

DOKUMENTÁCIA

Rekonštrukcia top kvarkov

Tvorba informačných systémov 19/20

Richard Mészáros

Martin Slimák

Magdaléna Kozubaľová

Veronika Benková

Február 2020

Obsah

I KATALÓG POŽIADAVIEK

1. Úvod.....	6
1.1 Účel katalógu požiadaviek	6
1.2 Rozsah využitia systému	6
1.3 Slovník pojmov	6
1.4 Odkazy a referencie	7
1.5 Prehľad nasledujúcich kapitol	7
2. Všeobecný popis	8
2.1 Perspektíva systému	8
2.2 Funkcie systému	8
2.3 Charakteristika používateľa.....	8
2.4 Všeobecné obmedzenia	9
2.5 Predpoklady a závislosti	9
3. Špecifické požiadavky	10
3.1 Funkčné požiadavky	10
3.2 Kvalitatívne požiadavky	11
3.3 Požiadavky rozhrania	11

II NÁVRH

1. Úvod.....	13
1.1 Účel dokumentu	13
1.2 Prehľad nasledujúcich kapitol	13
2. Podrobná špecifikácia vonkajších interfejsov	13
3. Používané technológie	13
3.1 HTML/CSS	13
3.2 ROOT	13
3.3 KERAS	13
3.4 Tensorflow.....	13
3.5 Matplotlib	14

3.6 NumPy.....	14
3.7 LWTNN.....	14
4. Používateľské rozhranie	15
5. Návrh implementácie	16
5.1 UML – state diagram.....	16
5.2 UML – class diagram	17
5.3 UML – component diagram	18
6. Popis jednotlivých súborov	19
6.1 config.txt	19
6.2 Layers.py	20
6.3 Topq_DNN.py	20
6.4 Data.py	20
7. Formáty jednotlivých súborov	21
7.1 .root	21
7.2 .json	21
7.3 .h5	21
7.4 .png.....	21

III TESTOVACIE SCENÁRE

1. Prihlasovanie	23
2. Voľba premenných.....	23

IV PRÍRUČKA

1. Používateľ.....	25
1.1 Prihlásenie	25
1.2 Nastavenie parametrov	25
1.3 Spustenie tréningu.....	25
1.4 Ukladanie modelu.....	25
1.5 Vizualizácia	26

Časť I

Katalóg požiadaviek

1. Úvod

1.1 Účel katalógu požiadaviek

Tento dokument slúži na špecifikovanie a súhrn požiadaviek od zadávateľa pre aplikáciu rekonštrukcia Top Kvarkov. Bol vytvorený na základe komunikácie so zadávateľom projektu. Je určený pre zadávateľa, užívateľa a kohokoľvek, kto sa chce dozvedieť na čo slúži a kto so systémom bude pracovať. Účel tohto dokumentu je zabezpečenie správnosti a úplnosti dohodnutých požiadaviek od zadávateľa. Taktiež je dokument záväzný pre obe strany.

1.2 Rozsah využitia systému

Cieľom projektu je vytvorenie systému, ktorý bude pomocou neurónovej siete priradovať jety ku kvarku z ktorého vznikol. Ďalším cieľom je porovnanie výstupu z KL Fitter-u a neurónových sietí vo webovom rozhraní prostredníctvom dát z databázy. Webové rozhranie bude slúžiť na komunikáciu s databázou, serverom atlas23 a interakciou s používateľom.

1.3 Slovník pojmov

Kvark - podľa štandardného modelu časticovej fyziky elementárne častice, z ktorých sa skladajú hadróny (teda napríklad protóny a neutróny).

Top kvark – je kvark tretej generácie s elektrickým nábojom $+(2/3)e$. Je to najťažší zo všetkých známych elementárnych častíc.

Jet - úzky kužeľ hadrónov a iných častíc produkovaných hadronizáciou kvarku alebo gluónu v experimente s fyzikou častíc alebo experimentom s ťažkými iónmi.

KL Fitter – (Kinematic Likelihood Fitter) knižnica na kinematickú montáž pomocou pravdepodobnosti. Je primárne vyvinutý pre prípad rekonštrukcie top kvarku, ale dá sa ľahko upraviť tak, aby vyhovoval aj iným procesom.

Neurónová sieť – výpočtový model, zostavený na základe abstrakcie vlastností biologických nervových systémov.

Hlboká neurónová sieť – výpočtový model, zostavený na základe abstrakcie vlastností biologických nervových systémov s viacerými vrstvami.

Tensorflow – bezplatná a otvorená softvérová knižnica pre dataflow a diferencovateľné programovanie v rôznych úlohách. Je to matematická knižnica a používa sa tiež na aplikácie strojového učenia, ako sú neurónové siete.

KERAS – vysokoúrovňové API pre neurónové siete, schopné bežať na vrchole Tensorflow. Umožňuje rýchle experimentovanie prostredníctvom vysokoúrovňového, ľahko použiteľného, modulárneho a rozšíriteľného API.

ATLAS23 – Server

TTbar event - zrážka top kvakových párov

konfiguračný súbor – súbor obsahujúci všetky potrebné parametre na tréning a následne používanie neurónovej siete.

Procesorový čas - čas potrebný na vykonanie výpočtov na procesore.

1.4 Odkazy a referencie

1.4.1. ROOT Framework na analýzu dát: <https://root.cern.ch/>

1.4.2. Ako čítať strom z ROOT: <https://root.cern.ch/how/how-read-tree>

1.4.3. KL Fitter: <https://github.com/KLFitter/KLFitter>

1.4.4. LWTNN knižnica na prácu s neurónovými sieťami: <https://github.com/lwttn/lwttn>

1.5 Prehľad nasledujúcich kapitol

V nasledujúcich kapitolách sa čitateľ oboznámi s funkciami, perspektívami a obmedzeniami systému, taktiež so špecifickými požiadavkami. Tretia kapitola hovorí o funkčných a kvalitatívnych požiadavkách systému.

2. Všeobecný popis

Implementácia neurónovej siete a optimalizácia metódy KLFitter slúžiacej na rekonštrukciu top kvarkových párov.

2.1 Perspektíva systému

Systém bude ponúkať 2 možnosti výberu rekonštrukcie tt-bar eventov :

1. hlboké neurónové siete
2. KLFitter

Tieto možnosti si bude užívateľ môcť vybrať vo webovom rozhraní. Vo webovom rozhraní bude možné spustiť tréning neurónovej siete, pre ktorú sa budú môcť vo webovom rozhraní nastaviť parametre. Systém bude využívať aj databázu, ktorá bude uložená na serveri ATLAS23. Po spracovaní dát metódami na rekonštrukciu top kvarkov sa výsledky z týchto metód uložia do databázy, aby sme ich mohli porovnávať s nastaveniami, ktoré sme na začiatku zvolili. Výstupom budú dáta v grafickej a textovej podobe, ktoré budú reprezentovať účinnosti použitých metód.

2.2 Funkcie systému

Systém bude ponúkať možnosť vybrať si KLFitter alebo neurónovú sieť na rekonštrukciu top kvarkov. Vstupom bude súbor typu ROOT , ktorý sa prekonvertuje pomocou nástroja na prácu (so súbormi typu ROOT) na vhodný dátový typ, s ktorým vie systém pracovať. V prípade, že si používateľ vyberie neurónovú sieť, systém vytvorí skonvertovaný konfiguračný súbor , ktorý je potrebný pre tréning neurónovej siete. Poslednú natrénovanú neurónovú sieť bude systém vyhodnocovať a následne výsledky uloží do databázy. Výsledky budú v databáze uložené v tabuľke, z ktorej si používateľ môže pomocou webového rozhrania alebo príkazového riadku vybrať všetky požadované údaje z jej histórie napr. dátum, konfiguračný súbor, výsledky, staršie natrénované neurónové siete. Webové rozhranie umožní porovnanie výstupov KLFittera a neurónovej siete alebo porovnanie dvoch neurónových sietí.

2.3 Charakteristika používateľa

Tento systém je určený pre fyzikálnych výskumníkov, ktorí pracujú s top kvarkami a jetmi. Systém bude použiteľný jedine po prihlásení.

2.4 Všeobecné obmedzenia

System bude využívať server, databázu a pripojenie na internet.

2.5 Predpoklady a závislosti

- úložný priestor na serveri
- internet
- server na ktorom bude bežať program

3. Špecifické požiadavky

3.1 Funkčné požiadavky

3.1.1. KLFitter - Systém ponúkne využitie metódy KLFitter na rekonštrukciu top kvarkov. Algoritmus pracuje s permutáciami a ráta pravdepodobnosti zoradenia jetov. Na výpočet pravdepodobnosti táto metóda využíva fyzikálne informácie. Výsledok uloží do databázy.

3.1.2. Hlboká neurónová sieť – Systém ponúkne ďalšiu metódu na rekonštrukciu top kvarkov. Podľa vstupných súborov pre natréňovanie neurónovej siete, systém natrénuje neurónovú sieť podľa konfiguračných súborov obsahujúcich všetky potrebné parametre a následne použije túto metódu.

3.1.3. Databáza - Výsledky z metód KLFitter a neurónovej siete bude systém ukladať do databázy. Okrem uložených výsledkov z metód KLFittera a neurónovej siete, bude obsahovať aj informáciu o dĺžke procesorového času, ktorý bol potrebný na vykonanie metód.

3.1.4. Porovnávanie – Systém dokáže aplikovať metódy KLFitter a neurónovej siete nezávisle od seba. Systém môže porovnať KLFitter s neurónovou sieťou alebo môže porovnať dve neurónové siete, každá s inými parametrami. Výsledky metód sa uložia do databázy. Používateľ si bude môcť vo webovom rozhraní výsledky pozrieť v grafe, porovnať ich.

3.1.5. Spúšťanie - Systém sa bude dať spustiť dvoma spôsobmi. Prvou možnosťou bude spustenie cez príkazový riadok. Druhá možnosť je spustenie cez webové rozhranie, čo znamená, vygenerovanie príkazu pre príkazový riadok.

3.1.6. Premenné - Vo webovom rozhraní si používateľ zvolí premenné ručne. Bude si môcť vybrať z 20 premenných. Ide o premenné, ktoré by štandardne boli vo vstupnom súbore.

3.1.7. Human readable - Výstup systému bude human readable – čitateľný formát napr.: yaml, json, plain text a pod., aby sa výsledky dali zhodnotiť aj ručne.

3.1.8. Vstup - Systém bude vedieť pracovať so súbormi typu root a bude schopný interagovať s knižnicou KERAS

3.1.9. Trénovanie - Výstup z tréňovania DNN bude použiteľný v C++ (pozrieť bod 1.4.4 - LWTNN [4])

3.2 Kvalitatívne požiadavky

- **Procesy** - Systém by mal vedieť spustiť viac procesov (vstupov) naraz.
- **Stabilita** - Systém by nemal padať.
- **Rýchlosť** - Systém by mal pracovať čo najefektívnejšie, avšak pre veľké vstupy sa doba bežania odhadnúť nedá. (tzn. systém môže bežať niekoľko hodín, dní... ale nevieme to vopred odhadnúť). Aj napriek tomu bude potrebné optimalizovať algoritmy KLFitteera, aby bežal rýchlejšie.

3.3 Požiadavky rozhrania

- **Výber metódy** - V rozhraní sa bude nachádzať možnosť výberu, či chceme robiť rekonštrukciu top kvarkov pomocou neurónovej siete alebo cez KLFIitter
- **Heslo** - Rozhranie bude zaheslované
- **Stav** - Rozhranie bude zobrazovať, na koľko percent je proces dokončený

Poznámka: V tomto projekte po dohode so zadávateľom nebudeme využívať KLFIitter na prácu s neurónovými sieťami.

Časť II

Návrh

1. Úvod

1.1 Účel dokumentu

Tento dokument slúži ako návrh pre systém na rekonštrukciu top kvarkov. Dokument dôkladne popisuje funkcie a metódy systému a podáva návrh na implementáciu.

1.2 Prehľad nasledujúcich kapitol

Nasledujúce kapitoly budú venované kompletnému návrhu systému, opísaného slovne aj pomocou diagramov.

2. Podrobná špecifikácia vonkajších interfejsov

Systém bude bežať na serveri atlas23 a komunikovať s databázou, kde budú uložené potrebné súbory na tréning neurónových sietí.

3. Používané technológie

3.1 HTML / CSS / PHP

Používateľské rozhranie aplikácie je tvorené pomocou HTML, CSS a PHP programovacích jazykov.

3.2 ROOT

Vstupné súbory na prácu s neurónovou sieťou. Nachádzajú sa na serveri (atlas23). Obsahujú údaje o permutáciách. Súbor je rozdelený na 2 tabuľky (keys). V týchto tabuľkách sa nachádzajú údaje o eventoch, premenných, permutáciách a pod.

3.3 KERAS

KERAS umožňuje rýchle experimentovanie prostredníctvom vysokoúrovňového, ľahko použiteľného, modulárneho a rozširiteľného API.

Aplikácia bude využívať KERAS pri tréningu neurónovej siete. Všetky údaje vloží do KERASU (na webovom rozhraní si používateľ vyberie aké chce premenné a pod.) Po spustení tréningu ponúkne napr. obrázok či tréning prebehlo správne alebo nie. Všetky tieto údaje sa uložia do databázy.

3.4 Tensorflow

Bezplatná a otvorená softvérová knižnica pre dataflow a diferencovateľné programovanie v rôznych úlohách. Je to matematická knižnica a používa sa tiež na aplikácie strojového učenia, ako sú neurónové siete.

3.5 Matplotlib

Knižnica na vykresľovanie grafov v Pythone. Podporuje dáta reprezentované ako *array* z *NumPy*. Pomocou tejto knižnice sa po natrénovaní neurónovej siete vizualizujú dáta v podobe grafov (plotov). Budú v ňom znázornené krivky tréovania a validovania.

3.6 NumPy

Knižnica pre Python, ktorá poskytuje viacrozmerné polia a objekty napr. matice. Ponúka rýchle matematické operácie napr. usporiadavanie (sorting), transformácie, základné algebraické operácie (atď.) na viacrozmerných objektoch.

3.7 LWTNN

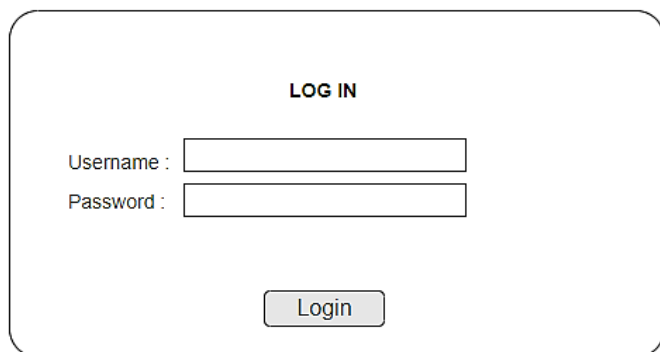
Knižnica na prácu s neurónovými sieťami. Obsahuje sadu scriptov na prevod uložených neurónových sietí na štandardný formát JSON a skupinu tried, ktoré rekonštruujú neurónovú sieť na použitie v prostredí C ++.

4. Používateľské rozhranie

V tejto časti sa nachádza návrh používateľského rozhrania. Bližšie informácie k funkciám možno nájsť aj v katalógu požiadaviek, z ktorého návrh vychádza.

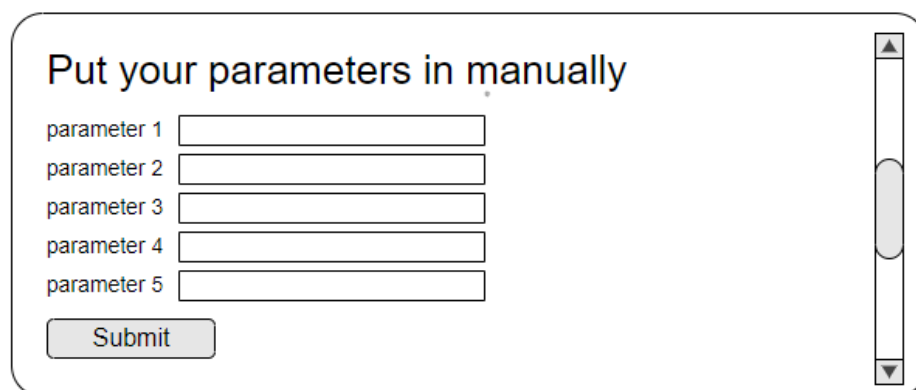
Návrh používateľského rozhrania sa nachádza na nasledujúcich stranách.

Používateľské rozhranie pred prihlásením.



A login form with a rounded rectangular border. At the top center is the text "LOG IN". Below it are two input fields: the first is preceded by the label "Username :" and the second by "Password :". At the bottom center is a button labeled "Login".

Používateľské rozhranie po prihlásení. Používateľ si zvolí premenné ručne. Bude si môcť vybrať z 20 premenných. (Ide o premenné, ktoré by štandardne boli vo vstupnom súbore.) Po kliknutí na tlačidlo *Submit* sa parametre uložia do konfiguračného súboru.



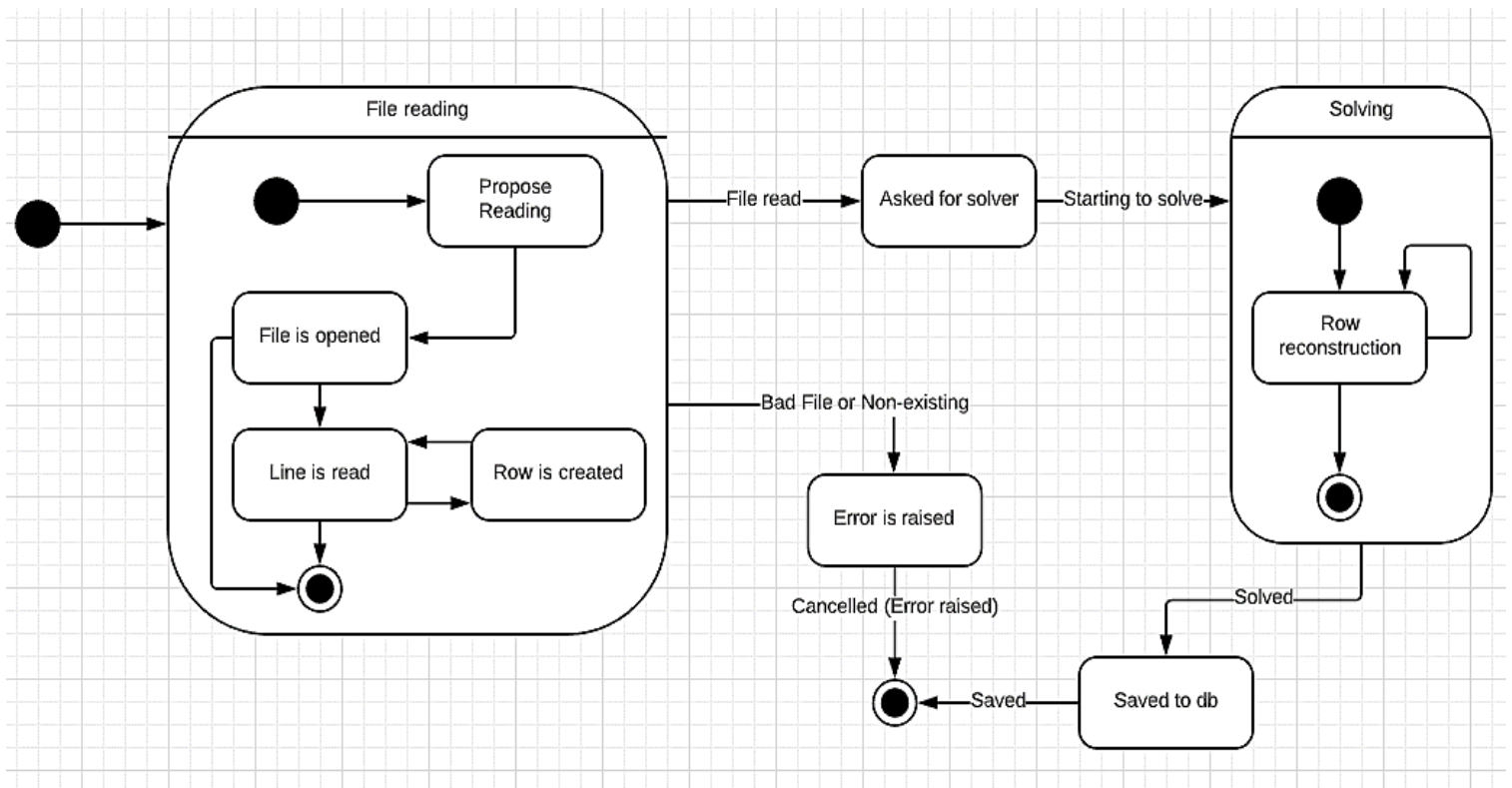
A form titled "Put your parameters in manually" with a vertical scrollbar on the right. It contains five rows, each with a label ("parameter 1" through "parameter 5") and an input field. At the bottom left is a button labeled "Submit".

5. Návrh implementácie

5.1 UML – state diagram

Stavový diagram opisuje životný cyklus entity rekonštrukcie top kvarkov.

V diagrame sú stavy entity označené oválmi. Prechod medzi týmito stavmi vyjadrujeme pomocou šípok, pričom stav môže prejsť aj sám do seba(cykliť sa). Do stavového diagramu entita vstupuje cez počiatočný stav, ktorý sa znázorňuje čiernym kruhom a vystupuje z neho koncovým stavom, ktorý sa označuje ako čierny kruh s kružnicou, ktorá má o trochu väčší polomer.



5.2 UML – class diagram

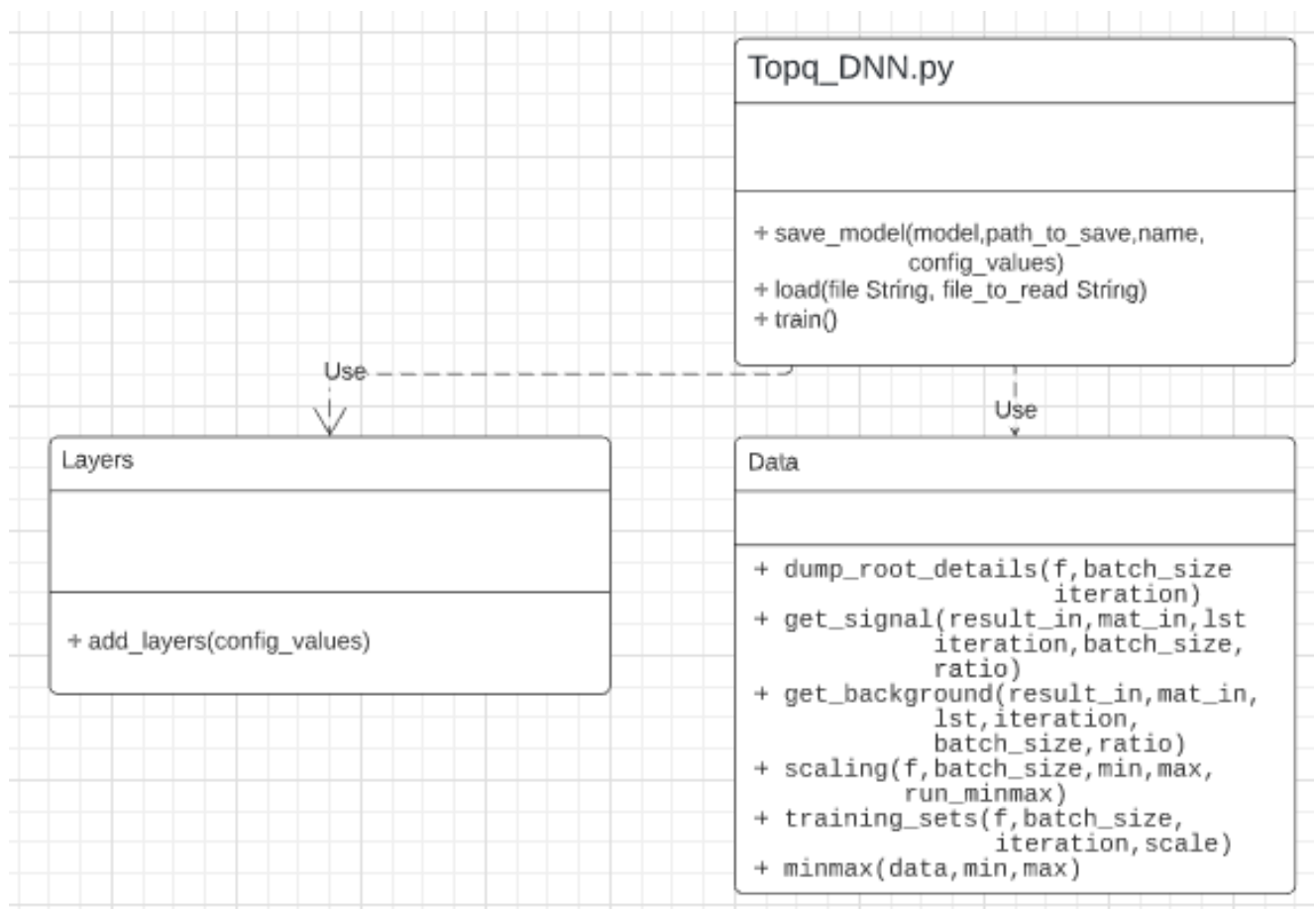
Class diagram zobrazuje vzťahy medzi triedami, ale aj ich metódy a premenné.

V diagrame sú triedy zobrazené v štvorcoch, pričom prvý riadok určuje názov triedy(v zložitejších triedach môžeme povedať, že sa jedná o riadok nad prvou čiarou v štvorci).

Premenné v týchto daných triedach sú zobrazené medzi prvou a druhou čiarou (z vrchu).

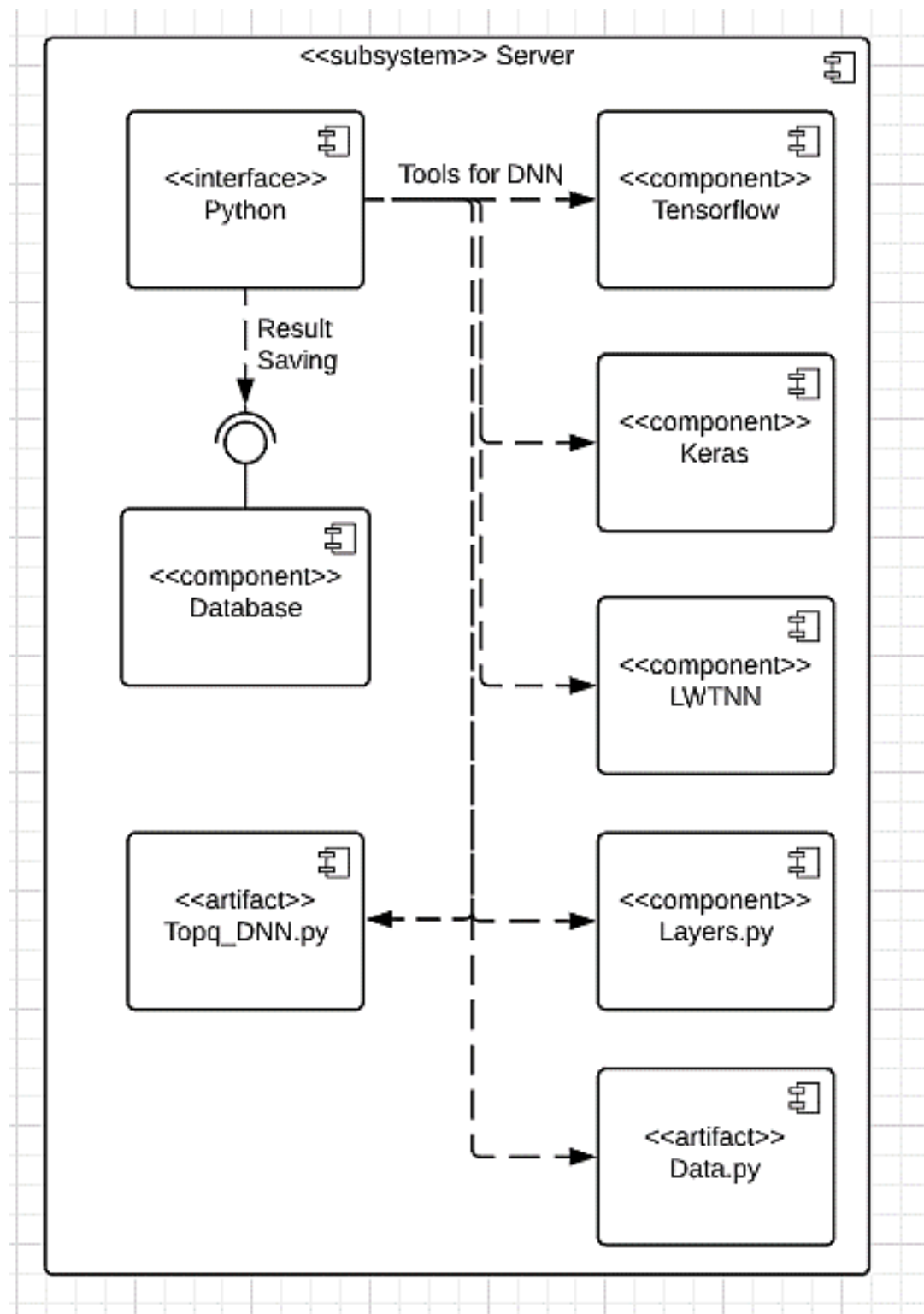
Metódy týchto tried sa značia pod druhú čiaru.

Všetky metódy a premenné v triednom diagrame sú popísané nižšie v návrhu v bode č.6.



5.3 UML – component diagram


Component diagram ponúka statický pohľad na rozdelenie systému podľa nejakého konkrétneho členenia. V tomto prípade na serverovú časť. Serverová časť zapuzdruje menšie komponenty, ktoré využíva.



6. Popis jednotlivých súborov

6.1 config.txt

V konfiguračnom súbore – config.txt sa nachádzajú parametre potrebné k natrénovaniu neurónovej siete.

 config – Poznámkový blok

```
Súbor  Úpravy  Formát  Zobrazit  Pomocník
filename = InputFile_10000.root
path = /home/topqproject/src/palo/input
batch_size = 1000
alpha = 0.001
iteration = 0
path_to_save = /home/topqproject/src/palo/saved/topqproject_model_DNN0
saved_as = topqproject_model_DNN_test
minimum = -500000
maximum = 500000
run_minmax = 1
loss = mse
metrics = accuracy
monitor = val_acc
mode = auto
512,relu
256,relu,10e-9
128,relu
64,relu
32,relu
1,sigmoid
```

filename - názov súboru typu .root, ktorý chceme natrénovať

path – „cesta“ k súboru, ktorý je zadaný vo *filename*

batch_size – počet vzoriek po koľkých sa natrénujú dáta

alpha – rýchlosť učenia siete

iteration – koľkým batch_size budeme brať dáta

path_to_save – „cesta“ kam sa uložia súbory po natrénovaní

saved_as – názov akým sa natrénovaná sieť uloží na server

minimum – spodné ohraničenie dát

maximum – horné ohraničenie dát

run_minmax – spúšťanie ohraničenia (1 – spúšťa sa, 0 – nespúšťa sa)

loss – určuje ako sa odlišujú predikované hodnoty od skutočných hodnôt v tréningových dátach (*mse* – *mean squared error* – priemer rozdielov medzi predikovanými a skutočnými hodnotami)

metrics – hodnoty metrics v tejto premennej sú zaznamenávané na konci každej epochy tréningovania

(*accuracy* – typ metriky)

monitor – ktorú veličinu sledujeme, kvôli skorému zastaveniu

mode – mód ktorým sledujeme tréningovanie

ostatné parametre - nastavenia siete - hodnoty vo vrstvách neurónovej siete, ktoré si používateľ môže meniť - pridávať / odoberať

6.2 Layers.py

Tento súbor obsahuje **add_layers** s parametrom **config_values**.

Config_values – pole v ktorom sa nachádzajú hodnoty z konfiguračného súboru – config.txt.

Metóda **add_layers** pridáva vrstvy do neurónovej siete a mení hodnoty na základe zadaných parametrov v konfiguračnom súbore.

6.3 Topq_DNN.py

Tento súbor obsahuje 6 metód.

- **train()** – tréning siete, vykresľovanie plotov (vizualizácia dát)
- **save_model(model, path_to_save, name, config_values)** – ukladá súbory po natrénovaní
- **load(file, file_to_predict)** – *file* načíta neurónovú sieť, *file_to_predict* – meno súboru, ktorý chceme nechať otestovať
- **get_model(path)** - ukladá load do formátu .json alebo .h5
- **contains_all(files, file)** – zisťuje či sa *file* nachádza vo *files* - kontroluje či sa všetky 3 súbory (.txt, .json, .h5) nachádzajú vo *files*
- **read_config(path, name)** – načítanie konfiguračného súboru

6.4 Data.py

Tento súbor obsahuje funkcie:

- **dump_root_details(f, batch_size, iteration)** – spracovanie .root súboru
- **get_signal(result_in, matica_s_in, lst, iteration, batch_size, ratio)** – vráti signálovú časť .root súboru.
- **get_background(result_in, matica_s_in, lst, iteration, batch_size, ratio)** – vráti background časť .root súboru
- **scaling(f, batch_size, minimum, maximum, run_minimax)** – naškálovanie dát v .root súbore podľa zadaných parametrov
- **training_sets(f, batch_size, iteration, scale)** – volanie iných funkcií (napr. *dump_root_details*)

7. Formáty jednotlivých súborov

7.1 .root

Štruktúra týchto súborov je strom. Sú to vstupné súbory na prácu s neurónovou sieťou. Nachádzajú sa na serveri (atlas23). Obsahujú údaje o permutáciách. Súbor je rozdelený na 2 tabuľky (keys). V týchto tabuľkách sa nachádzajú údaje o eventoch, premenných, permutáciách a pod.

7.2 .json

Súbory typu .json – v týchto súboroch sú uložené štruktúry neurónovej siete. Vytvoria sa až po natrénovaní. Sú uložené na serveri(atlas23) v databázovom priečinku *saved*.

7.3 .h5

Súbory typu .h5 – v týchto súboroch sú uložené váhy neurónovej siete. Vytvoria sa až po natrénovaní. Sú uložené na serveri(atlas23) v databázovom priečinku *saved*.

7.4 .png

Súbory typu .png – v týchto súboroch sú uložené grafy modelov. V grafe na x-ovej osi sú zobrazené epochy, na y-ovej osi zobrazujeme v jednom modeli *loss* a v druhom *accuracy*. V grafe sú zobrazené krivky validácie a tréovania.

Časť III

Testovacie scenáre

1. Prihlasovanie

- Po zadaní nesprávneho prihlasovacieho mena alebo hesla sa zobrazí chybová hláška
- Po zadaní správneho prihlasovacieho mena a hesla, užívateľa webové rozhranie presmeruje na ďalšiu stránku, kde sa nachádza tabuľka na voľbu premenných

2. Voľba premenných

- Pokiaľ užívateľ nezvolí všetky premenné a pokúsi sa spustiť program, zobrazí sa chybová hláška
- Pokiaľ užívateľ zvolí všetky premenné a klikne na tlačítko "spustiť" program sa spustí

Časť IV

Príručka

1. Používateľ

1.1. Prihlásenie

Po otvorení webového prehliadača, sa používateľ môže po zadaní mena a hesla prihlásiť do systému. Meno používateľa – kvarky, heslo – uJa6Dn9r. Systém bude po prihlásení ponúkať nasledujúce kroky.

1.2. Nastavenie parametrov

Používateľ vo webovom rozhraní zvolí parametre na natrénovanie neurónovej siete. (Ide o premenné, ktoré by štandardne boli vo vstupnom súbore.) Tieto súbory budú následne uložené ako config súbor, ktorý bude uložený na serveri v priečinku *web*.

1.3 Spustenie tréovania

Používateľ sa pomocou Putty pripojí na server. Pomocou príkazu spustí tréovanie alebo load.

Tréovanie – python “*meno_triedy*“(Topq_DNN.py) train

Load – python “*meno_triedy*“(Topq_DNN.py) load “*nazov_modelu*” “*cesta k priečinku na predikciu*”

1.4. Ukladanie modelu

Po natrénovaní neurónovej siete budú výsledky uložené na serveri v priečinku s názvom *saved*. V priečinku sa vytvorí súbor s názvom, ktorý používateľ zadal v config súbore. V ňom budú uložené ďalšie 3 súbory. Jeden so súborov je json file – štruktúra neurónovej siete, v druhom súbore (.h5) budú uložené váhy siete, v treťom bude uložený textový súbor, ktorý bude mať názov taký, aký si používateľ zvolil na začiatku a dátum kedy bola sieť natrénovaná.

1.5 Vizualizácia

Po natrénovaní sa používateľovi zobrazí vizualizácia dát - pomocou grafov, kde si bude môcť porovnať presnosť daného modelu - model accuracy a stratu – model loss - model, ktorý bol natrénovaný s testovacím modelom.

