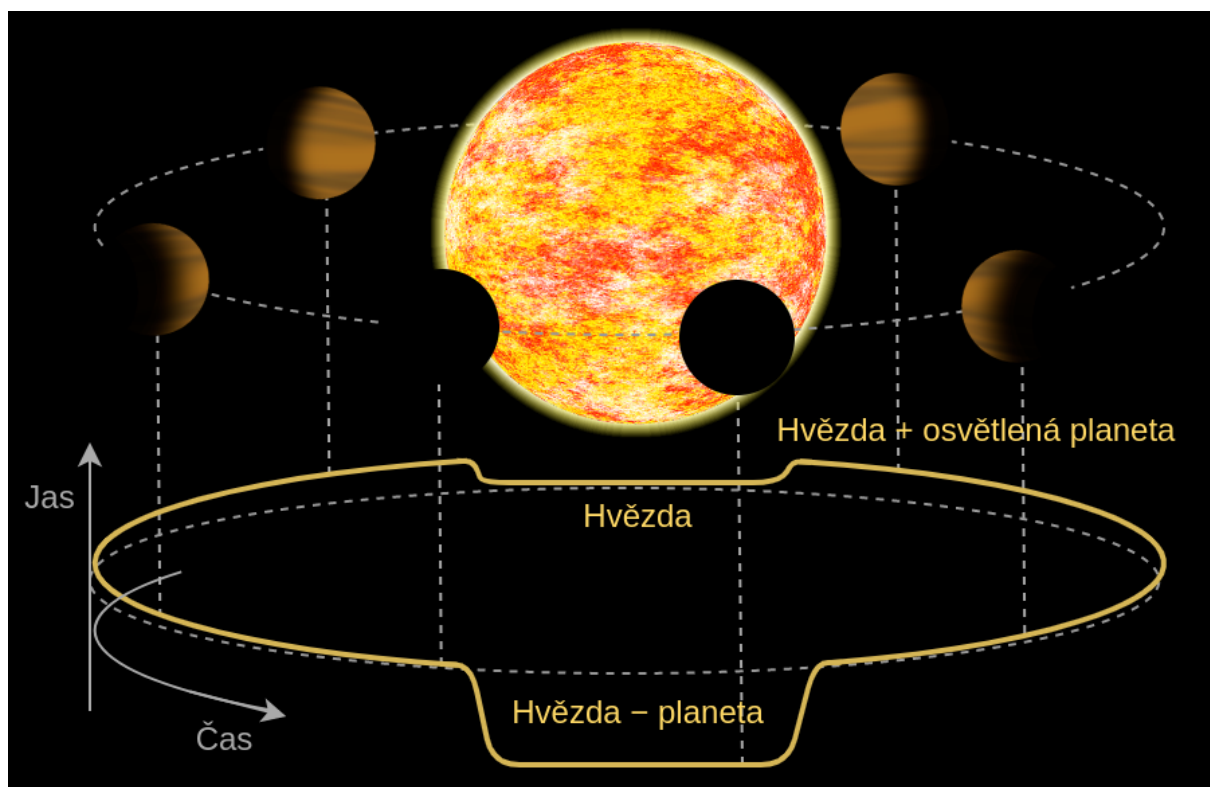


Obrázek 3: Známé exoplanety dle jejich vlastností a metody objevení ¹

¹Vytvořeno autorem, zdroj dat: [12].

1.1 Tranzitní metoda

Někdy se planeta při obíhání dostane mezi svou hvězdu a Zemi. Tento jev se pro pozorovatele na Zemi projeví jako mírný pokles jasu hvězdy (obvykle ve zlomku procenta). Při dlouhodobém pozorování je možné v těchto změnách jasu hvězdy odhalit opakující se složku. To by mohlo indikovat přítomnost planety v blízkosti této hvězdy. [8, 6]



Obrázek 4: Přechod planety přes kotouč hvězdy ¹

Tyto změny však nemusí být na první pohled viditelné, protože v soustavě může být více planet, které svou hvězdu zastiňují různou měrou a obíhají kolem ní s různou periodou. Navíc i v situaci, kdy je ve změnách jasu hvězdy objevena periodická složka nemusí jít vždy o obíhající planetu. Hvězda může být např. sama o sobě proměnlivá nebo se může jednat o dvojhvězdu, jejíž složky se vzájemně zastiňují. [9]

Tranzitní metoda vzbuzuje velký zájem především kvůli možnosti objevovat i malé planety podobné Zemi – takové planety by mohly spíše splňovat podmínky pro život. Nevýhodou je, že většina exoplanet obíhá svou hvězdu v takové rovině, v jaké pozorovatel na Zemi nemůže transit spatřit. Odhadem 99 % všech potenciálních exoplanet podobných Zemi nemůže být tranzitní metodou nikdy zachyceno. [11, 6]

¹Vytvořeno autorem v <https://www.draw.io> a GIMP.

$$P = \frac{d_s}{a} \quad (1)$$

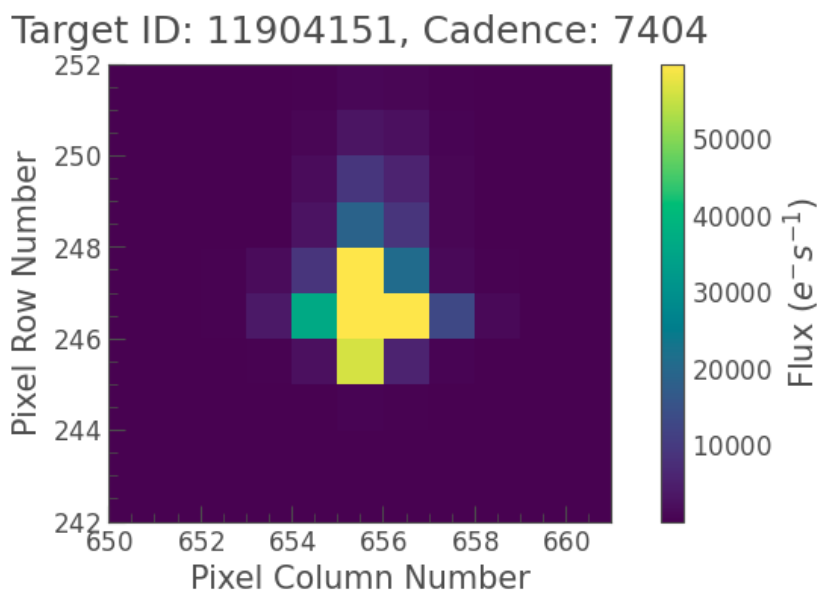
Vzorec 1: Pravděpodobnost zpozorování tranzitu planety přes hvězdu

d_s = průměr hvězdy a = vzdálenost exoplanety od hvězdy

1.1.1 Target pixel file

Prvním krokem v analýze hvězdy tranzitní metodou je její fyzické pozorování. Teleskop obvykle pozoruje část oblohy po dobu několika měsíců, přičemž každých několik desítek minut vytvoří snímek dané části oblohy. Z výsledných fotografií se následně vyextrahují jednotlivé hvězdy, čímž vzniknou tzv. target pixel files.

TPF obsahuje část oblohy o velikosti několika pixelů, na které se v původní fotografii nacházela zkoumaná hvězda a její okolí. Barva pixelů je určena jasnem.

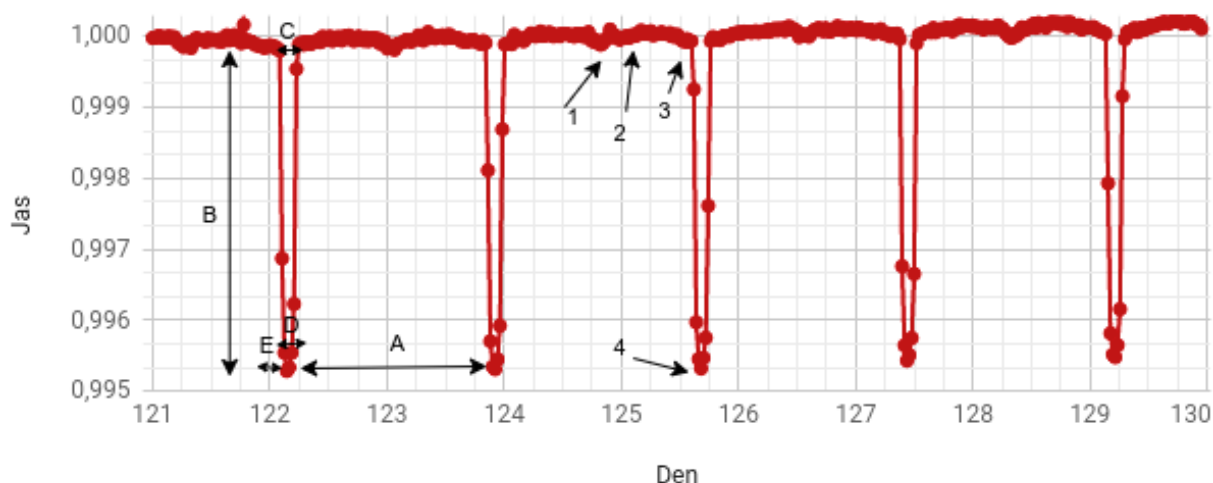


Obrázek 5: Target pixel file hvězdy Kepler-10 ¹

1.1.2 Světelná křivka hvězdy

Po složení všech TPF do časové řady a vypočítání jejich jasů dostaneme světelnou křivku. Na obrázku 5 je světelná křivka hvězdy Kepler-13 očištěná od dlouhodobého trendu, šumu a extrémních hodnot. Křivka vykazuje velice výraznou periodickou složku s periodou 1,763 dne. Ve většině případů ale vliv planety není takto výrazný a detekovat planetu je obtížnější.

¹Vytvořeno autorem, zdroj dat: [12].



Obrázek 6: Světelná křivka soustavy Kepler-13 ¹

Na obrázku 6 jsou čísla označeny jednotlivé fáze dále popsané v tabulce 1:

Fáze	Co vidí pozorovatel na Zemi
1 Planeta je za hvězdou (sekundární zákryt).	Hvězda
2 Planeta je vedle hvězdy.	Hvězda + osvětlená část planety
3 Planeta je vedle hvězdy.	Hvězda + neosvětlená část planety
4 Planeta je před hvězdou (tranzit).	Hvězda – planeta

Tabulka 1: Fáze oběhu tranzitující exoplanety

Dále jsou na stejném obrázku písmeny označeny důležité veličiny:

Veličina	Popis
A Perioda	Perioda tranzitu udává periodu oběhu planety kolem hvězdy.
B Hloubka	Čím větší je hloubka tranzitu, tím je planeta vůči hvězdě větší.
C Trvání	Čím je trvání delší, tím delší trajektorii přes hvězdu planeta má.
D Trvání minima	Závisí na úhlu mezi rovinou oběhu planety vůči pozorovateli
E Trvání nástupu	

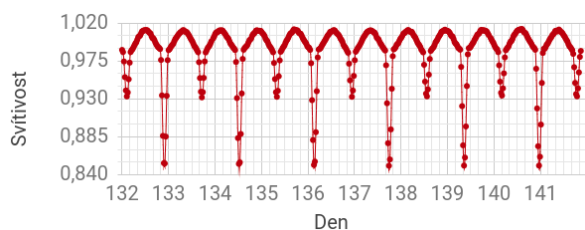
Tabulka 2: Veličiny tranzitu

TODO: Transit types image.

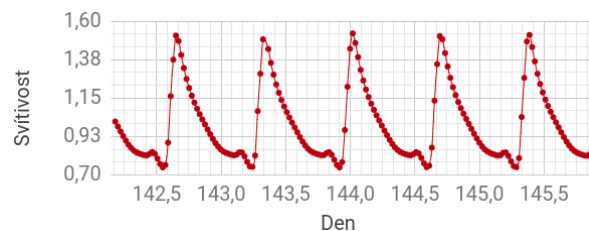
¹Vytvořeno autorem, zdroj dat: [12].

1.1.3 Vyřazení false positives

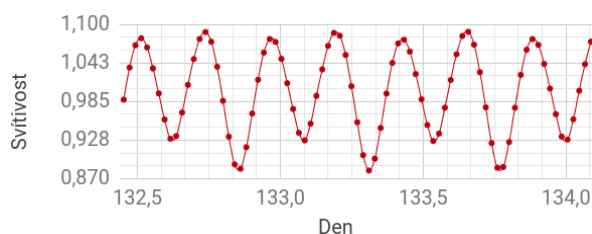
Většina periodických složek ve světelných křivkách hvězd jsou *false positive* – patří jiným jevům, než je obíhající planeta. Tyto případy je třeba odfiltrovat, což byla až donedávna především manuální práce lidí – vědců či dobrovolníků. Protože ale tranzit planety vykazuje specifický průběh popsany v předchozí kapitole, je možné ho s určitou úspěšností rozpoznat pomocí naučené umělé neuronové sítě automaticky. [9]



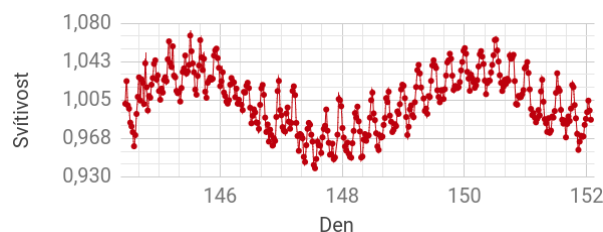
Obrázek 7: Dvojhvězda KIC 8262223 ¹



Obrázek 8: Cefeida KIC 3733346 ¹

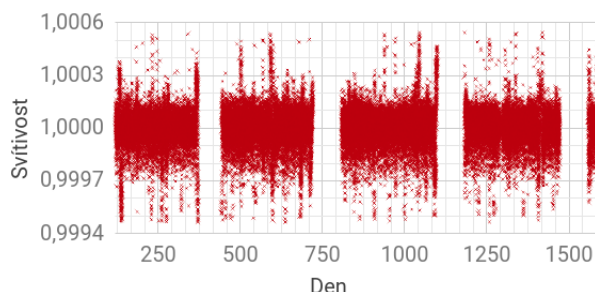


Obrázek 9: Proměnná hvězda KIC 9832227 ¹

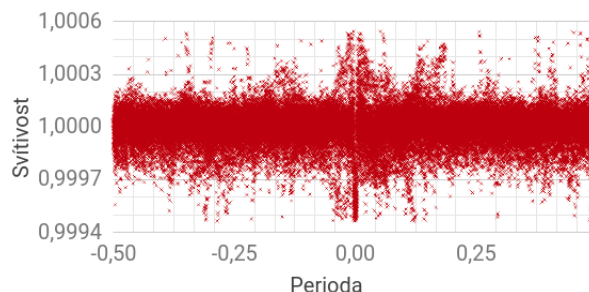


Obrázek 10: Kataklyzmická proměnná hvězda KIC 9406652 ¹

Je třeba aby vstupní data do neuronové sítě měla stejné rozměry i formát. Vzhledem k různorodosti světelných křivek je nutno provést několik kroků, abychom dosáhli standardizovaného formátu. Prvním krokem je složení časové řady do jedné periody, čímž dojde k posílení viditelnosti transitu (pokud zde nějaký je), nebo naopak k jeho vyrušení (pokud zde žádný není). [9]



Obrázek 11: Světelná křivka Kepler-10 ¹

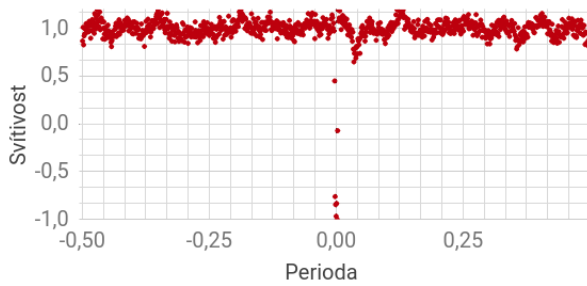


Obrázek 12: Složená světelná křivka ¹

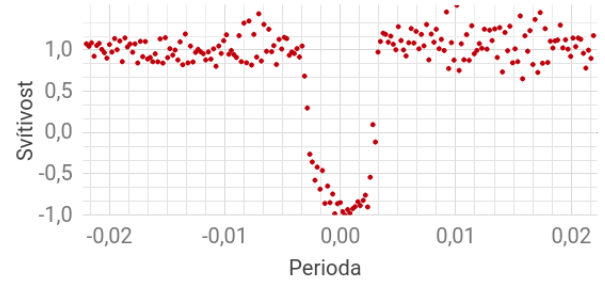
¹Vytvořeno autorem, zdroj dat: [12].

Na obrázku 12 je uprostřed slabě patrný transit. Má malou šířku, protože trvá pouze 0,25 dne, zatímco celá perioda je dlouhá 45,3 dne. Z této složené časové řady se vytvoří dva pohledy, které budou vstupem do neuronové sítě:

- **Globální pohled** (obr. 13) – Šířka periody a počet bodů v časové řadě jsou fixní. Nevýhodou je, že u planet s dlouhou periodou bude transit velice nepatrný, proto pouze globální pohled nestačí. [9]
- **Lokální pohled** (obr. 14) – Šířka tranzitu a počet bodů v časové řadě jsou fixní. Nevýhodou je, že není viditelný celý průběh světelné křivky. Naproti tomu je ale zřetelný transit. [9]



Obrázek 13: Globální pohled na transit Kepler-10 c ¹



Obrázek 14: Lokální pohled na transit Kepler-10 c ¹

Fixní počet bodů lze zajistit nahrazením každých $\frac{\alpha}{\beta}$ sousedních bodů (α – současný počet bodů, β – požadovaný počet bodů) jediným, který bude reprezentovat jejich medián. Pro potřeby neuronové sítě je nutno oba pohledy taktéž normalizovat tak, aby platilo $H = \langle 1; -1 \rangle$. U případů, které neuronová síť vyhodnotí jako planety, je možné pokračovat výpočtem dalších informací o planetě. [9]

1.1.4 Výpočet vlastností planety

Velkou poloosu dráhy planety lze vypočítat, pokud známe hmotnost hvězdy a periodu oběhu z třetího Keplerova zákona. [7]

$$a = \sqrt[3]{\frac{GMP^2}{4\pi^2}} \quad (2)$$

Vzorec 2: Výpočet velké poloosy dráhy planety

a = velká poloosa G = gravitační konstanta M = hmotnost hvězdy P = perioda oběhu planety

Ze světelné křivky a poloměru hvězdy lze vypočítat poloměr tranzitující planety po vyjádření z následující rovnice [6, 7]:

¹Vytvořeno autorem, zdroj dat: <https://exoplanetarchive.ipac.caltech.edu>.

$$\frac{r^2}{R^2} = \frac{\Delta F}{F} \quad (3)$$

Vzorec 3: Výpočet poloměru planety

r = poloměr planety R = poloměr hvězdy F = jas hvězdy ΔF = změna jasu

Dále je možno odhadnout i průměrnou rychlost pohybu planety po oběžné dráze. Skutečná průměrná rychlost však může být jiná, protože se nepočítá s excentricitou dráhy [7]:

$$v \approx \frac{2\pi a}{T} \quad (4)$$

Vzorec 4: Odhad průměrné rychlosti oběhu planety

v = rychlost oběhu planety a = velká poloosa dráhy planety T = perioda oběhu planety

Další z důležitých charakteristik orbity je inklinace (sklon), která nám řekne, jaký úhel svírají roviny pozorovatele a oběhu exoplanety. Bohužel lze zjistit minimální úhel sklonu dráhy a nikoliv jeho přesnou hodnotu. Inklinace se bude zpravidla blížit 90° . [7]

$$\cos i \leq \frac{R + r}{a} \quad (5)$$

Vzorec 5: Výpočet inklinace dráhy

i = inklinace R = poloměr hvězdy r = poloměr planety a = velká poloosa dráhy planety

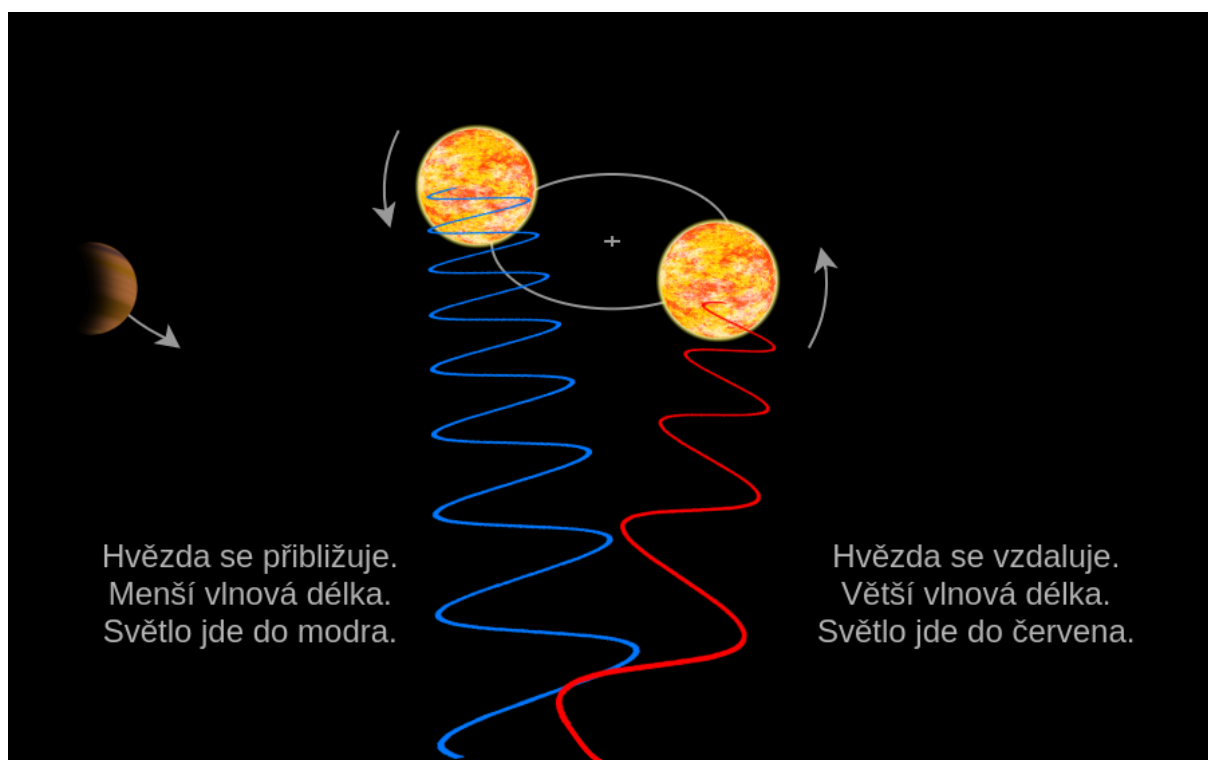
1.1.5 Výpočet hmotnosti planety

Hmotnost planety z tranzitní metody nelze vypočítat. Avšak z empirických dat je možné ji odhadnout na základě kombinace jiných veličin pomocí umělé neuronové sítě. TODO. [10]

1.2 Metoda radiálních rychlostí

Stejně jako hvězda ovlivňuje obíhající planetu, tak i planeta gravitačně ovlivňuje svou hvězdu a obě tělesa obíhají kolem společného těžiště. Tento pohyb se může projevit jako opakované přibližování a vzdalování hvězdy vůči pozorovateli na Zemi. Právě pojem radiální rychlost označuje rychlost pohybu ve směru přímky k pozorovateli. [11]

Pokud se zdroj elektromagnetického záření (hvězda) přibližuje vůči pozorovateli, záření má menší vlnovou délku a jeví se více do modra, protože právě modrá (a fialová) barva má z viditelného spektra nejmenší vlnovou délku. Obdobná situace nastává při vzdalování se zdroje vlnění od pozorovatele. Vlnová délka se zvětšuje a barva jde do červena. Tomuto efektu se říká červený (resp. modrý) posuv. [11]



Obrázek 15: Metoda radiálních vzdáleností ¹

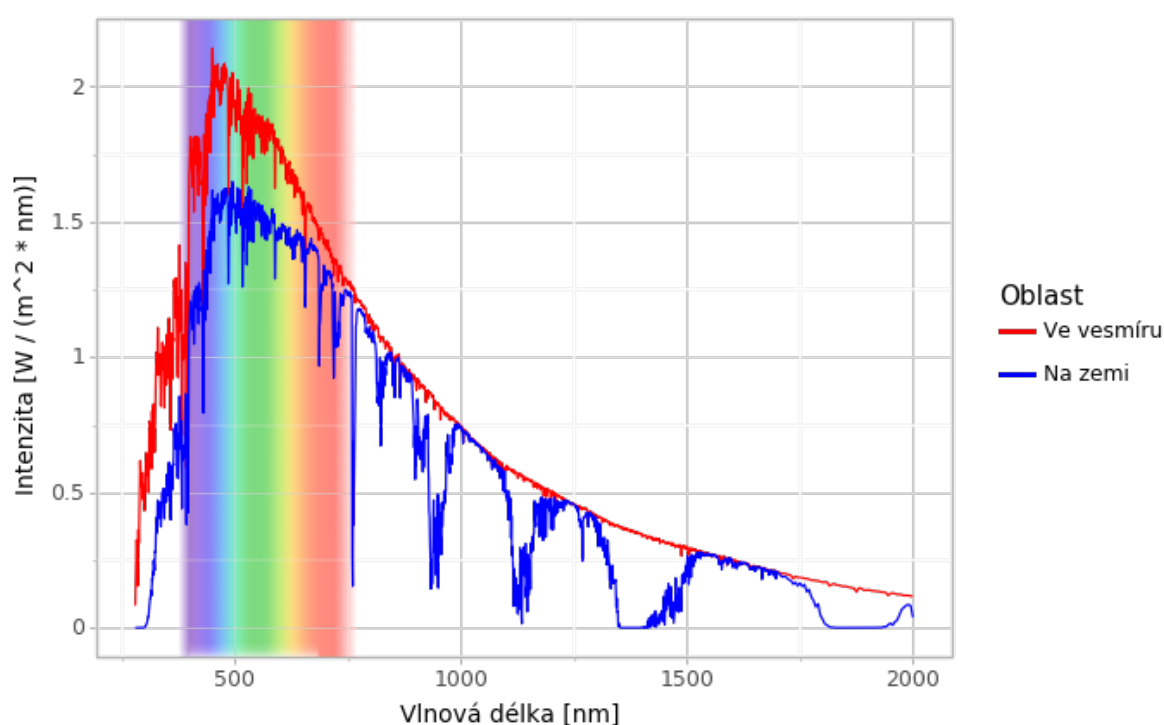
Příčinou červeného/modrého posuvu je v tomto případě Dopplerův jev, který lze uplatnit i pro jiné druhy vlnění, než to elektromagnetické – zvuk. Pokud se k nám zdroj zvuku přibližuje (např. siréna na jedoucím autě), zvuk zpravidla vnímáme vyšším tónem, protože má menší vlnovou délku (vyšší frekvenci). Při vzdalování zdroje má zvuk větší vlnovou délku a je vnímán hlubším tónem. [4]

¹Vytvořeno autorem v <https://www.draw.io> a GIMP.

Periodicky se opakující změny ve vlnové délce záření hvězdy tak mohou být důsledkem existence tělesa v této soustavě. [11]

1.2.1 Měření vlnové délky záření hvězdy

Hvězdy nevyzařují světlo pouze jedné jediné vlnové délky, nýbrž celé spektrum. Tento efekt lze vidět u duhy v zemské atmosféře, kdy jsou jednotlivé složky slunečního světla odděleny. Záření z hvězd se dokonce neomezuje pouze na viditelné světlo, ale pokrývá značnou část celého elektromagnetického spektra. Jak velká část záření přísluší jednotlivým vlnovým délkám lze měřit pomocí spektrometru. [4, 13]



Obrázek 16: Spektrum slunečního záření ¹

Změna vlnové délky záření se projevuje jako horizontální posun barevného spektra hvězdy. Nutno podotknout, že zemská atmosféra některé vlnové délky pohlcuje. Proto i přesto, že Slunce má barvu spíše do zelena vidíme tuto hvězdu ze Země žlutě. [TODO]

1.2.2 Výpočet radiální rychlosti hvězdy

Poté, co teleskop sesbírá dostatečně velkou časovou řadu vlnových délek záření hvězdy může dojít k vypočítání změn radiální rychlosti v čase. Lze tak učinit dle vzorce 3. Platí,

¹Vytvořeno autorem, zdroj dat: [13].

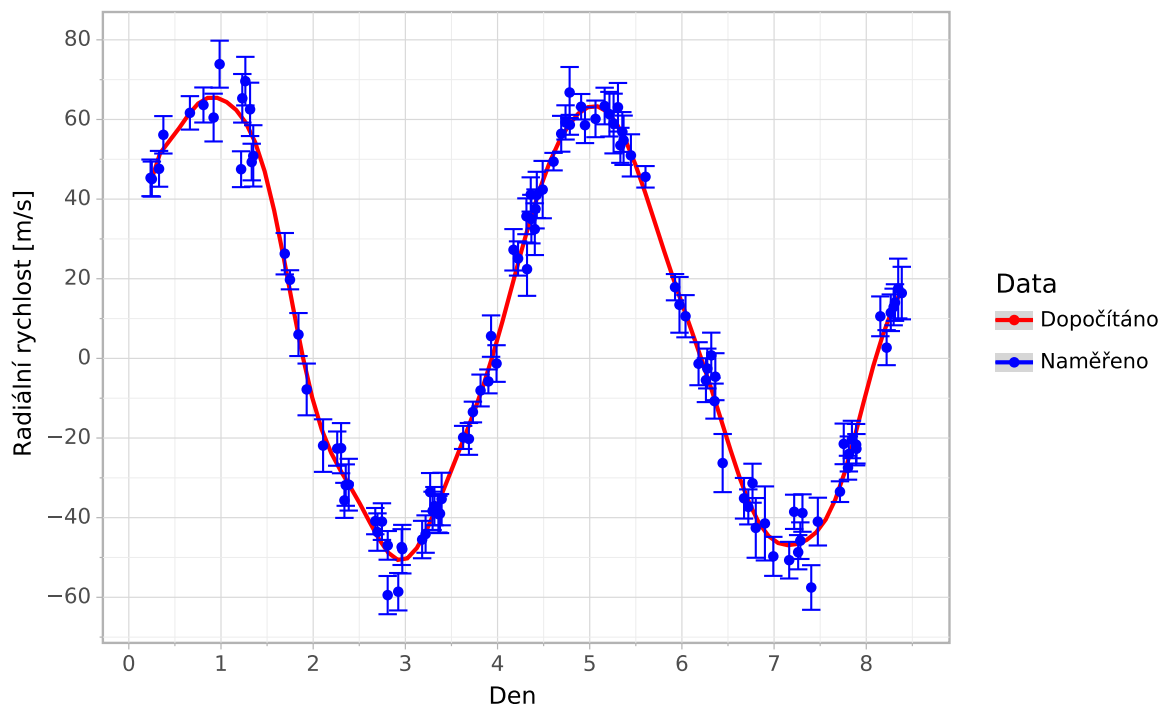
že radiální rychlost je kladná, pokud se zdroj od pozorovatele vzdaluje a záporná pokud se přibližuje. [11]

$$v = c * \frac{\Delta\lambda}{\lambda_0} \quad (6)$$

Vzorec 6: Radiální rychlost na základě změny vlnové délky

$\Delta\lambda$ = změna vlnové délky λ_0 = klidová vlnová délka v = radiální rychlost

Graf znázorňující radiální rychlost hvězdy 51 Pegasi může vypadat takto:



Obrázek 17: Radiální rychlost hvězdy 51 Pegasi ¹

Na první pohled je patrná jedna periodická složka s periodou 4,23 dne a amplitudou 56,04 $\frac{m}{s}$, která značí, že kolem této hvězdy obíhá jedna planeta. Pokud zde jsou přítomna i jiná tělesa, důvodů, proč je metoda radiálních rychlostí neodhalila může být několik:

- Hmotnost tělesa je vůči hmotnosti hvězdy zanedbatelná,
- Perioda oběhu tělesa je příliš velká (desítky let a více), a proto se ji nepodařilo zachytit na tak krátkém časovém úseku,
- Těleso obíhá po dráze, jejíž rovina je kolmá k přímce směrem k pozorovateli (k Zemi). To vede k tomu, že je hvězda vychylována takovým směrem, který nemá vliv na radiální rychlost hvězdy vůči Zemi.

¹Vytvořeno autorem, zdroj dat: [5].

1.2.3 Výpočet hmotnosti planety

Hlavní výhodou metody radiálních rychlostí oproti tranzitní metodě je možnost spočítat hmotnost exoplanety. Její přesnou hodnotu však lze odvodit pouze se znalostí sklony dráhy exoplanety vůči pozorovateli. V opačném případě lze vypočítat pouze dolní mez hmotnosti planety, a to zejména kvůli $M_p * \sin(i)$ ve vzorci 5. [11]

$$\Delta v_{max} = \sqrt[3]{\frac{2\pi G}{T}} * \frac{M_p * \sin(i)}{\sqrt[3]{(M_p + M_s)^2}} * \frac{1}{\sqrt{1 - e^2}} \quad (7)$$

Vzorec 7: Radiální rychlost

Δv_{max} = amplituda změny rychlosti M_s = hmotnost hvězdy M_p = hmotnost planety
 $\sin(i)$ = sklon dráhy vůči pozorovateli e = excentricita dráhy T = oběžná doba

V tabulce 2 jsou uvedeny příklady výpočtu hmotnosti některých planet.

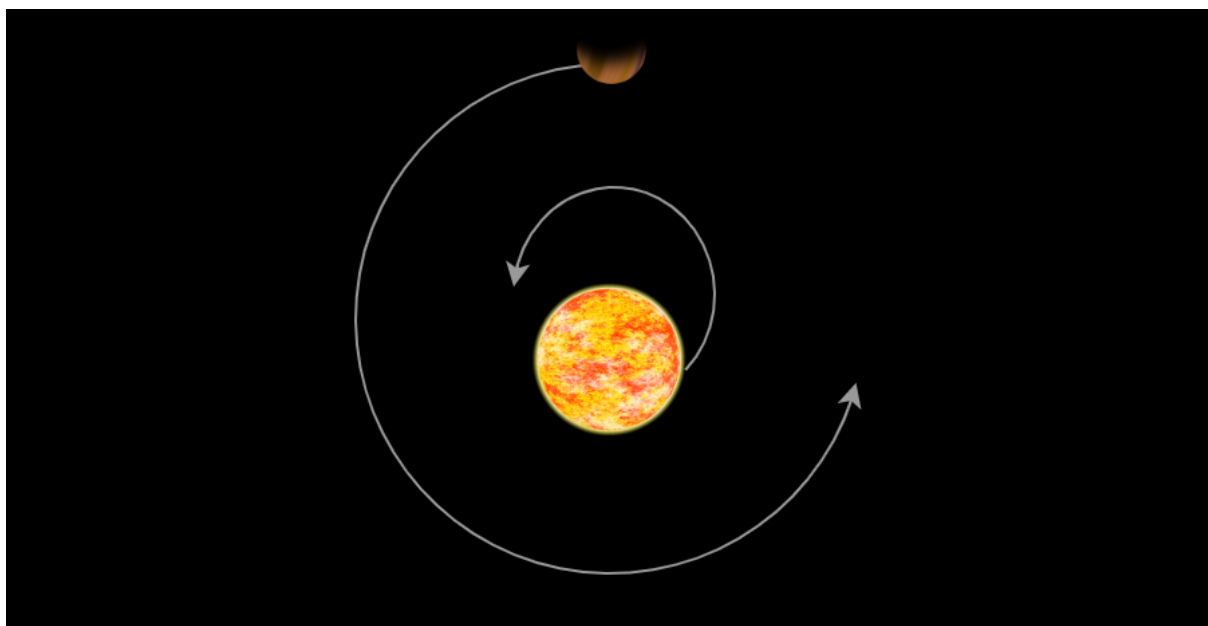
Těleso	Hvězda	$\Delta v_{max} [\frac{m}{s}]$	$M_p [kg]$	$\sin(i)$	e	T [r]	$M_s [kg]$
Země	Slunce	0,089	$2 * 10^{30}$	1	0,017	1	$5,97 * 10^{24}$
Jupiter		12,4			0,048	11,86	$1,9 * 10^{27}$
Pluto		0,00003			0,247	247,41	$1,3 * 10^{22}$
α Cen Bb	α Cen B	0,51	$1,8 * 10^{30}$		0	0,0089	$6,75 * 10^{24}$
51 Pegasi b	51 Pegasi	55,9	$2,22 * 10^{30}$		0,013	0,0116	$0,88 * 10^{27}$

Tabulka 3: Příklady výpočtu hmotnosti planet ¹

¹Vytvořeno autorem, zdroj dat: [4, 5, 11, 12].

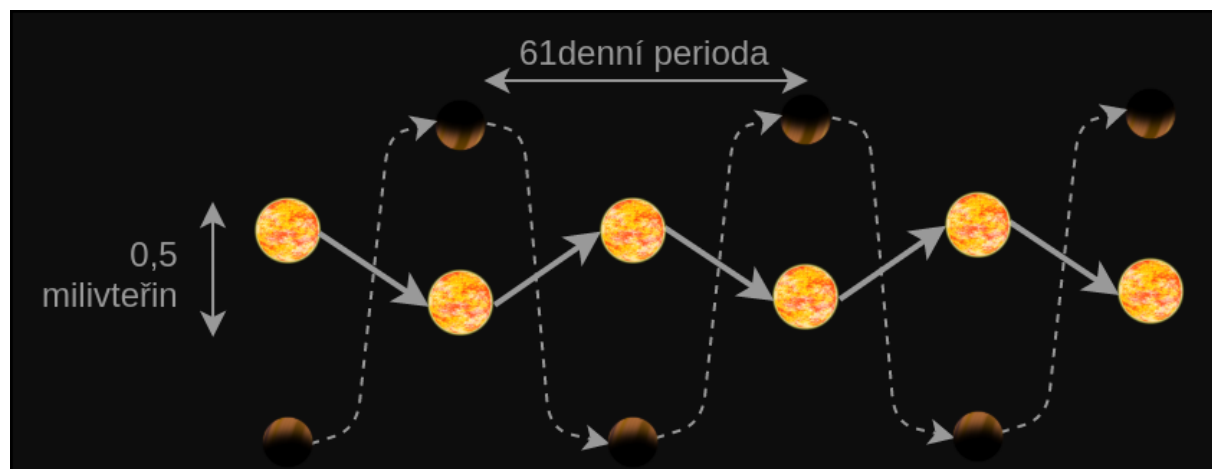
1.3 Astrometrická metoda

Astrometrická metoda využívá stejné vlastnosti vzájemného působení těles jako metoda radiálních rychlostí. Namísto zkoumání vlnové délky záření se však zaměřuje na polohu hvězdy. Hvězda, kolem níž obíhá dostatečně hmotné těleso, se bude v důsledku působení tělesa nepatrně vychylovat ze své pozice – bude obíhat kolem těžiště soustavy. [11]



Obrázek 18: Astrometrická metoda ¹

Pohyb hvězdy tak není přímočarý, ale vlnitý. Kolísání hvězdy je však pro pozorovatele na Zemi často pouze v řádu stovek úhlových mikrovteřin až jednotek milivteřin. [11] Z tohoto důvodu byla astrometrickou metodou dosud objevena pouze jediná exoplaneta. [12]



Obrázek 19: Kolísání hvězdy Gliese 876 s planetou ¹

¹Vytvořeno autorem v <https://www.draw.io> a GIMP.

Dá se však očekávat, že se zlepšující se technikou bude tato metoda úspěšnější.

1.4 Gravitační mikročočky

1.5 Přímé zobrazení

1.6 Transit timing

1.7 Pulsar timing

2 UMĚLÁ INTELIGENCE

2.1 Umělé neuronové sítě

2.2 Evoluční algoritmy

2.2.1 Genetický algoritmus

2.2.2 Algoritmus diferenciální evoluce

2.3 Fuzzy systémy

2.4 Expertní systémy

3 UMĚLÉ NEURONOVÉ SÍTĚ

Jednou z oblastí patřících do oboru umělé inteligence jsou umělé neuronové sítě, jež se inspirovaly biologickými neuronovými sítěmi. Jejich použití je vhodné na problémy, u kterých neexistuje přesný matematický popis řešení nebo sice existuje, ale jeho realizace by byla příliš složitá. Hlavními oblastmi, kde se neuronové sítě využívají, jsou např.:

- **Predikce** (předpověď počasí, vývoj cen akcií na burze, ...),
- **Rozpoznávání vzorů** (rozpoznávání ručně psaného textu, detekce tranzitu planety ve světelné křivce hvězdy, ...),
- **Analýza nebo transformace signálů** (odstranění šumu ze signálu, komprese, převod psaného textu na mluvený signál, ...),
- **Řízení v dynamicky se měnících podmínkách** (autopilot, ...). [2]

3.1 Neuron

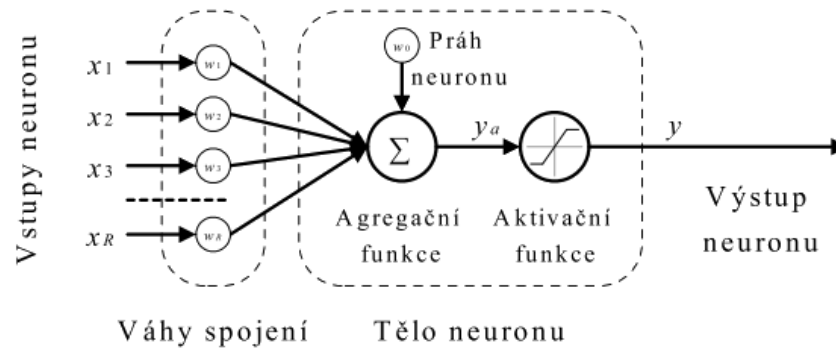
Základním stavebním prvkem umělé neuronové sítě je umělý neuron, který reprezentuje zjednodušenou biologickou nervovou buňku. Nejpoužívanějším typem umělého neuronu je tzv. formální neuron. Ten se skládá z následujících částí:

Komponenta	Popis	Příklad
Vstupy $x_{1...R}$	Jeden nebo více vstupů.	[0.23, 0.58, -0.15]
Váhy spojení (vstupů) $w_{1...R}$	Každý vstup má svou váhu.	[0.01, 0.56, 0.12]
Práh neuronu w_0	Přičte se ke vstupu do aktivační funkce.	0.17
Agregační funkce \sum	Vstupy a váhy transformuje na jednu hodnotu – potenciál y_a .	$y_a = w_0 + \sum_{n=1}^R x_n w_n$
Aktivační funkce	Převéde hodnotu vstupního potenciálu y_a na výstup neuronu y .	$y = y_a$ $y = \tanh y_a$

Tabulka 4: Komponenty formálního neuronu ¹

Jednotlivé neurony se umísťují do vrstev a vrstvy pak tvoří neuronovou síť. Způsob zapojení neuronů, počet neuronů, počet vrstev a stejně tak volba agregační a aktivační funkce závisí na typu problému, pro který má být neuronová síť použita.

¹Zdroj: [2]



Obrázek 20: Model formálního neuronu ¹

3.1.1 Vstupy neuronu

Každý neuron může mít jeden nebo více vstupů. I když vstupy nemusí být nutně číselné, my to v této práci budeme předpokládat. Vstupy mohou být dvojího druhu:

- Výstup jiného neuronu,
- Vstup z vnějšího prostředí (např. od uživatele).

Všechny vstupy neuronu můžeme reprezentovat vektorem $X = [x_1, x_2, \dots, x_R]$. Hodnoty v X pak mohou být v:

- **Kvantitativní** formě, kdy vstup vyjadřuje *ano* nebo *ne* a nabývá tak binárních hodnot (např. 0 nebo 1),
- **Kvalitativní** formě, kdy vstup vyjadřuje konkrétní hodnotu nějaké veličiny, často normalizované na interval $[0; 1]$ nebo $[-1; 1]$.

3.1.2 Váhy spojení

Každý vstup neuronu x_i je ohodnocen číselnou vahou w_i . Čím vyšší váha vstupu je, tím citlivěji bude výstup neuronu reagovat, pokud se daný vstup změní (tím „důležitější“ vstup bude). Váhy jsou z počátku většinou pouze náhodné hodnoty a teprv během procesu učení neuronové sítě dochází k jejich úpravě tak, aby síť byla schopna řešit předložený problém co nejlépe. Vztah vstupu a jeho váhy se obecně označuje operátorem konfluence. [2]

$$z_i = x_i \oplus w_i \quad (8)$$

Vzorec 8: Obecný vztah vstupu neuronu a jeho váhy

Formální neuron pak využívá lineární hodnotu tohoto operátoru, který je tak možné nahradit prostým součinem. [2]

¹Převzato z: [2].

$$z_i = x_i w_i \quad (9)$$

Vzorec 9: Lineární vztah vstupu neuronu a jeho váhy

3.1.3 Práh neuronu

Prahem umělého neuronu w_0 se rozumí bariéra vstupu do neuronu z vnějšího okolí (nikoliv z jiného neuronu). Vedle váhy je to další parametr, který se mění při procesu učení sítě. [2]

3.1.4 Agregáčn  funkce

Proto  neuron m je m t v ce vstup , ale v dy pouze jedin  v stup, je t eba v echny vstupy n jak m zp sobem slou it do jedn  hodnoty. Pr v  toho se sna j  doc lit agrega n  funkce neuronu. V p p d  form ln ho neuronu se nej ast ji pou j v  prost  suma sou in  vstup  s v hami. V stupem agrega n  funkce je potenci l y_a . [2]

$$y_a = \sum_{i=0}^R z_i = \sum_{i=0}^R w_i \oplus x_i = \sum_{i=0}^R w_i x_i \quad (10)$$

Vzorec 10: Agrega n  funkce neuronu

3.1.5 Aktiva n  funkce

Je t e n   je potenci l y_a p iveden na v stup neuronu, je na n   aplikov na aktiva n  funkce. Tato funkce je z visl  na typu ře en ho probl mu a stejn  tak i na poloze neuronu v s ti. Je b  n ,  e pro v stupn  neurony se pou j v j  jin  aktiva n  funkce, n   pro neurony ve skryt ch vrstv ch. Funkce mohou b t line rn , neline rn , spojite i diskretn . [2]

$$y = \tanh y_a \quad (11)$$

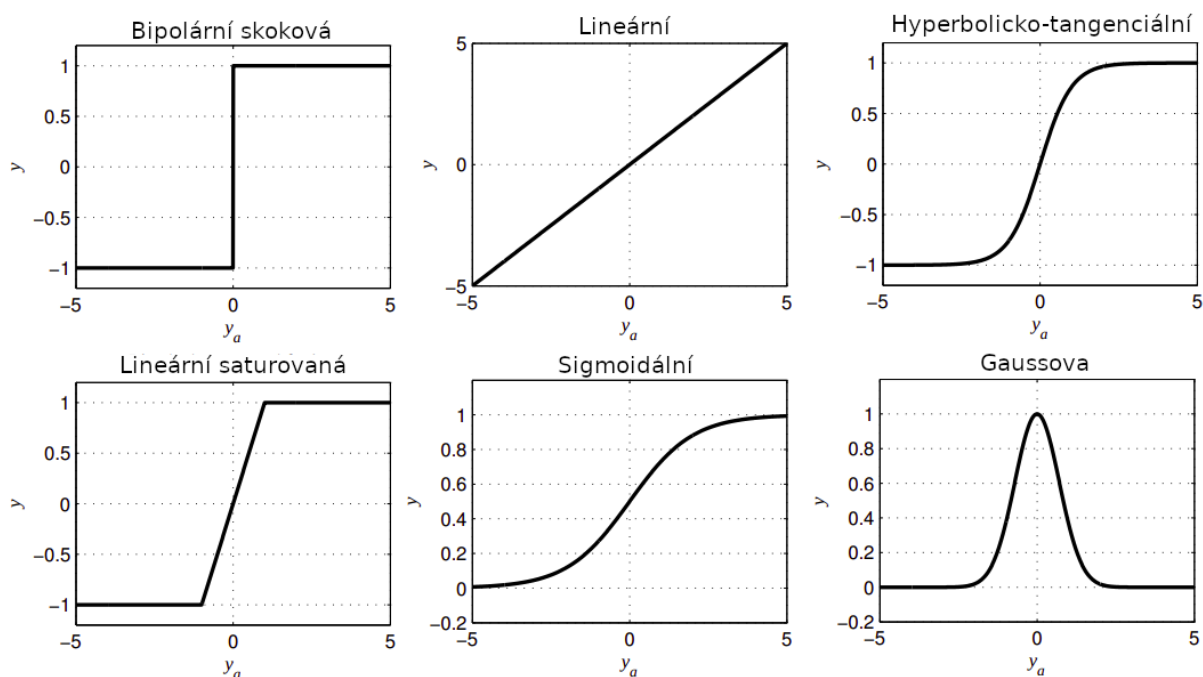
Vzorec 11: Hyperbolicko-tangenci ln  aktiva n  funkce

$$y = \frac{e^x}{1 + e^x} \quad (12)$$

Vzorec 12: Sigmoid ln  aktiva n  funkce

$$y = e^{-y_a^2} \quad (13)$$

Vzorec 13: Gaussova aktiva n  funkce



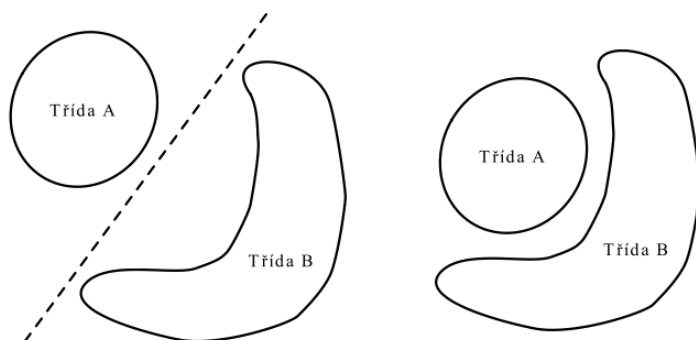
Obrázek 21: Příklady aktivačních funkcí neuronu ¹

3.2 Typy neuronových sítí

V této kapitole budou zmíněny některé typy neuronových sítí. Uvedený výčet však v žádném případě není kompletní. Vedle zmíněných typů existují např. rekurentní neuronové sítě nebo samoorganizující se neuronové sítě.

3.2.1 Jednoduchý perceptron

Jednoduchý perceptron je neuronová síť tvořená jediným neuronem. Jedná se o neuronovou síť s lineárně váženou agregační funkcí, učení s učitelem a skalárním výstupem nabývajícím binárních (1, 0) nebo bipolárních (1, -1) hodnot. [2]



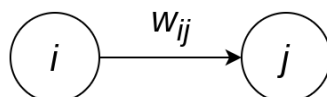
Obrázek 22: Lineárně separovatelná (vlevo) a neseparovatelná (vpravo) úloha ¹

Tento typ neuronové sítě lze použít pouze pro řešení úloh, které jsou lineárně separovatelné. Pro složitější úlohy je třeba využít vícevrstvý perceptron. [2]

¹Převzato z: [2].

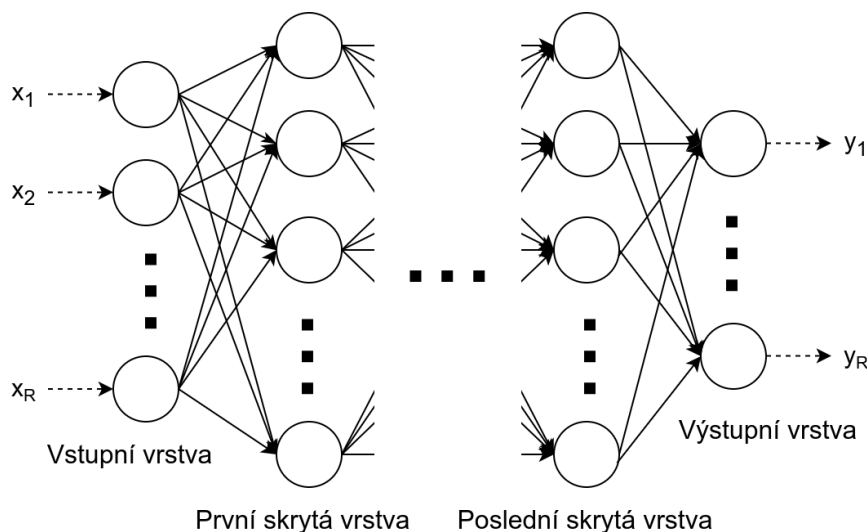
3.2.2 Dopředná vícevrstvá umělá neuronová síť

Dopředná vícevrstvá umělá neuronová síť (dále jen FFNN, někdy také vícevrstvý perceptron) je díky své univerzálnosti jedním z nejpoužívanějších typů sítí. Skládá se z neuronů, které jsou umístěny do jednotlivých vrstev. Existuje zde minimálně vrstva vstupní a vrstva výstupní. Dle povahy řešeného problému zde ale může být i libovolný počet dalších, tzv. skrytých vrstev. [2]



Obrázek 23: Zapojení neuronů ¹

Spojení existují pouze mezi neurony sousedních vrstev. Ani neurony stejné vrstvy, ani neurony nesousedících vrstev nejsou přímo propojeny. Orientace těchto spojů je navíc směrem pouze od vstupů k výstupu sítě (proto dopředná) – výstupy neuronů v předešlé vrstvě jsou vstupem do všech neuronů v následující vrstvě. [2]



Obrázek 24: Architektura dopředné vícevrstvé umělé neuronové sítě ¹

Pro vytváření architektury neuronové sítě neexistuje analytický postup, který by vedl k nejefektivnějšímu návrhu. Při řešení úlohy je třeba buď inspirovat se již nějakou existující neuronovou sítí, s jejíž pomocí někdo podobný problém řešil v minulosti, nebo postupovat experimentálně a „zkoušet“, jaká architektura povede k nejmenší chybě. V tomto případě je možno postupovat dvěma způsoby. Buď začneme s triviální sítí, do které postupně budeme přidávat nové neurony a vrstvy, až dokud bude úspěšnost sítě při řešení úkolu

¹Vytvořeno autorem v <https://www.draw.io>.

stoupat, nebo naopak začneme s komplexní neuronovou sítí, ze které budeme neurony a vrstvy postupně odebírat. [2]

3.2.3 Konvoluční neuronová síť

Zpracování vícedimenzionálních dat o velkých rozměrech by s použitím klasické FFNN bylo velice problematické. Uvažujme např. rozpoznávání psaných číslic z obrázku. Číslice se v obrázku může nacházet na různých místech, může být různě velká, natočená či barevná a také způsob psaní se u každého člověka může lišit.

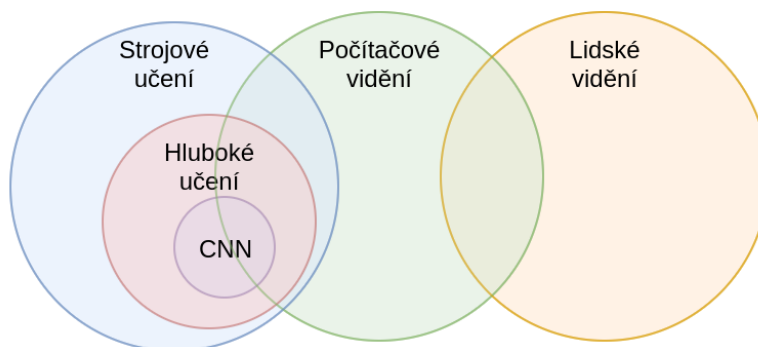


Obrázek 25: Ručně psané číslice 9. ¹

Pro člověka je jednoduché rozpoznat, že všechny útvary na obrázku 25 jsou číslice 9 i přesto, že každá vypadá jinak. Sestrojit algoritmus, který by to dokázal také, už ale triviální úkol rozhodně není. Obecně, získáváním informací z obrázku se zabývá obor počítačového vidění. [3]

Zásadní myšlenkou počítačového vidění je nesnažit se rozpoznávat celý objekt, ale nejdříve extrahovat fragmenty (vlastnosti), ze kterých se daný objekt skládá. U číslice 9 by těmito vlastnostmi mohly být např. elipsa a z ní vycházející křivka či úsečka. K extrakci vlastností z obrázku lze přistupovat dvěma způsoby založenými na:

- Manuálním inženýrstvím (např. histogram orientovaných gradientů),
- Strojovém učení (např. konvoluční neuronová síť). [3]



Obrázek 26: Vztah mezi počítačovým viděním, strojovým učení a konvoluční sítí ²

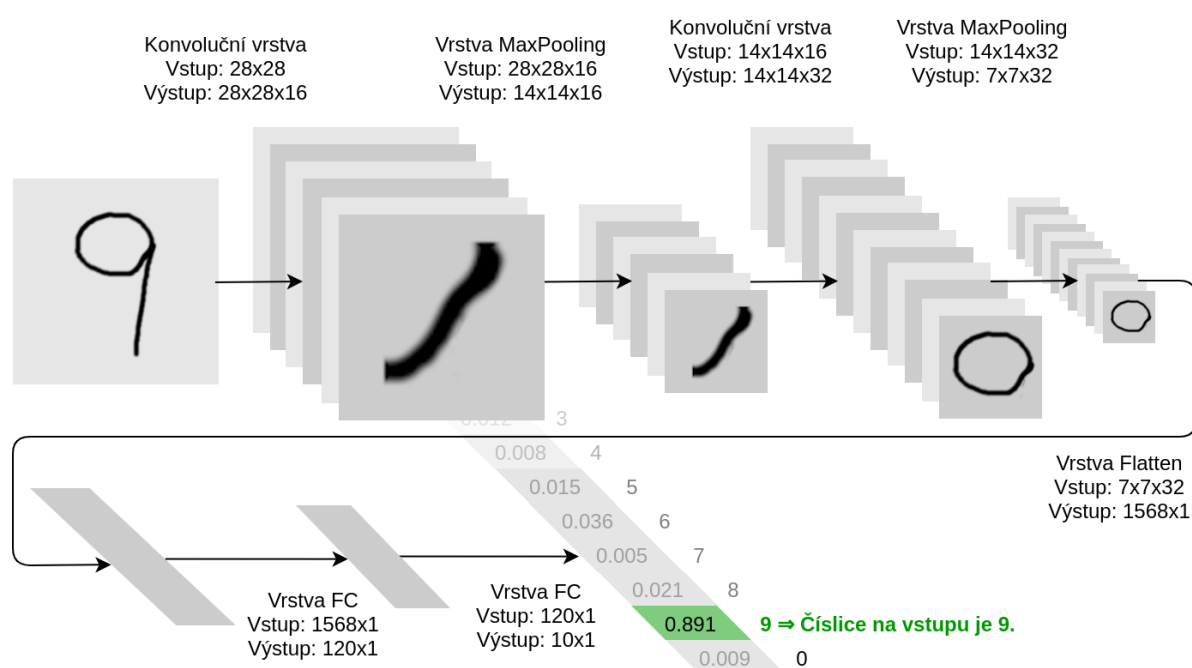
My se zde budeme zabývat pouze konvoluční neuronovou sítí (CNN). Tu můžeme chápat jako klasickou FFNN, jejíž vstupy jsou ovšem nejdříve modifikovány tak, aby

¹Převzato z: MNIST dataset.

²Vytvořeno autorem v <https://www.draw.io>.

z nich byly vyextrahovány výše zmíněné vlastnosti. Tato extrakce je prováděna pomocí konvoluční vrstvy, jejíž způsob fungování je detailněji popsán v kapitole 4.3.2.

Dále je třeba brát v úvahu velikost obrázku. Full HD obrázek ($1\,920 \times 1\,080$) se skládá z více jak 2 milionů pixelů. V případě RGB obrázku se množství informací zvýší až na 6 milionů (3 složky pro každý pixel). Posílat takové množství hodnot do FFNN, která se skládá z plně propojených vrstev, kde jsou neurony v sousedních vrstvách propojeny způsobem „každý z každým“, by bylo problematické. Z tohoto důvodu existuje tzv. pooling vrstva popsaná v kapitole 4.3.3, která umožňuje zmenšit velikost vstupu a přitom zachovat vzory charakteristické pro vstup.



Obrázek 27: Příklad konvoluční sítě pro rozpoznávání ručně psaných číslic ¹

Nutno podotknout, že použití CNN se neomezuje pouze na dvourozměrné obrázky. Stejně vhodné jsou i na rozpoznávání vzorů v 1D útvarech (např. světelná křivka hvězdy nebo jiné druhy časových řad) nebo i ve vícerozměrných strukturách. [3]

¹Vytvořeno autorem v <https://www.draw.io> a GIMP.

3.3 Typy vrstev neuronových sítí

3.3.1 Plně propojená vrstva

Plně propojená vrstva (FC, Fully connected nebo Dense layer) je vrstva, pro jejíž každý neuron platí, že na jeho vstup jsou přivedeny výstupy všech neuronů v předchozí vrstvě. Neurony sousedních vrstev FC jsou tak mezi sebou propojeny stylem „každý s každým“.

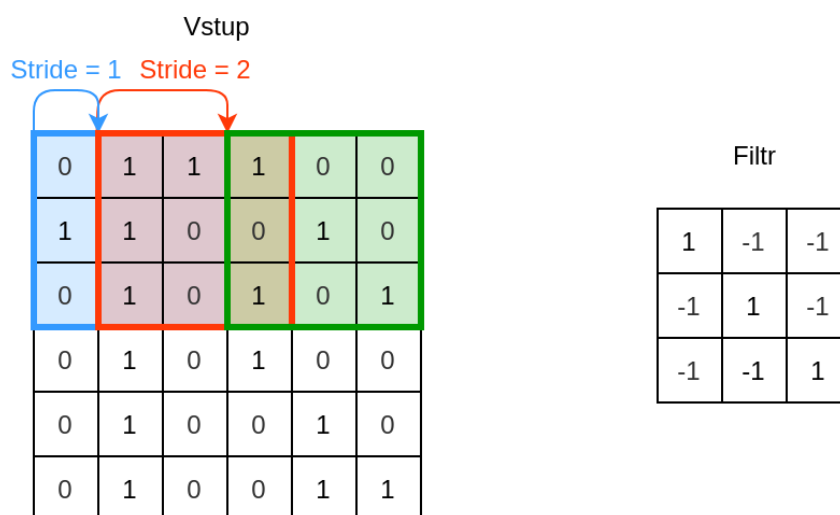
Parametr	Popis
Počet neuronů	Je určen složitostí a typem řešeného problému a určuje velikost výstupu vrstvy.

Tabulka 5: Parametry plně propojené vrstvy ¹

Počet neuronů se volí dle řešeného problému. Pokud bude počet neuronů příliš nízký, neuronová síť nemusí mít kapacitu pro to, aby byla schopna se naučit řešit daný problém. Pokud bude počet neuronů příliš vysoký, síť se až příliš dobře naučí řešit problém na trénovací množině a nebude schopna generalizace (snadno se tzv. přetrénuje).

3.3.2 Konvoluční vrstva

Konvoluční vrstva sestává z filtrů, které jsou aplikovány na vstup a provádí operaci „konvoluce“. Filtrem můžeme chápat vzor, jenž je vyhledáván ve vstupu a který zastupuje nějakou vlastnost vstupního objektu (např. obrázku).



Obrázek 28: Postupné posouvání konvolučního filtru přes vstup ²

¹Zdroj: [15]

²Vytvořeno autorem v <https://www.draw.io>.

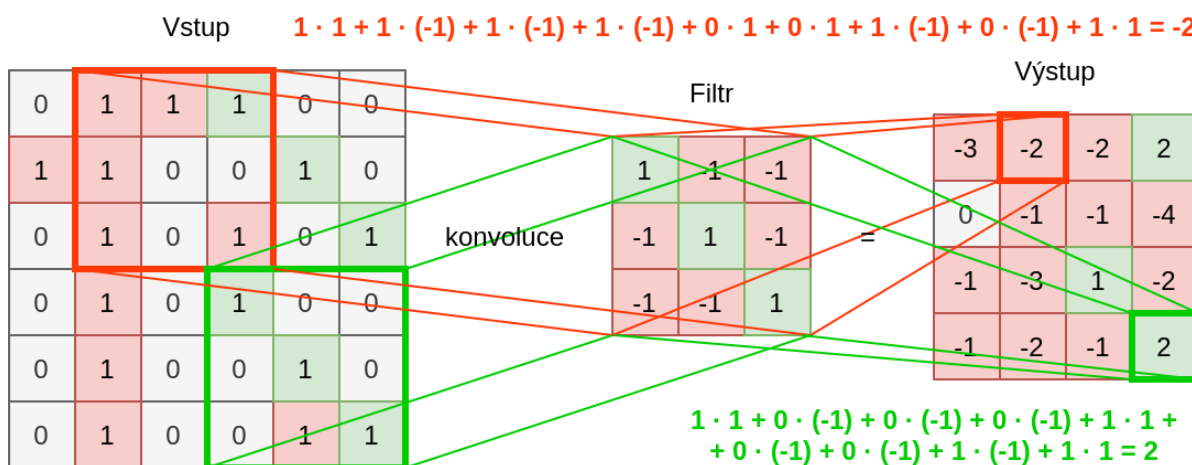
Pro výřez ze vstupu a filtr v konvoluční vrstvě se hodnoty na stejných pozicích ve výřezu i filtru vynásobí a tyto součiny se následně sečtou.

$$y = \sum_{i=1}^{sy} \sum_{j=1}^{sx} v_{ij} f_{ij} \quad (14)$$

Vzorec 14: Provádění konvoluce pro výřez vstupu a filtr

y = výstup $|$ sx, sy = horizontální/vertikální velikost filtru $|$ v = výřez vstupu $|$ f = filtr

Výřez se postupně posouvá (*stride*) v rámci vstupu a pro každou kombinaci výřezu a daného filtru se výsledek uloží do výstupního tenzoru na odpovídající místo vzhledem ke vstupu. Výsledkem je tak tenzor (*feature map*) stejně velký nebo menší, než vstup – v závislosti na vyplnění/oříznutí okrajů (*padding*).



Obrázek 29: Příklad aplikace konvolučního filtru na vstup ¹

Na obrázku 29 je znázorněno rozpoznávání diagonály. Ve výstupu jsou v oblastech, kde jsou ve vstupu diagonály (v pravém horním i dolním rohu), vyšší číselné hodnoty než tam, kde ve vstupu diagonály nejsou. Celý tento postup se opakuje pro všechny filtry, jimiž konvoluční vrstva disponuje. Výstupem vrstvy je tak jedna nebo více *feature maps*. [3]

Parametr	Popis
Počet filtrů	Čím více filtrů, tím více vlastností ve vstupu lze vyhledávat.
Velikost filtrů (kernel)	Čím větší filtr, tím složitější vzory lze ve vstupu vyhledávat. Filtr může mít v každém svém rozměru jinou velikost (v případě 2D to nemusí být čtverec, ale i obdélník).
Posun filtrů (stride)	Posunutí filtru v každé fázi konvoluce.
Okraj (padding)	Způsob zpracování okrajů vstupu, které se ve výstupu buď useknou, nebo vyplní nějakou hodnotou.

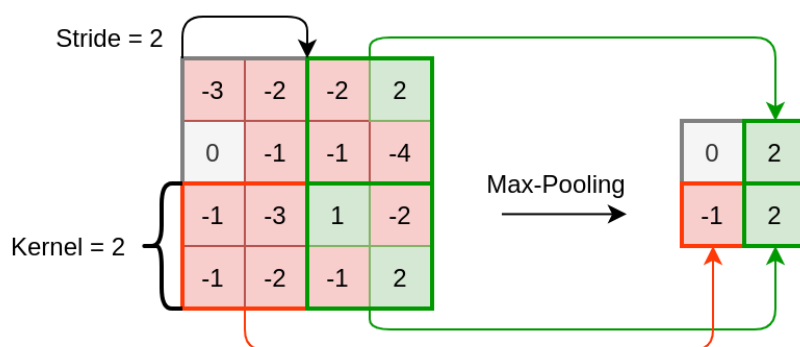
Tabulka 6: Parametry konvoluční vrstvy ²

¹Vytvořeno autorem v <https://www.draw.io>.

²Zdroj: [15]

3.3.3 Vrstva Max-Pooling

Vrstva Max-Pooling si klade za cíl zmenšit velikost vstupu a zároveň zachovat co nejvíce důležitých informací, které se v něm nachází. Podobně jako v konvoluční vrstvě i zde se posouvá okno o určité velikosti (kernel) o určitý počet polí (stride). Na výstup se však dostane maximální hodnota, která je v aktuálním výřezu vstupu. [3, 15]



Obrázek 30: Příklad operace Max-Pooling ¹

Na obrázku 30 je ukázána aplikace operace Max-Pooling na vstup, který byl výstupem v minulé kapitole – *feature map* ukazující umístění diagonál v původním obrázku. Jak je vidět, velikost struktury se zmenšila na čtvrtinu, a přesto v ní zůstaly všechny důležité informace. V pravém dolním i horním rohu máme stále vysoké číselné hodnoty indikující přítomnost diagonály.

Parametr	Popis
Velikost okna (kernel)	Velikost okna, ze kterého se vybírá maximum.
Stride okna (stride)	Posunutí okna v každé fázi operace Max-Pooling.
Okraj (padding)	Způsob zpracování okrajů vstupu, které se ve výstupu buď useknou nebo vyplní nějakou hodnotou.

Tabulka 7: Parametry vrstvy Max-Pooling ²

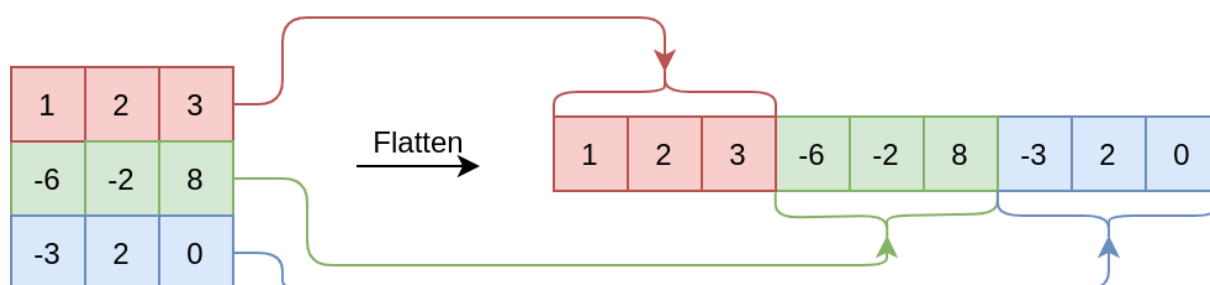
Obdobně k vrstvě Max-Pooling existuje např. i Min-Pooling nebo Average-Pooling. Jejich fungování je stejné s tím rozdílem, že na výstup předávají místo maximální hodnoty tu minimální, resp. průměrnou. Jejich použití je opět závislé na typu řešeného problému (např. zda vstupní obrázek má světlé pozadí a tmavé popředí či naopak). [15]

¹Vytvořeno autorem v <https://www.draw.io>.

²Zdroj: [15]

3.3.4 Vrstva Flatten

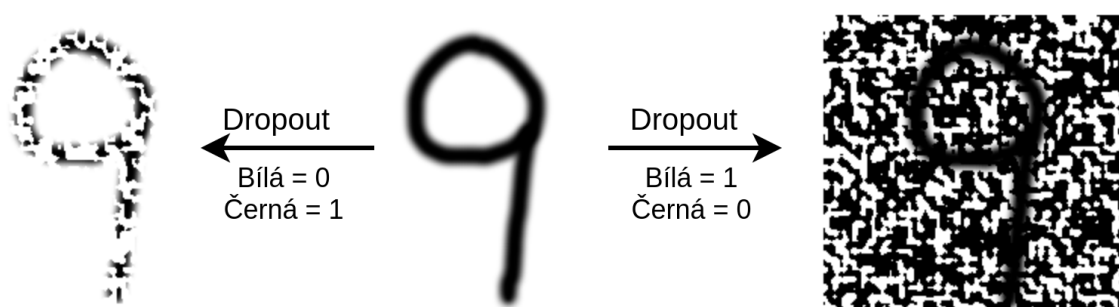
Konvoluční vrstvy umí pracovat s vícerozměrnými vstupy, vrstvy typu FC však očekávají pouze jednorozměrný vektor vstupních hodnot. Protože např. v konvolučních sítích je třeba oba typy vrstev kombinovat, je nutné nějakým způsobem převést n-rozměrný tenzor na vektor. Právě k tomu slouží vrstva Flatten. [15]



Obrázek 31: Příklad fungování vrstvy Flatten ¹

3.3.5 Vrstva Dropout

Pro snížení pravděpodobnosti, že se neuronová síť přetrénuje, je možné využít vrstvu Dropout. Ta při procesu trénování zahazuje vstupy (na výstup místo nich posílá hodnotu 0). Tím v trénovací množině vniká náhodný šum, což způsobí, že síť nebude trénovat pouze na stále stejných vstupech, ale trénovací množina bude vždy nepatrně odlišná – různorodější. [15]



Obrázek 32: Příklad fungování vrstvy Dropout ¹

Parametr	Popis
Rate	Frekvence zahazování vstupů.

Tabulka 8: Parametry vrstvy Dropout ²

¹Vytvořeno autorem v <https://www.draw.io> a GIMP.

²Zdroj: [15]

3.4 Učení umělé neuronové sítě

Vytvořená neuronová síť dle předchozích kapitol nebude umět řešit žádný problém – je to pouze množina neuronů a dalších parametrů poskládaných k sobě. Je potřeba ji daný problém nejdříve naučit řešit. Učení je proces, při kterém síť upravuje nastavitelné parametry (např. váhy vstupů neuronů) za účelem zvýšení úspěšnosti řešení předkládané úlohy. Učení může být:

- **S učitelem** – Síti je předložen vstup z trénovací množiny a ta pro tento vstup stanoví odezvu (výstup), který je porovnán s požadovaným výstupem. Na základě chyby mezi skutečným a požadovaným výstupem pak síť za využití tzv. učícího algoritmu upraví váhy mezi neurony tak, aby minimalizovala chybovou funkci.
- **Bez učitele** – Neuronová síť na základě schopnosti rozeznávat ve svých vstupech podobné vlastnosti a podle těchto vlastností třídit vstupy dokáže řešit určité problémy i bez znalosti požadovaných výstupů. [2]

3.4.1 Algoritmus zpětného šíření chyby

Algoritmus zpětného šíření chyby je učící algoritmus určený pro FFNN s diferencovatelnými spojitými aktivačními funkcemi. Neuronové síti je předložen vzor, pro který síť stanoví odezvu. Chyba pro jeden vzor je definována dle vzorce 15. [2]

$$E = \sum_{i=1}^Q (t_i - y_i)^2 = \sum_{i=1}^Q e_i^2 \quad (15)$$

Vzorec 15: Výpočet chyby pro jeden vzor u algoritmu zpětného šíření chyby

$$\begin{array}{l|l} E = \text{chyba jednoho vzoru} & y_i = \text{Skutečná hodnota } i\text{-tého výstupu} \\ Q = \text{počet výstupů NN} & t_i = \text{Očekávaná hodnota } i\text{-tého výstupu} \end{array} \quad \left| \begin{array}{l} e_i = \text{Chyba } i\text{-tého výstupu} \end{array} \right.$$

Následně je ke každé váze připočten přírůstek Δw_{ij}^k (při online učení – rychlejší) nebo se přírůstek kumuluje a přičte se až na konci epochy (při offline učení – stabilnější). [2]

$$\Delta w_{ij}^k = \frac{\partial E}{\partial w_{ij}^k} \implies \Delta w_{ij}^k = \alpha \delta_j^k y_i^{k-1} \quad (16)$$

Vzorec 16: Přírůstek váhy u algoritmu zpětného šíření chyby

$$\begin{array}{l|l} w_{ij}^k = \text{Váha spoje mezi } i\text{-tým a } j\text{-tým neuronem v } k\text{-té vrstvě} & \alpha = \text{rychlost učení} \\ \delta_j^k = \text{lokální gradient neuronu aktualizované váhy} & E = \text{chyba jednoho vzoru} \\ y_j^{k-1} = \text{výstup } j\text{-tého neuronu v } k-1. \text{ vrstvě} & \end{array}$$

Přičemž lokální gradient neuronu je vypočítán dle vzorce 17.

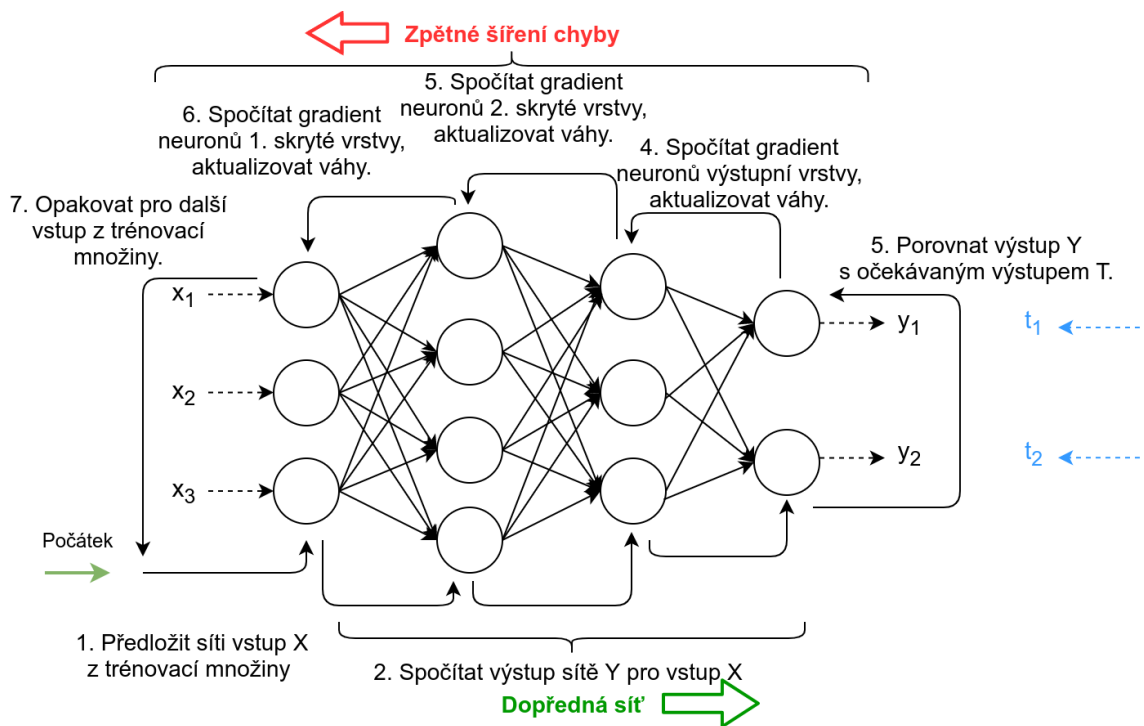
$$\delta_j^k = \begin{cases} \phi^k(y_{aj}^k) \sum_l \delta_l^{k+1} w_{jl}^{k+1}, & \text{pro neurony ve skrytých vrstvách} \\ e_j \phi^k(y_{aj}^k), & \text{pro neurony ve výstupní vrstvě} \end{cases} \quad (17)$$

Vzorec 17: Výpočet lokálního gradientu neuronu

$$\begin{array}{l|l} \delta_l^{k+1} = \text{lokální gradient l-tého neuronu } k+1. \text{ vrstvy} & e_j = \text{chyba výstupu j-tého neuronu} \\ y_{aj}^k = \text{vstupní potenciál j-tého neuronu } k\text{-té vrstvy} & \phi^k = \text{derivace aktivační funkce } k\text{-té vrstvy} \\ w_{ij}^k = \text{váha spoje mezi i-tým a j-tým neuronem } k\text{-té vrstvy} & \end{array}$$

Tento postup je opakován pro všechny vzory v trénovací množině – této iteraci se říká *epocha*. Učení sestává z jedné nebo více epoch a může skončit:

- Po pevném počtu epoch,
- Po neurčitěm počtu epoch, až chyba učení na klesne na určitou hodnotu. [2]



Obrázek 33: Příklad algoritmu zpětného šíření chyby ¹

Počáteční hodnoty vah spojů mezi neurony mohou být nastaveny např. náhodně za použití rovnoměrného nebo normálního rozdělení s nulovou střední hodnotou. Rychlost učení α určuje rychlost, s jakou se váhy spojů mezi neurony upravují dle chyby při trénování. Obvykle nabývá hodnot jako 0,1, 0,01, 0,001, apod. [2]

Čím vyšší rychlost učení je, tím rychleji bude síť natrénovaná, ale tím pravděpodobněji trénování skončí v nějakém lokálním minimu chybové funkce a nebude dosaženo tak dobrého výsledku. [3]

¹Vytvořeno autorem v <https://www.draw.io>.

4 PROJEKTY PRO HLEDÁNÍ EXOPLANET

4.1 Planet Hunters

TODO

4.2 Astronet

TODO: [9]

5 POUŽITÉ TECHNOLOGIE

5.1 TypeScript

5.1.1 React

5.1.2 Styled Components

5.1.3 NPM

5.2 Python

5.2.1 Flask

5.2.2 Astropy

5.2.3 LightKurve

5.2.4 TensorFlow

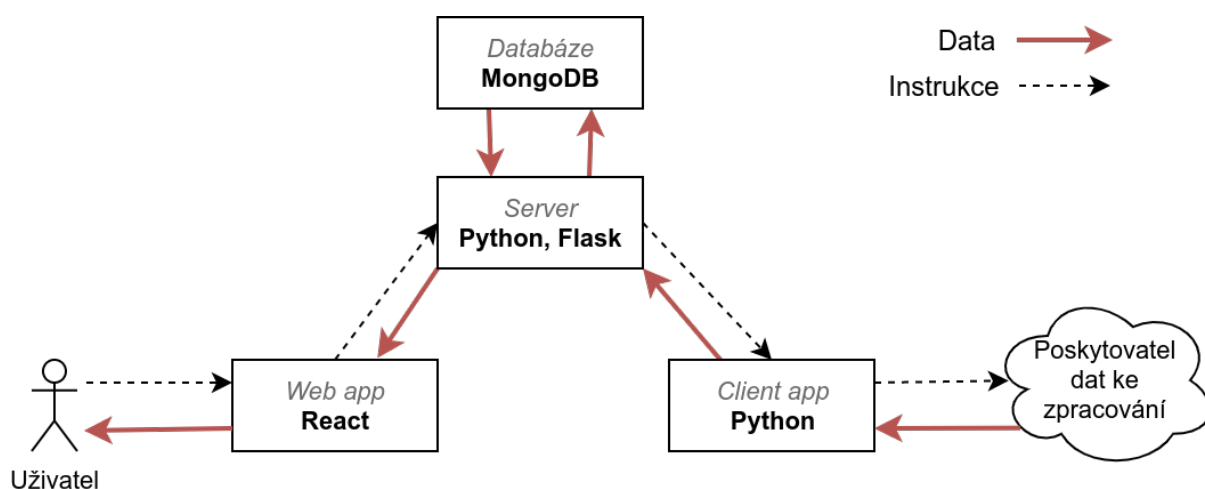
5.2.5 Pip

5.3 MongoDB

5.4 Socket.io

5.5 Git

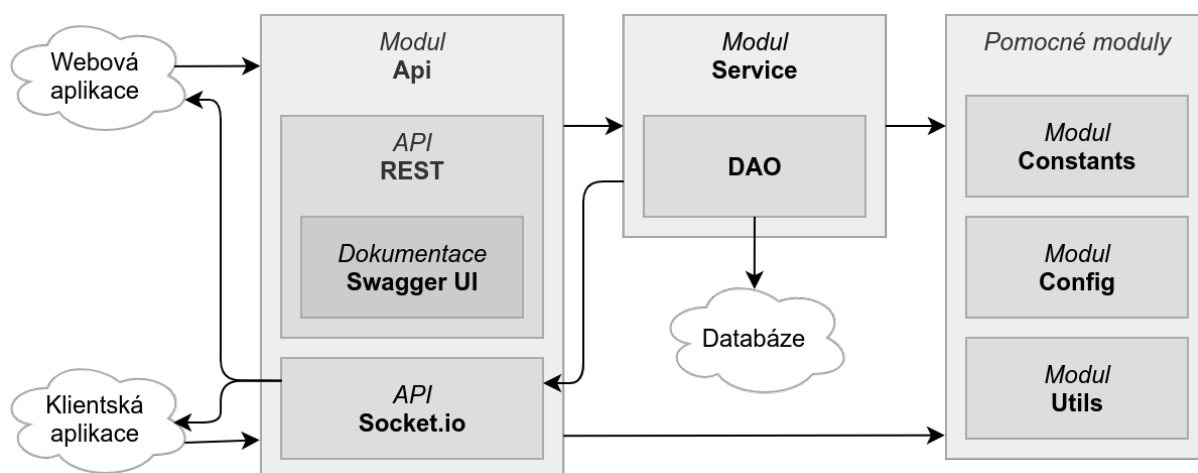
6 NÁVRH A VÝVOJ APLIKACE



Obrázek 34: Komponenty projektu a jejich komunikace ¹

6.1 Server

Úkolem serveru je přidělování výpočetních úloh připojeným klientským aplikacím, ukládání persistentních dat do databáze a komunikace s uživatelem skrze webovou aplikaci. Je naprogramován ve webovém frameworku Flask v jazyce Python a jeho adresářová struktura se dělí na následující moduly:



Obrázek 35: Architektura serveru ¹

- **api** – Definice REST a Socket API a zpracování příchozích požadavků. To zahrnuje autentizaci a autorizaci odesílatele a následné provedení operace v servisní vrstvě.

¹Vytvořeno autorem v <https://www.draw.io>.

- **config** – Konfigurace serveru (např. připojení k databázi). Základním konfiguračním souborem je `base.cfg`, který je použit automaticky. Bez nutnosti tento soubor měnit je možné vytvořit nový soubor (např. `test.cfg`) s novou konfigurací a pomocí argumentu při spuštění serveru nastavit, aby se použila tato konfigurace (`./src/main.py --env test`). [TODO]
- **constants** – Veškeré konstantní položky, ať už technického či fyzikálního typu.
- **service** – Servisní vrstva obsahující logiku serveru a manipulaci s databází.
- **utils** – Pomocné třídy uchovávající specifickou funkcionalitu.

6.1.1 REST API

K vytvoření rozhraní je nejdříve třeba nadefinovat příslušné struktury, které rozhraní bude očekávat nebo naopak vracet. Modul `fields` umožňuje určit složky jednotlivých struktur a taktéž pro ně nastavit pravidla jako např. maximální délka, datový typ nebo seznam povolených hodnot.

```
credentials = api.model("Credentials", {
    "email": fields.String(required=True, max_length=50),
    "password": fields.String(required=True, max_length=100)
})

user = api.model("User", {
    "name": fields.String(required=True, max_length=30),
    "role": fields.Integer(required=True, enum=UserRole.values())
})
```

Zdrojový kód 1: Vytvoření modelů v REST API.

Následně může dojít k nadefinování koncových bodů rozhraní. Koncovým bodem jsou metody jakékoliv třídy, jež je potomkem `Resource`, přičemž názvy metod třídy odpovídají názvům metod HTTP.

```
@api.route("/login")
class Login(Resource):

    @api.response(HttpStatus.OK, user)
    @api.response(HttpStatus.BAD_REQUEST, "Invalid credentials.")
    @api.expect(credentials)
```

```
def post(self, credentials):
    return user_service.authenticate(credentials)
```

Zdrojový kód 2: Vytvoření koncového bodu v REST API.

Veškeré vstupní parametry od uživatele jsou automaticky parsovány, při nesprávnosti vstupních parametrů je odeslána chybová odpověď a díky anotacím je také automaticky generována dokumentace REST API v nástroji Swagger UI dostupná na adrese `exoplanets.now.sh/api-docs`. Ukázka dokumentace REST API je umístěna v příloze [TODO] na konci dokumentu.

6.1.2 Socket.io API

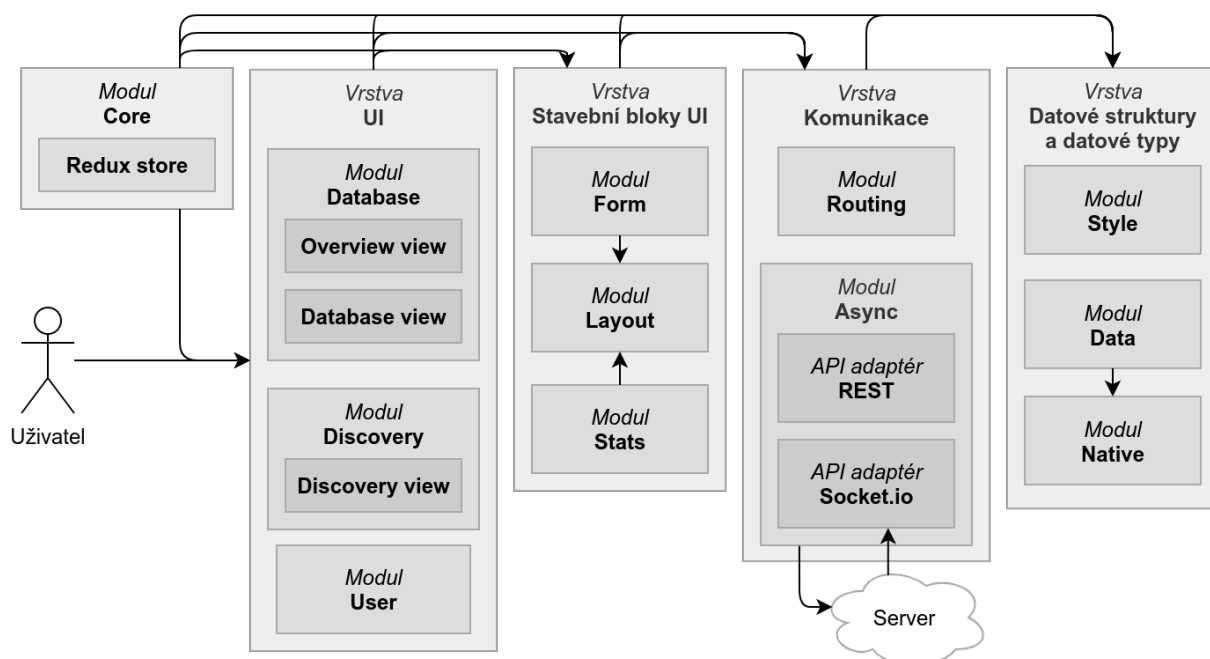
Protože je třeba v reálném čase synchronizovat webovou a klientskou aplikaci a umožnit serveru iniciovat s těmito aplikacemi komunikaci, server kromě rozhraní HTTP poskytuje také rozhraní Socket.io.

```
# Client:
@socketio.event
def process(task):
    print(f"Data processing from address {task['url']}...")

# Server:
task = {'type': "LIGHT_CURVE", "url": "http://exoplanetarchive.ipac"}
socketio.emit("process", task)
```

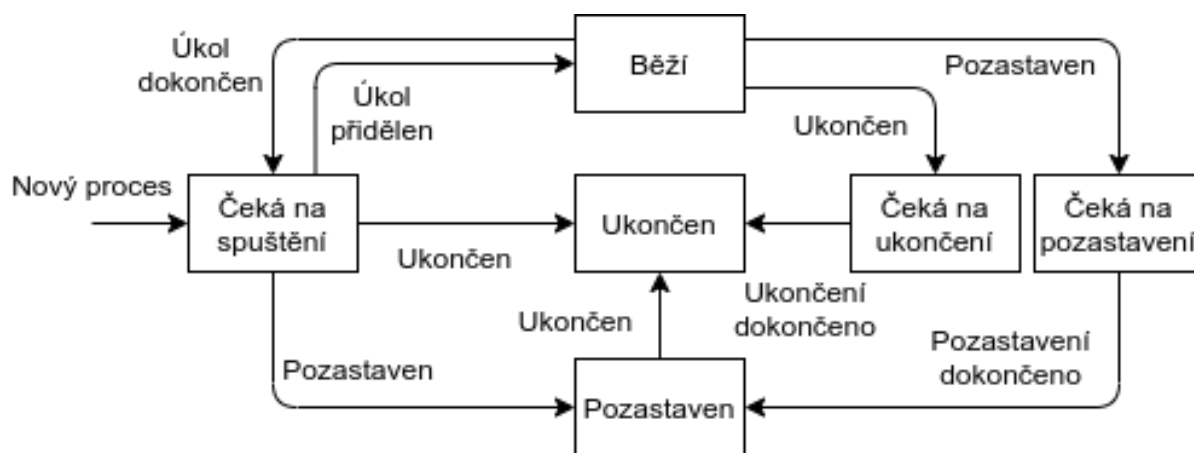
Zdrojový kód 3: Ukázka komunikace pomocí socket.io.

6.2 Webová aplikace



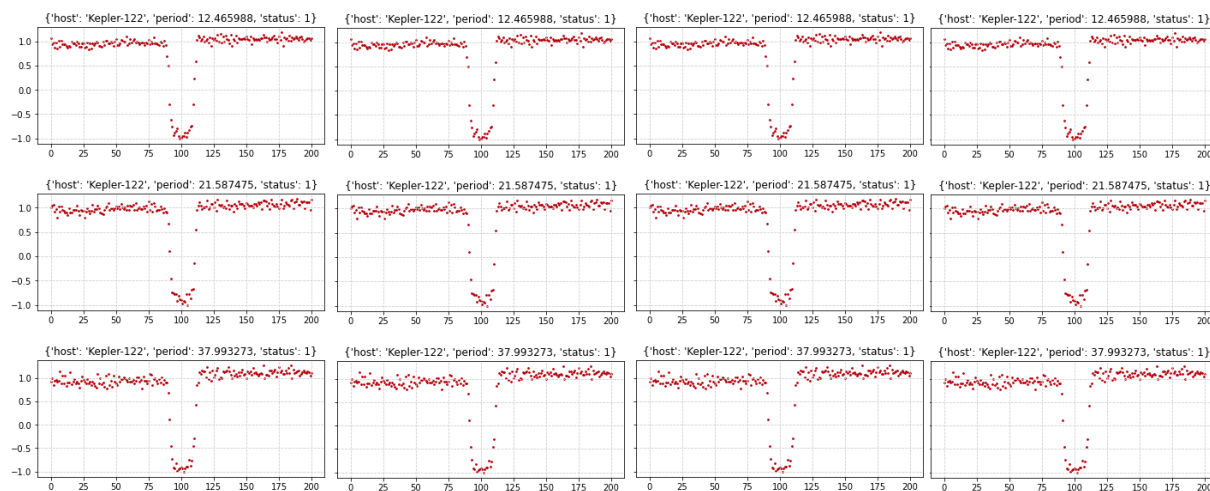
Obrázek 36: Architektura webové aplikace ¹

6.3 Klientská aplikace



Obrázek 37: Stavy procesů a přechody mezi nimi ¹
¹Vytvořeno autorem v <https://www.draw.io>.

6.4 Neuronová síť



Obrázek 38: Trénovací množina ¹

TODO: Odebrat osy a popisky z obrázku.

6.5 Databáze

6.6 Datasety

Většina dat se do databáze nevkládá manuálně, jelikož je třeba pracovat se stovkami tisíc či miliony datových položek. Namísto toho administrátor pouze definuje přístupový bod k nějakému datasetu dostupnému přes webové rozhraní a následně dojde k automatickému zpracování datasetu a všech položek v něm. Některé typy datasetů jsou určeny k jednorázovému uložení do databáze a není nutno je nijak dále zpracovávat. Zpracování jiných je naopak výpočetně náročné a připojené klientské aplikace tak činí postupně položku po položce. Jednotlivé typy datasetů jsou popsány níže.

U všech typů datasetů je také vyřešena situace, kdy by se jedna a ta samá položka (např. hvězda) nacházela ve dvou datasetech zároveň. V takovém případě dojde k uložení obou hodnot a uživatel v aplikaci pak bude vidět údaje dané položky z obou datasetů vedle sebe.

¹Vytvořeno autorem v <https://www.draw.io>.

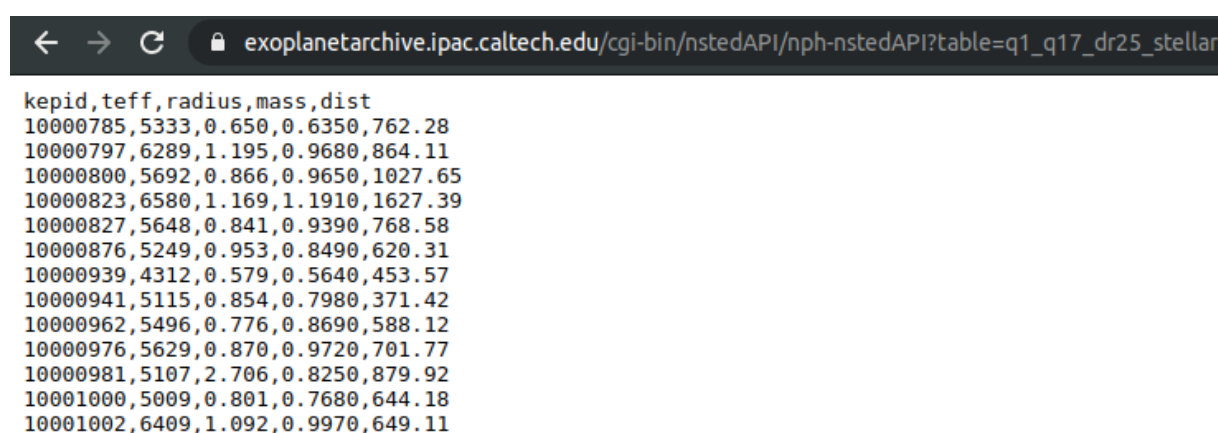
6.6.1 Target pixel files

6.6.2 Světelné křivky hvězd

6.6.3 Radiální rychlosti hvězd

6.6.4 Hvězdy

Informace o vlastnostech hvězd nejsou pro hledání exoplanet nutné, avšak pro spočítání dalších údajů o nalezené exoplanetě je třeba je zahrnout do výpočtů. Datasetsy tohoto typu jsou k dostání přes webová rozhraní nejčastěji ve formátu `csv`.



Obrázek 39: Část datasetu s vlastnostmi hvězd ¹

Tyto datasetsy neobnáší žádné další složité výpočty, pouze jsou společně s dopočítanými údaji jednorázově uloženy do databáze. Základními údaji, které jsou pro každou hvězdu třeba, jsou (šedou barvou jsou dopočítané údaje):

Název	Zdánlivá magnituda	Rovníkový průměr	Spektrální typ
Vzdálenost od Země	Hmotnost	Absolutní magnituda	Povrchová gravitace
Metalicita	Povrchová teplota	Průměrná hustota	Obyvatelná zóna

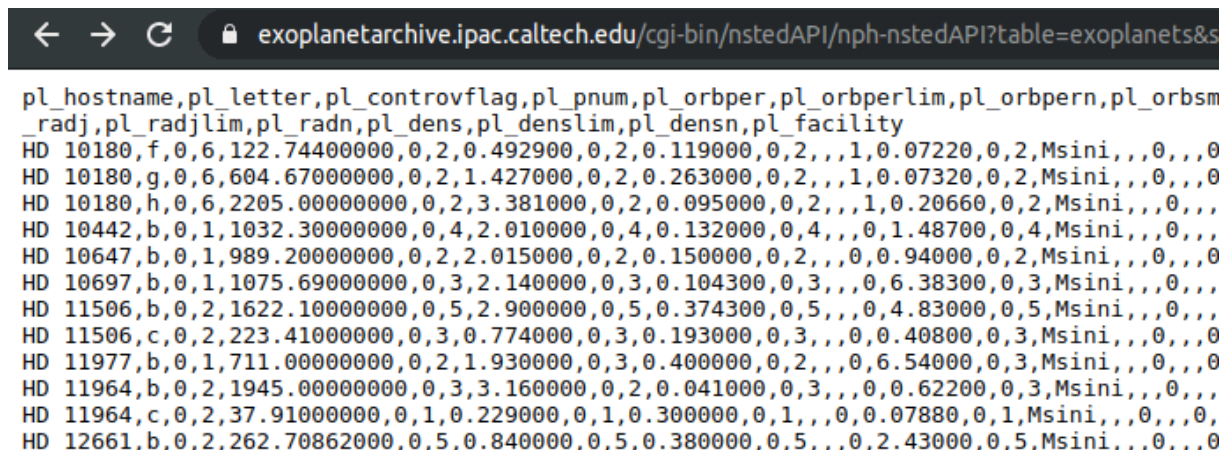
Tabulka 9: Údaje o hvězdách ukládané do databáze

Aplikace je postavena flexibilně a umožňuje libovolně namapovat sloupce z datasetů do sloupců v databázi. Datasetsy tak mohou mít pořadí i názvy jednotlivých sloupců libovolné.

¹Dataset pochází ze stránek <https://exoplanetarchive.ipac.caltech.edu>.

6.6.5 Planety

Datasety s planetami nejsou pro běh aplikace taktéž nutné, protože veškeré informace o exoplanetách jsou vypočítány z jiných datasetů. Platí však, že čím více nezávislých zdrojů se na údajích o exoplanetě shodne, tím spíše budou tyto údaje platné. Proto je umožněno ukládat do databáze i datasety s údaji o planetách – mohou potvrdit nebo vyvrátit údaje vypočtené v rámci aplikace.



Obrázek 40: Část datasetu s vlastnostmi planet ¹

Opět se jedná o datasety nejčastěji ve formátu `csv`, které není nutno nijak složitě zpracovávat, pouze uložit do databáze.

Název	Povrchová teplota	Excentricita dráhy	Rovníkový průměr
Velká poloosa	Typ	Hmotnost	Perioda oběhu
Tranzit přes hvězdu	Průměrná hustota	Rychlost oběhu	Podmínky pro život

Tabulka 10: Údaje o planetách ukládané do databáze

6.6.6 Názvy

Často se stává, že jedno a to samé těleso je v různých datasetech pod různým označením.

K zamezení toho, aby byly tyto položky vedeny jako dvě různé soustavy je nutno aplikaci poskytnout informace o používaných názvech jednotlivých objektů. Právě k tomu slouží tento typ datasetů.

¹Zdój: <https://exoplanetarchive.ipac.caltech.edu>.

²Zdój dat: <http://simbad.u-strasbg.fr/simbad/sim-id?Ident=Kepler-10>.

Katalog	Účel	Označení Kepler-10
KIC (K epler I ntput C atalog)	Hledání exoplanet (Kepler)	KIC 11904151
KOI (K epler O bject of I nterest)	Výběr z KIC	KOI-72
Kepler	Potvrzené exoplanety z KOI	Kepler-10
2MASS (2 Micron A ll- S ky S urvey)	Infra. průzkum oblohy	2MASS J19024305+5014286
GSC (G uide S tar C atalog)	Pozorování hvězd (Hubble)	GSC 03549-00354
Gaia DR (G aia D ata R elease)	Měření polohy hvězd (Gaia)	Gaia DR2 2132155017099178624
USNO-B1.0	Pozorování hvězd a galaxií	USNO-B1.0 1402-00324696
UCAC3	Pozorování hvězd	UCAC3 281-142262

Tabulka 11: Pojmenování soustavy Kepler-10 v různých katalozích ²

7 ROZVRŽENÍ APLIKACE

7.1 Přehled

7.2 Databáze

7.3 Detail systému

Na jediné stránce jsou shrnuty všechny známé informace o systému. Kromě hodnot veličin hvězdy a případných planet jsou zde zaznamenána všechna pozorování, vizuální porovnání velikostí a vzdáleností oproti sluneční soustavě, seznam referencí a v poslední řadě také interaktivní 3D model systému.

7.4 Objevování

7.5 Náповěda

7.6 Autentizace

ZÁVĚR

TODO: Verifikace výsledků.

POUŽITÁ LITERATURA

- [1] DATTIO, Anne. Identifying Exoplanets with Deep Learning. II. Two New Super-Earths Uncovered by a Neural Network in K2 Data. *The Astronomical Journal* 157(5). [online]. 9. 4. 2019. [cit. 23. 10. 2020]. Dostupné z: <https://iopscience.iop.org/article/10.3847/1538-3881/ab0e12>
- [2] DOLEŽEL, Petr. Úvod do umělých neuronových sítí. *Univerzita Pardubice, Fakulta elektrotechniky a informatiky*. 2016. [cit. 9. 10. 2020]. ISBN 978-80-7560-022-6
- [3] KHAN, Salman. A Guide to Convolutional Neural Networks for Computer Vision. *Morgan & Claypool*. 2018. [cit. 26. 10. 2020]. ISBN 781681730226
- [4] LOVIS, Christophe, FISCHER, Debra A. Radial Velocity Techniques for Exoplanets. *University of Arizona Press*. [online]. 2011. [cit. 25. 12. 2019]. Dostupné z: https://www.researchgate.net/publication/253789798_Radial_Velocity_Techniques_for_Exoplanets
- [5] MARCY, Geoffrey. The planet around 51 Pegasi. *The astrophysical journal* 481(2). [online]. 1997. [cit. 29. 12. 2019]. Dostupné z: <https://iopscience.iop.org/article/10.1086/304088>
- [6] MOUTOU, Claire, PONT, Frédéric. Detection and characterization of extrasolar planets: the transit method. *Strasbourg: Observatoire astronomique de Strasbourg et Société Française d'Astronomie et d'Astrophysique*. [online]. 2006. [cit. 8. 10. 2020]. Dostupné z: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.125.4155&rep=rep1&type=pdf>
- [7] OGBUEFI, Calvin. Photometry Analysis of Exoplanets WASP-80b & HD 189733b. *Baylor University*. [online]. 2013. [cit. 23. 10. 2020]. Dostupné z: <https://www.baylor.edu/content/services/document.php/208057.pdf>
- [8] PERRYMAN, Michael. Extra-solar planets. *Reports on Progress in Physics* 63(8). [online]. 31. 5. 2000. [cit. 8. 10. 2020]. Dostupné z: <https://arxiv.org/abs/astro-ph/0005602>

- [9] SHALLUE, Christopher, VANDERBURG, Andrew. Identifying Exoplanets with Deep Learning: A Five-planet Resonant Chain around Kepler-80 and an Eighth Planet around Kepler-90. *The Astronomical Journal* 155(2). [online]. 30. 1. 2018. [cit. 8. 10. 2020]. Dostupné z: <https://iopscience.iop.org/article/10.3847/1538-3881/aa9e09>
- [10] TASKER, Elizabeth, LANEUVILLE, Matthieu, GUTTENBERG, Nicholas. Estimating Planetary Mass with Deep Learning. *The Astronomical Journal* 159(2). [online]. 25. 11. 2019. [cit. 9. 10. 2020]. Dostupné z: <https://arxiv.org/abs/1911.11035>
- [11] Metody objevování planet. *Astronomia*. [online]. 23. 1. 2013. [cit. 23. 12. 2019]. Dostupné z: <http://hvezdy.astro.cz/exoplanety/51-metody-objevovani-planet>
- [12] NASA Exoplanet Archive. *NASA Exoplanet Science Insititute*. [online]. 12. 8. 2019. [cit. 25. 12. 2019]. Dostupné z: https://exoplanetarchive.ipac.caltech.edu/docs/API_exoplanet_columns.html
- [13] Reference Solar Spectral Irradiance: ASTM G-173. *nrel*. [online]. ???. ?. ????. [cit. 27. 12. 2019]. Dostupné z: <https://rredc.nrel.gov/solar/spectra/am1.5/ASTMG173/ASTMG173.html>
- [14] TESS Exoplanet Mission. *NASA*. [online]. 24. 8. 2020. [cit. 8. 10. 2020]. Dostupné z: <https://www.nasa.gov/content/about-tess>
- [15] Keras API reference. *Keras*. [online]. 2020. [cit. 29. 10. 2020]. Dostupné z: <https://keras.io/api>

Exoplanets data http://exoplanets.org/detail/alpha_Cen_B_b

Wavelength <http://spiff.rit.edu/classes/phys240/lectures/expand/expand.html>

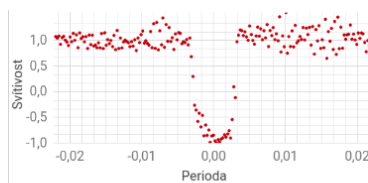
TODO: Doplnit reference ze zadání diplomové práce.

SEZNAM PŘÍLOH

Příloha A	Architektura neuronové sítě	61
-----------	-----------------------------------	----

PŘÍLOHA A – ARCHITEKTURA NEURON. SÍTĚ

Conv1D		<i>Input: 1001x1</i>
Filters: 16	Kernel: 3	<i>Output: 1001x1</i>
Stride: 1		Activation: ReLu

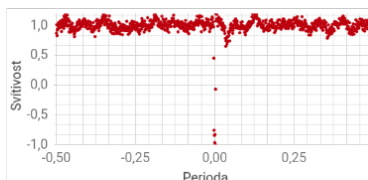


Conv1D		<i>Input: 1001x1</i>
Filters: 16	Kernel: 3	<i>Output: 1001x1</i>
Stride: 1		Activation: ReLu

MaxPool1D		<i>Input: 1001x1</i>
Pool: 2	Stride: 1	<i>Output: 999x16</i>

MaxPool1D		<i>Input: 1001x1</i>
Pool: 2	Stride: 1	<i>Output: 999x16</i>

Conv1D		<i>Input: 1001x1</i>
Filters: 16	Kernel: 3	<i>Output: 1001x1</i>
Stride: 1		Activation: ReLu



Conv1D		<i>Input: 1001x1</i>
Filters: 16	Kernel: 3	<i>Output: 1001x1</i>
Stride: 1		Activation: ReLu

MaxPool1D		<i>Input: 1001x1</i>
Pool: 2	Stride: 1	<i>Output: 999x16</i>

MaxPool1D		<i>Input: 1001x1</i>
Pool: 2	Stride: 1	<i>Output: 999x16</i>

Conv1D		<i>Input: 1001x1</i>
Filters: 16	Kernel: 3	<i>Output: 1001x1</i>
Stride: 1		Activation: ReLu

Conv1D		<i>Input: 1001x1</i>
Filters: 16	Kernel: 3	<i>Output: 1001x1</i>
Stride: 1		Activation: ReLu

MaxPool1D		<i>Input: 1001x1</i>
Pool: 2	Stride: 1	<i>Output: 999x16</i>

MaxPool1D		<i>Input: 1001x1</i>
Pool: 2	Stride: 1	<i>Output: 999x16</i>

Flatten	<i>Input: 1001x1</i>	
	<i>Output: 1001</i>	

Flatten	<i>Input: 1001x1</i>	
	<i>Output: 1001</i>	

Concatenate		<i>Input: 1001x1</i>
		<i>Output: 1001</i>

Dense		<i>Input: 1001x1</i>
Neurons: 16	Activ.: Tanh	<i>Output: 999x16</i>

Dropout		<i>Input: 1001x1</i>
		<i>Output: 1001</i>

Dense		<i>Input: 1001x1</i>
Neurons: 16	Activ.: Tanh	<i>Output: 999x16</i>

PŘÍLOHA B – ARCHITEKTURA DATABÁZE

