

Spis treści

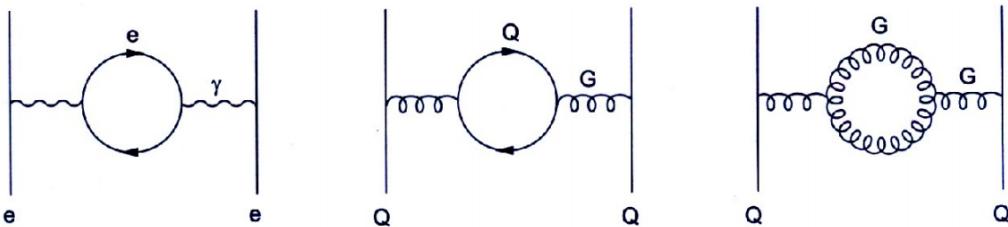
| | |
|------------------------------------------------|-----------|
| 1 Fizyka w | 2 |
| 1.1 Plazma kwarkowo-gluonowa | 3 |
| 1.2 Dzety | 3 |
| 1.3 Identyfikacja dżetów b | 4 |
| 1.4 Eksperyment ALICE | 6 |
| 2 Uczenie maszynowe | 9 |
| 2.1 Wzmacniane drzewa decyzyjne | 9 |
| 2.2 Sieci neuronowe | 10 |
| 2.3 Dyskusja użycia dwóch algorytmów | 15 |
| 3 Dane | 16 |
| 4 Analiza | 18 |
| 4.1 Dobór metryki | 18 |
| 4.2 Wyniki dla zmiennych SV | 21 |
| 4.3 Wyniki dla zmiennych constit | 21 |

16 1 Fizyka w

17 Chromodynamika kwantowa

18 Chromodynamika kwantowa (ang. *Quantum Chromodynamics – QCD*) to kwantowa teoria
19 pola opisująca oddziaływanie silne. Wprowadza ona dla kwarków nową liczbę kwantową nazy-
20 waną kolorem lub ładunkiem kolorowym, który jest odpowiednikiem ładunku elektrycznego w
21 elektrodynamice kwantowej (ang. *Quantum Electrodynamics – QED*), ale w przeciwnieństwie do
22 niego może przyjmować 3 różne wartości (i trzy antykolory dla antykwarków). Elementarne od-
23 działywanie w obu teoriach przenoszone są przez bezmasowe bozony pośredniczące: w *QED* jest
24 to elektrycznie obojętny foton a w *QCD* gluony, które występują w 8 odmianach i są kolorowo
25 naładowane, przez co możliwe jest oddziaływanie zachodzące między dwoma gluonami.

26 Próżnia, w rozumieniu klasycznym będąca zupełnie pusta, w teoriach kwantowych wypeł-
27 niona jest pojawiającymi i znikającymi wirtualnymi cząstkkami. Cząstki te ekranują ładunek
28 próbny umieszczony w kwantowej próżni, wywołując zjawisko polaryzacji próżni (analogiczne
29 do polaryzacji dielektryków), które efektywnie zmniejsza pole wytwarzane przez ten ładunek.
30 Siła tego zjawiska zależy od liczby ekranujących cząstek, czyli pośrednio od skali odległości.
31 Skala ta wyznaczona jest przez długość fali próbującej cząstki, zatem także jej energii (im
32 większa energia, tym mniejsza długość fali i mniejsza ilość ekranujących cząstek obserwowanych
33 w pobliżu rzeczywistego ładunku, zatem tym słabszy efekt ekranowania i większy efektywny
34 ładunek). Prowadzi to do zależnej od energii stałej sprzężenia α , którą nazywamy efektywną
35 lub biegnącą stałą sprzężenia (ang. *running coupling constant*).

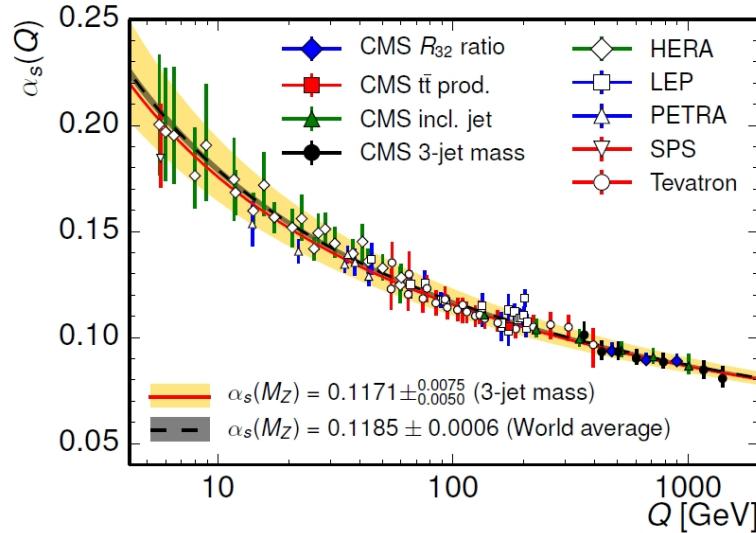


Rysunek 1: Diagramy Feynmana opisujące polaryzację próżni w *QED* (lewy) i *QCD* (środkowy i prawy). Rysunki lewy i środkowy są swoimi odpowiednikami w tych dwóch teoriach, natomiast prawy, w którym oddziałują jedynie bozony pośredniczące nie ma swojego odpowiednika w *QED*. Źródło: [1].

36 Zarówno pary elektron-pozyton jak i kwark-antykwark działają ekranującą kolejno ładunek elektryczny i kolorowy. Jednak jak zostało to już wspomniane, w przypadku *QCD* możliwe
37 jest także samooddziałanie gluonów, przez co dopuszczalne są diagramy Feynmana jak ten
38 przedstawiony na Rys. 1 po prawej. Pętle gluonowe działają anty-ekranującą – zwiększą efek-
39 tywną wartość stałej sprzężenia, ponadto jest to efekt dominujący nad przyczynkiem od par
40 kwark-antykwark, co sprawia że zależność biegnącej stałej sprzężenia w *QCD* jest odwrotna i
41 dużo silniejsza niż w przypadku *QED*. Wartość α_{em} rośnie od wartości $\frac{1}{137}$ przy energii bliskiej
42 zeru do $\frac{1}{128}$ przy energiach ok. 90 GeV, co oznacza zmianę o kilka procent. Tymczasem α_S
43 rośnie asymptotycznie w miarę zbliżania się do niskich energii (por. Rys. 2) osiągając wartości
44 $\alpha_S > 1$ dla energii poniżej 200 MeV. Prowadzi to do dwóch zjawisk charakterystycznych dla
45 chromodynamiki kwantowej:

- 46
- asymptotyczna swoboda – dla wysokich energii ($\gtrsim 100$ GeV) silna stała sprzężenia jest
47 mała ($\lesssim 0.1$) i kwarki wewnątrz hadronów zachowują się jak cząstki quasi-swobodne. W
48 tym zakresie energii możliwe jest stosowanie rachunku perturbacyjnego.

- 50 • uwięzienie koloru – przy zwiększaniu odległości między partonami siła oddziaływanie ro-
 51 śnie do nieskończoności, dlatego nigdy nie obserwuje się swobodnych cząstek obdarzonych
 52 ładunkiem kolorowym.



Rysunek 2: Zależność silnej stałej sprzężenia α_S od przekazu czteropędu Q .

53 1.1 Plazma kwarkowo-gluonowa

54 Plazma kwarkowo-gluonowa (ang. *quark-gluon plasma – QGP*) to stan materii, który istniał w
 55 pierwszych ułamkach sekund po Wielkim Wybuchu. Przewiduje się, że materia w takim stanie
 56 obecna jest także w jądrach gwiazd neutronowych [2].

57 Obecnie aby uzyskać dostęp do materii w stanie plazmy kwarkowo-gluonowej potrzebne są
 58 wysokoenergetyczne zderzenia cząstek. Powszechnie mówi się o niej w kontekście zderzeń cięż-
 59 kich jonów, chociaż istnieją także prace doszukujące się obecności *QGP* w mniejszych systemach
 60 np. w zderzeniach proton-proton [3], [4].

61 *QGP* jest stanem o ekstremalnej gęstości i temperaturze. Jego cechą charakterystyczną jest
 62 obecność wolnych kwarków i gluonów (zbiorczo nazywanych partonami). W każdym innym sta-
 63 nie materii są one zawsze związane i tworzą hadrony. Zjawisko to, zwane uwięzieniem koloru
 64 (ang. *color confinement*) uniemożliwia obserwację cząstek obdarzonych ładunkiem kolorowym
 65 (a takimi są kwarki i gluony) w stanie wolnym. Wolne kwarki i gluony powstające w zderzeniach
 66 muszą zatem przejść przez proces hadronizacji, w którym rekombinują one ze spontanicznie wy-
 67 twarzanymi nowymi partonami, tworząc hadrony. W wyniku tego procesu, z każdego partonu
 68 obecnego w początkowym etapie zderzenia może powstać wiele cząstek poruszających się po-
 69 dobrym kierunku, tworząc stożek z wierzchołkiem blisko punktu interakcji wiązek. Taki stożek
 70 skolimowanych cząstek nazywany jest dżetem cząstek.

71 1.2 Dżety

72 Przedstawiona powyżej definicja dżetu nie jest precyzyjna z punktu widzenia pracy eksperymen-
 73 talnej. W detektorze obserwuje się tylko cząstki w stanie końcowym, nie jest znana natomiast
 74 ich historia (tj. parton, z którego powstały), nie jest zatem możliwe przyporządkowanie cząstki
 75 według jej pochodzenia. W związku z tym, konieczne jest użycie algorytmu klasteryzującego,
 76 dostającego na wejściu tylko obserwowalne eksperymentalnie cząstki. To jakie dżety zostaną

zaobserwowane w danym zdarzeniu zależy od użytego algorytmu. Oznacza to, że precyzyjną definicję dżetu stanowi algorytm klasteryzujący wraz z zestawem parametrów. Obecnie najpowszechniej stosowanym algorytmem jest algorytm *anti-kt* [5].

Eksperymentalne ograniczenia związane z obserwacją tylko końcowego stanu oddziaływań nie występują w analizie danych z symulacji Monte Carlo (MC), gdzie ma się dostęp do pełnej informacji na temat historii każdej cząstki. Można pomyśleć, że daje to możliwość lepszej klasteryzacji dżetów. Rekonstrukcja przy użyciu informacji o partonach - matkach sprawia jednak, że definicja dżetu wykorzystana w badaniach danych symulacyjnych jest inna od tej wykorzystywanej w eksperymencie. Tracona jest przez to cecha odpowiedniości między obiektymi nazywanymi dżetami w symulacji i w eksperymencie, która to cecha jest niewątpliwie jedną z podstawowych wymagań stawianych przed dobrą symulacją.

Dżety stanowią ważny element w badaniach plazmy kwarkowo-gluonowej. Dają one pośredni wgląd we właściwości *QGP* na podstawie jej wpływu na oddziaływanie z nią partony. Przykładową obserwablą mierzoną w zderzeniach ciężkich jonów jest czynnik modyfikacji jądrowej (ang. *nuclear modification factor*), który jest miarą strat energii przez parton przechodzący przez medium.

Oprócz globalnego wpływu medium na dżety, analizuje się także różnice między dżetami pochodząymi z gluonów oraz kwarków o różnych zapachach (ang. *flavours*). Modele teoretyczne przewidują między innymi większe straty energii w wyniku interakcji z *QGP* dla dżetów gluonowych niż kwarkowych [6] oraz zależność strat energii od masy partonu [7] – w tym przypadku precyzyjne pomiary rozróżniające typy dżetów pozwalają lepiej zrozumieć mechanizm odpowiadający za straty energii przez partony. Zagadnienie rozpoznania z jakiego rodzaju partonu powstał dżet, nazywane jest identyfikacją lub tagowaniem dżetu.

1.3 Identyfikacja dżetów *b*

Poza badaniami właściwości *QGP*, szczególnie znaczenie ma identyfikacja dżetów pochodzących z ciężkich kwarków: *b* i *c*. Są one ważnym elementem w poszukiwaniu łamania symetrii *CP* w rozpadach hadronów B i D oraz innych sygnatur tzw. *Nowej Fizyki* wykraczającej poza ramy Modelu Standardowego. Kwarki *piękne* pojawiają się także często w kanałach rozpadu cząstek takich jak bozon Higgsa i kwark *t*.

Identyfikacja dżetów *b* jest sporem wyzwaniem ze względu na zdecydowanie częściej występujące dżety lekkie, tj. powstałe z hadronizacji kwarków *u,d,s* oraz gluonów. Rozpoznawanie dżetów *b* bazuje na charakterystycznych właściwościach hadronów zawierających kwark piękny: relatywnie długim czasie życia oraz (w mniejszym stopniu) na ich pół-leptonowym rozpadach o względnej częstości rozpadu w tym kanale (ang. *branching ratio*) na poziomie 10%.

przegląd algorytmów używanych w identyfikacji b-jetów: TBD

Używane w eksperymetach na LHC: ATLAS, CMS i ALICE algorytmy można podzielić na trzy kategorie: wykorzystujące wtórne wierzchołki, informację o odległości najbliższego zbliżenia (parametrach zderzenia) (ang. *Distance of Closest Approach – DCA*, *Impact Parameter – IP*) cząstek tworzących dżet oraz identyfikujące produkty półleptonowych rozpadów pięknych lub powabnych hadronów. Dokładne opisy omawianych algorytmów można znaleźć w: [8], [9] (ATLAS), ... (CMS), ... (ALICE).

Najprostszym algorytmem jest dyskryminacja na podstawie istotności statystycznej (wynik pomiaru podzielony przez jego niepewność) odległości wtórnego wierzchołka od wierzchołka pierwotnego *L*. Jest to metoda wykorzystywana w każdym z trzech wymienionych eksperymentów (por. ATLAS: algorytm SV0, CMS: algorytm SSV, ALICE). Może być ona rozszerzona poprzez użycie dodatkowych zmiennych opisujących wtórnego wierzchołek jak na przykład jego masa, ułamek niesionej przez niego całkowitej energii dżetu (por. ATLAS: SV1) lub użycie "pseudowierzchołków" (kombinacji dwóch cząstek o dużych *DCA*) w celu poprawienia wydaj-

125 ności detekcji o przypadki, w których wtórny wierzchołek nie został zrekonstruowany (por.
126 CMS: CSV).

127 Algorytmy wykorzystujące informację o poszczególnych cząstkach mogą zasadniczo bazować
128 albo na sumie logarytmów prawdopodobieństw pochodzenia każdej cząstki z pierwotnego wierz-
129 chołka (por. ATLAS: IP3D, CMS: JP) lub tym samym prawdopodobieństwie ale dla wybranej,
130 np. drugiej lub trzeciej cząstki na liście posortowanej według malejącego *IP* (por. ALICE i
131 CMS: TC). Bardziej złożonym podejściem, w którym cząstki nie są traktowane jako niezależne,
132 jest użycie rekurencyjnych sieci neuronowych (por. ATLAS: RNNIP).

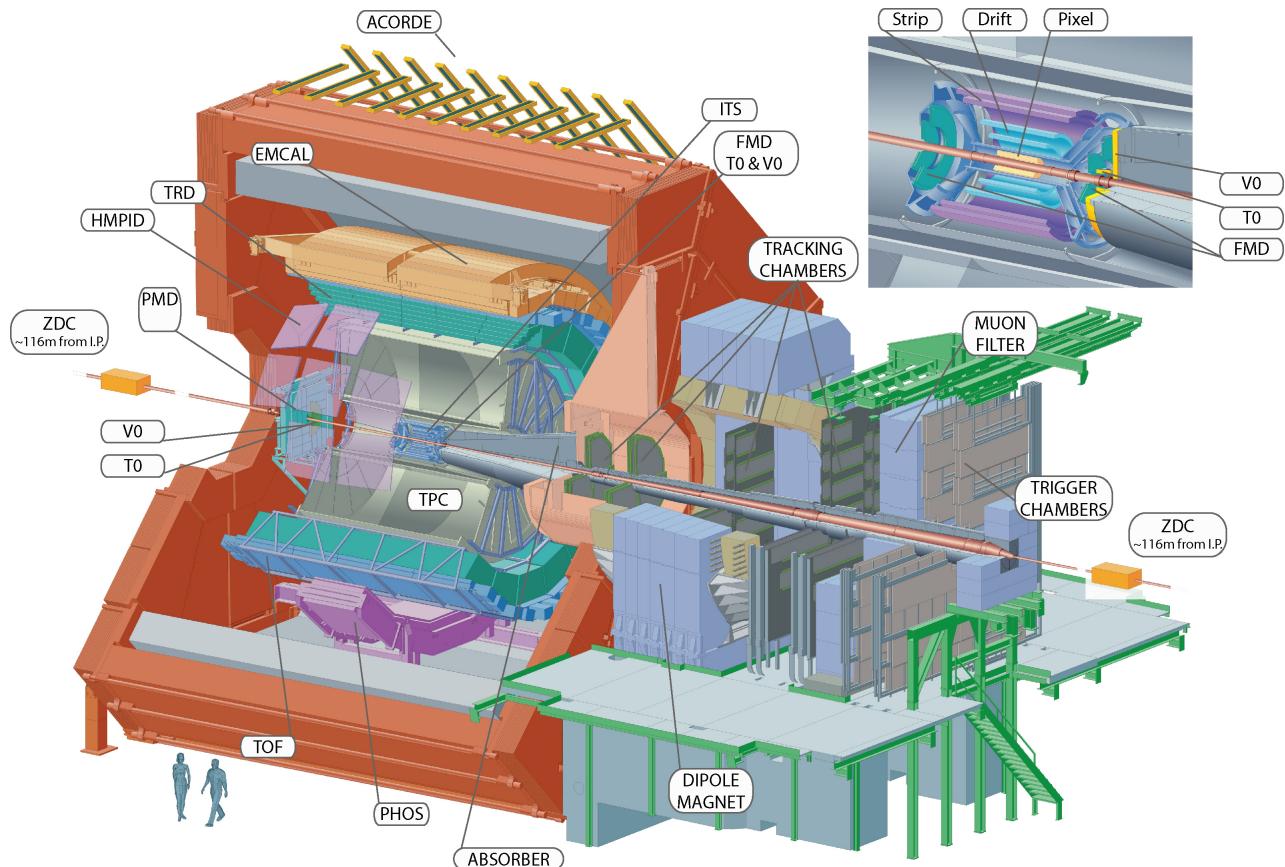
133 Do wykorzystania półleptonowego kanału rozpadu ciężkich hadronów do identyfikacji dżetów
134 *b* w eksperymetach ATLAS (SMT) i CMS (SE, SM) użyto wzmacnianych drzew decyzyjnych
135 trenowanych na kilku ręcznie zdefiniowanych w tym celu zmiennych takich jak pęd leptonu
136 transwersalny względem osi dżetu.

137 Znaczącą poprawę zdolności predykcyjnej można uzyskać łącząc kilka różnych modeli. Al-
138 gorytm łączący może pobierać na wejściu albo tylko predykcje klasyfikatorów niższego poziomu
139 (CMS: cMVAv2) lub dodatkowo także ich zmienne wejściowe (ATLAS: MV2, DL1). Do scalania
140 używane są zwykle wzmacniane drzewa decyzyjne lub sieci neuronowe.

141 Przykład innego podejścia zaprezentowała współpraca przy eksperymencie LHCb, gdzie
142 wykorzystano dwa zestawy wzmacnianych drzew decyzyjnych operujących na zmiennych zwią-
143 zanych z wtórnymi wierzchołkami. Pierwszy zapewnia separację dżetów lekkich od ciężkich a
144 drugi odróżnia dżety *b* od *c*. Do wyboru punktu pracy, zamiast jednowymiarowego rozkładu
145 predykcji używany jest dwuwymiarowy rozkład wag przypisany przez oba klasyfikatory.

146 1.4 Eksperyment ALICE

147 Eksperyment ALICE jest jednym z czterech największych eksperymentów na LHC. Jest on
 148 dedykowany zderzeniom ciężkich jonów (w LHC są to jony ołowiu PbPb), ale mierzone są tak-
 149 że mniejsze systemy, tj. proton-proton pp (głównie jako referencję dla pomiarów PbPb) oraz
 150 proton-ołów p+Pb, które dostarczają także okazji do badania asymetrycznych zderzeń. Cechą
 151 charakterystyczną pomiarów ciężkojonowych jest ich znacznie większa niż w przypadku zderzeń
 152 pp krotność, tj. liczba cząstek wyprodukowana w pojedynczym zderzeniu. W przypadku zde-
 153 rzeń PbPb może powstawać nawet do 8000 naładowanych cząstek na jednostkę pseudorapidity.
 154 Detektor ALICE został zoptymalizowany do mierzeniach takich przypadków, jak również pod-
 155 kątem rekonstrukcji i identyfikacji cząstek o szerokim zakresie pędów (100 MeV – 100 GeV).



Rysunek 3: Schemat detektora ALICE.

156 Detektor ALICE jest urządzeniem złożonym z wielu subdetektorów, schematycznie przed-
 157 stawionych na Rys. 3. Można je podzielić według pełnionej w pomiarach roli. ITS, TPC, TRD
 158 oraz TOF pokrywają pełen kąt azymutalny oraz zakres pseudopospieszności $|\eta| < 0.9$.

- 159 • Detektory śladowe – mierzące trajektorie cząstek zakrzywiane w polu magnetycznym o
 160 wartości $B = 0.5$ T.
 - 161 – Inner Tracking System (ITS) – zespół krzemowych detektorów śladowych znajdujący-
 162 się najbliżej miejsca interakcji wiązek. Składa się on z 6 cylindrycznych warstw
 163 o promieniach od 4 do 43 cm, wykonanych w trzech różnych technologiach. Jego
 164 główną rolą jest rekonstrukcja pierwotnego oraz wtórnego wierzchołków. Bierze tak-
 165 że udział w rekonstrukcji trajektorii i strat energetycznych cząstek, szczególnie tych
 166 niskopędowych, które nie docierają do dalej położonych detektorów.

167 – Time Projection Chamber (TPC) – dłuża na 5m i o takiej średnicy komora projekcji
168 czasowej. Jest to główny detektor śladowy ALICE, wraz z ITS służy do wyznaczania
169 trajektorii cząstek i na ich podstawie również wierzchołków zderzenia. Elektryny
170 uwolnione ze zjonizowanego przez poruszające się w nim na ładowane cząstki gazu
171 dryfują wzdłuż kierunku wiązki w stronę końcowych elektrod. Następnie są tam
172 zbierane dostarczając informacji o dwóch współrzędnych toru cząstki: odległości od
173 wiązki i kącie azymutalnym. Trzecia składowa trajektorii jest otrzymywana na pod-
174 stawie czasu dotarcia elektronów do elektrod. TPC jest najwolniejszym detektorem
175 ALICE (ze względu na ograniczający czas dryfu elektronów wynoszący $\sim 90 \mu\text{s}$),
176 użycie detektora tego typu podyktowane jest jego zdolnością do rozwikłania śladów
177 tysięcy cząstek spodziewanych w centralnych zderzeniach PbPb.

178 Znajomość toru ruchu cząstki pozwala na wyznaczenie jej pędu. Oprócz dokładnej
179 trajektorii każdej cząstki próbkowanej do 159 razy, TPC mierzy straty energii czą-
180 stek dE/dx . Pozwala to na ich identyfikację na podstawie wzoru Bethego-Blocha,
181 najwyższą zdolność rozdzielczą TPC osiąga dla cząstek o $p_T < 1 \text{ GeV}$.

182 • detektory służące identyfikacji cząstek (ang. *particle identification – PID*)

183 – Transition Radiation Detector (TRD) – detektor wykrywający promieniowanie przej-
184 ścia, służy głównie do odróżniania wysokopędowych ($p_T > 1 \text{ GeV}$) elektronów od
185 pionów. Promieniowanie przejścia emitowane jest podczas przechodzenia relatywi-
186 stycznych cząstek przez granicę ośrodków, jego intensywność jest proporcjonalna do
187 czynnika Lorentza γ , co pozwala na odróżnienie cząstek o tym samym pędzie na pod-
188 stawie różnicy mas (elektryny są ponad 250 razy lżejsze od pionów). TRD oprócz
189 identyfikacji elektronów uczestniczy także w rekonstrukcji śladów wysokopędowych
190 cząstek i może być użyty w systemach wyzwalania (ang. *trigger*).
191 – Time-Of-Flight (TOF) – detektor czasu przelotu o zdolności rozdzielczej $\sim 80 \text{ ps}$.
192 Pozwala na separację pionów i kaonów o pędach do ok. 2.5 GeV i protonów do 4 GeV .
193 – High-Momentum Particle Identification Detector (HMPID) – detektor typu RICH
194 (ang. *ring-imaging Cherenkov*), wykrywający fotony emitowane podczas przejścia
195 przez ośrodek naładowanej cząstki o prędkości większej od prędkości fazowej światła
196 w tym ośrodku (promieniowanie Cherenkowa). Na podstawie kąta pod jakim emitowano
197 są fotony określana jest prędkość cząstki. HMPID pozwala na identyfikację pio-
198 nów, kaonów i protonów o $p_T > 1 \text{ GeV}$. Pokrywa przestrzeń kątów: $1.2^\circ < \phi < 58.8^\circ$
199 oraz $|\eta| < 0.6$ (5% akceptancji TPC).

200 • kalorymetry

202 – Photon Spectrometer (PHOS) – elektromagnetyczny kalorymetr o wysokiej rozdziel-
203 czości energetycznej i przestrzennej (podzielony na kryształy o rozmiarze poprzecznym
204 $2.2 \times 2.2 \text{ cm}$, co odpowiada rozmiarowi w dziedzinie η , ϕ 0.004×0.004). Pokrywa
205 zakres pseudopospieszności $|\eta| < 0.12$ i kąta azymutalnego równy 100° . PHOS ma
206 za zadanie identyfikację fotonów, w szczególności tych niepochodzących z rozpadu
207 innych cząstek (ang. *direct photons*)

208 – EMCal

209 • Muon

210 • detektory przednie

- 211 – ZDC
- 212 – PMD
- 213 – FMD
- 214 – V0
- 215 – T0

216 2 Uczenie maszynowe

217 Uczenie maszynowe jest bardzo szerokim i obecnie dynamicznie się rozwijającym obszarem
218 wiedzy. Występuje w wielu odmianach łącząc w sobie w zależności od wariantu wiele dziedzin
219 takich jak matematyka (statystyka, algebra) informatyka (algorytmika, teoria informacji) a
220 także elementy robotyki i sterowania. Dziedzinami, w których jest najczęściej wykorzystywane
221 są min. widzenie maszynowe, przetwarzanie języka naturalnego, autonomiczne roboty i pojazdy,
222 systemy decyzyjno - eksperckie, optymalizacyjne oraz rekomendacyjne.

223 W tej pracy wykorzystywana jest gałąź uczenia maszynowego nazywana uczeniem nadzorowanym lub "uczeniem z nauczycielem" (ang. *supervised learning*), gdzie uczenie występuje na podstawie poprawnie oznaczonych przykładów. Terminami bliskoznacznymi dla tak rozumianego uczenia maszynowego są uczenie statystyczne (ang. *statistical learning*) i rozpoznawanie wzorców (ang. *pattern recognition*).

228 Problem identyfikacji dżetów jest klasycznym przykładem zagadnienia klasyfikacji, gdzie
229 poprawna odpowiedź jest jedną ze skończonej ilości opcji (klas) w przeciwieństwie do regresji,
230 gdzie szukana odpowiedź algorytmu ma charakter ciągły.

231 Występuje wiele algorytmów uczenia maszynowego takich jak regresja liniowa i logistyczna,
232 drzewa decyzyjne i ich wariacje, maszyny wektorów wspierających, sztuczne sieci neuronowe
233 oraz wiele innych. Uczenie polega na znalezieniu pewnej funkcji dopasowującej do przyjmowanego na wejściu zestawu (wektora) cech (zmennych, kolumn) pewną odpowiedź (predykcję),
234 która minimalizuje zadaną funkcję straty. Jej rolę w przypadku regresji często pełni błąd średniokwadratowy a w przypadku klasyfikacji np. entropia krzyżowa (ang. *cross entropy*)¹. Różne
235 algorytmy szukają przy tym funkcji dopasowującej należącej do różnych klas funkcji: przykładowo klasyczne drzewa decyzyjne przeszukują tylko przestrzeń funkcji dających się opisać
236 skończonym zbiorem reguł "jeśli – to" (ang. *if – else*).

240 W pracy wykorzystane zostały dwa rodzaje algorytmów: wzmacniane drzewa decyzyjne oraz
241 sieci neuronowe.

242 2.1 Wzmacniane drzewa decyzyjne

243 Wzmacniane drzewa decyzyjne są jednym z rozwinięć klasycznego algorytmu drzewa decyzyjnego. Pojedyncze drzewo decyzyjne dzieli przestrzeń cech uczących przy pomocy prostopadłych cięć, na mniejsze/większe niż zadana wartość w przypadku zmiennej ciągłej lub na należące/nie należące do danej klasy w przypadku zmiennej kategorycznej. Każdy podział, nazywany węzłem, daje dwie gałęzie, które można dalej niezależnie dzielić aż do ostatniego poziomu (liści). Kolejne podziały wybierane są tak, aby zbiory przykładów wpadające do poszczególnych gałęzi były jak najbardziej jednorodne. Stosuje się różne miary nieporządku takie tak: indeks Gini G_L lub entropia S_L ².

251 Drzewa decyzyjne są często łączone w komitety klasyfikatorów (ang. *ensemble methods*).
252 Wiele "słabych" klasyfikatorów jest łączonych w jeden "silny" na dwa sposoby: workowanie
253 (ang. *bagging*) oraz wzmacnianie (ang. *boosting*), które są często ze sobą porównywane.

254 *Bagging* – w zastosowaniu dla drzew decyzyjnych nazywany algorytmem lasów losowych
255 (ang. *random forest*) - polega na wytrenowaniu wielu drzew, każdego na podstawie N przykładów losowo wylosowanych z powtórzeniami spośród N -licznego zbioru treningowego. Dodatko-

¹ $J = -\frac{1}{N} \sum_{i=1}^N [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)]$, gdzie y_i to prawidłowa klasa i -tego przykładu a \hat{y}_i to predykcja algorytmu

² $G_L = 1 - \sum_k p_k^2$ oraz $S_L = \sum_k -p_k \log p_k$, gdzie p_k to stosunek liczby przypadków klasy k do liczby wszystkich przypadków w liściu L

257 wo, do uczenia każdego drzewa używa się tylko podzbioru wszystkich cech uczących. Końcową
258 predykcję algorytmu otrzymuje się poprzez "głosowanie" wszystkich drzew z odpowiednimi
259 wagami.

260 *Boosting* – wzmacniane drzewa decyzyjne (ang. *boosted decision trees*) – jest metodą po-
261 dobną do *baggingu*. Główną różnicą jest zwiększanie wag przykładom uczącym, które przez po-
262 przednie drzewo zostały źle zaklasyfikowane – każde kolejne drzewo koncentruje się bardziej na
263 poprawie błędów poprzednich drzew. Widać tu kolejną ważną cechę odróżniającą obie metody:
264 *boosting* jest algorytmem sekwencyjnym podczas gdy *bagging* daje się trywialnie zrównoleglić
265 (każde drzewo trenowane jest w osobnym wątku).

266 Parametry i sposób trenowania drzew decyzyjnych na analizowanych danych

267 W niniejszej pracy wykorzystano wzmacniane drzewa decyzyjne zaimplementowane w wydaj-
268 nej bibliotece **XGBoost** [10]. Szybkość obliczeń jest bardzo ważna, gdyż oprócz komfortu pracy
269 z algorytmem, przekłada się na jakość otrzymanych wyników – krótszy czas obliczeń oznacza
270 możliwość przeprowadzenia większej ilości eksperymentów i lepsze dobranie parametrów oraz
271 danych. Implementacja wzmacnianych drzew decyzyjnych w **XGBoost** wykorzystuje wszystkie
272 rdzenie procesora, pomimo że sam algorytm ma charakter sekwencyjny – jest to możliwe dzięki
273 paralelizacji procesu tworzenia każdego drzewa (przed każdym podziałem konieczne jest spraw-
274 dzenie pewnej ilości możliwych zmiennych i wartości progowych i ten proces jest wykonywany
275 równolegle).

276 Dzięki szybkiemu uczeniu się algorytmu, możliwe było użycie kosztownego obliczeniowo au-
277 tomatycznego przeszukiwania przestrzeni parametrów przy pomocy przeszukiwania losowego
278 (ang. *random search*), które jest zwykle preferowane nad przeszukiwanie sieciowe [11]. W tym
279 celu cały zbiór danych dzielony był na dwie części: trenującą oraz testową (80/20%). Następ-
280 nie algorytm był trenowany i oceniany z użyciem trzy- lub pięciokrotnej walidacji krzyżowej
281 (ang. *cross-validation*) na zbiorze trenującym dla różnych zestawów parametrów. Model z naj-
282 lepszym wynikiem uzyskanym w walidacji krzyżowej był sprawdzany na zbiorze testowym.

283 Parametry optymalizowane w opisanym procesie to:

- 284 • *max_depth* – maksymalna głębokość każdego drzewa (niekoniecznie osiągana)
- 285 • *n_estimators* – liczba drzew
- 286 • *learning_rate* – parametr szybkości uczenia, komplementarny do *n_estimators*, w prak-
287 tyce można ustalić liczbę drzew i szukać optymalnej szybkości uczenia
- 288 • *subsample*, *colsample_bytree*, *colsample_bylevel* – parametry regularyzacyjne określają-
289 ce ułamek kolejno: danych użytych do trenowania każdego drzewa, kolumn użytych w
290 każdym drzewie (cechy losowane raz dla danego drzewa), kolumn użytych przy każdym
291 podziale (cechy losowane przy każdym podziale)
- 292 • γ – minimalny zysk w postaci zmniejszenia wartości funkcji straty konieczny do wykonania
293 podziału

294 2.2 Sieci neuronowe

295 Sieci neuronowe (ang. *neural networks* – *NN*) są szczególnym algorytmem uczenia maszyno-
296 wego. Występują w bardzo wielu odmianach i są wykorzystywane w rozwiązywaniu szerokiej
297 gamy problemów. Nawet bardzo pobiczny opis sieci neuronowych wymaga dużo więcej miejsca
298 niż może być temu poświęcone w tej pracy. Wprowadzenia do sieci neuronowych na różnym

299 poziomie zaawansowania można znaleźć m.in. w [#REF]x100. Tu przedstawione zostaną wy-
300 łącznie wybrane zagadnienia mające ścisłyjszy związek z pracą. Używane mogą być terminy,
301 których znaczenie wyjaśniane jest w podanych źródłach.

302 W niniejszej pracy, wykorzystane zostały dwa rodzaje sieci neuronowych: sieci w pełni połą-
303 czone (ang. *fully connected NN – FC NN*), nazywane także wielowarstwowymi perceptronami
304 (ang. *multi-layer perceptron – MLP*) oraz sieci konwolucyjne (ang. *convolutional NN – Co-*
305 *nvNets, CNN*).

306 Sieci w pełni połączone

307 W nierekurencyjnych sieciach neuronowych (tylko takie są używane w tej pracy), informacja
308 jest przekazywana kolejno od warstw wejściowych, poprzez warstwy ukryte aż do wyjściowej. W
309 sieciach typu *FC* wszystkie warstwy składają się z identycznych neuronów – każdy neuron do-
310 staje na wejściu wektor, natomiast zwraca skalar – wartość pewnej zadanej, nieliniowej funkcji,
311 jako argument podając średnią ważoną z elementów wektora wejściowego. Wartości zwraca-
312 ne przez neurony w danej warstwie składają się na wektor wejściowy dla neuronów kolejnej
313 warstwy. Wejściem dla pierwszej warstwy są natomiast kolejne przykłady ze zbioru uczącego.
314 Trenowanie sieci neuronowych polega na zmienianiu wag (parametrów) w liczonej w każdym
315 neuronie średniej, każdy neuron posiada własny, niezależny zestaw wag.

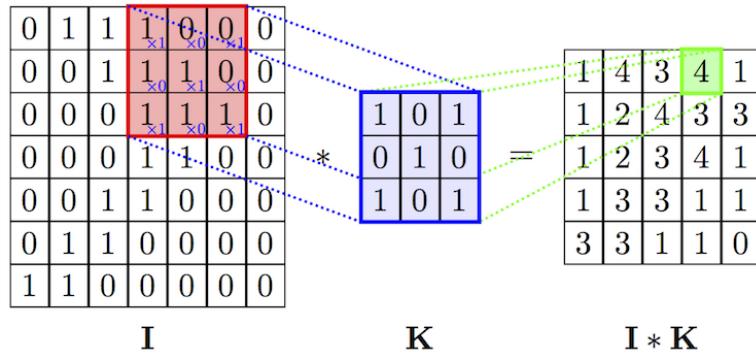
316 Istnieje twierdzenie o sieciach neuronowych jako uniwersalnych aproksymatorach funkcji
317 (ang. *universal approximation theorem*) [12], mówiące, że już sieć neuronowa o jednej warstwie
318 ukrytej jest zdolna do przybliżenia dowolnej funkcji z dowolną dokładnością. Twierdzenie to
319 nie podaje niestety liczby potrzebnych neuronów a przede wszystkim – sposobu ich trenowania.
320 Trenowanie jest prostsze w przypadku zastosowania wielu warstw, które odpowiadają kolejnym
321 poziomu abstrakcji jednak nadal jest dużym wyzwaniem ze względu na fakt, że nawet stosun-
322 kowo niewielka sieć może posiadać bardzo dużą liczbę parametrów, przykładowo sieć o czterech
323 warstwach, w każdej po 128 neuronów ma ich ponad 65 tysięcy.

324 Sieci konwolucyjne

325 Jednym ze sposobów na ograniczenie liczby trenowanych parametrów jest użycie konwolucyj-
326 nych sieci neuronowych (bardziej poprawną choć rzadko używaną nazwą w języku polskim jest
327 sieć splotowa). Są one inspirowane połączeniami w korze wzrokowej zwierząt i wywodzą się z ba-
328 dań w obszarze widzenia komputerowego, gdzie liczby parametrów są szczególnie duże (wektor
329 wejściowy ma wymiar równy liczbie pikseli w obrazie), na takim przykładzie również najłatwiej
330 zrozumieć ich działanie.

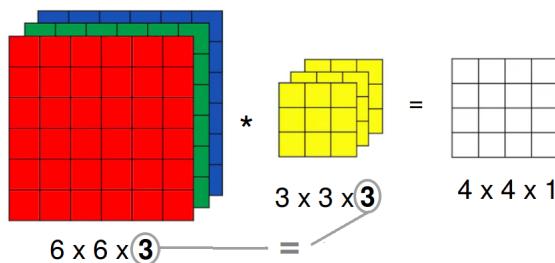
331 Sieci konwolucyjne różnią się od sieci typu *FC* tym, że część wag połączeń między warstwami
332 jest dzielona. Występuje w nich nowy rodzaj warstwy, nazywany warstwą konwolucyjną. Każda
333 jednostka w warstwie konwolucyjnej (filtr) ma pewną stałą (niewielką) liczbę wag. Połączenie z
334 dużym wejściem realizowane jest przez powielanie tych samych wag w połączeniach z kolejnymi
335 fragmentami wektora wejściowego (por. Rys. 4). Rezultatem działania filtra na macierz jest wy-
336 nik operacji splotu. Liczba parametrów przypadająca na każdy filtr jest równa jego rozmiarowi
337 i nie zależy od wielkości wektora wejściowego.

338 W przypadku gdy zamiast wejścia dwuwymiarowego (jak np. obraz czarno-biały), mamy do
339 czynienia z wejściem trójwymiarowym (np. trzeci wymiar to kolejne kolory w kodowaniu RGB),
340 filtry również muszą mieć trzy wymiary, przy czym rozmiar w ostatnim wymiarze musi być
341 równy rozmiarowi w tym kierunku wektora wejściowego. Wynik operacji splotu jest ponownie
342 dwuwymiarowy, gdyż filtr przesuwany jest tylko w dwóch pierwszych wymiarach. Trzeci wymiar
343 powstaje przez składanie kolejnych filtrów. Widać zatem, że również w przypadku gdy na

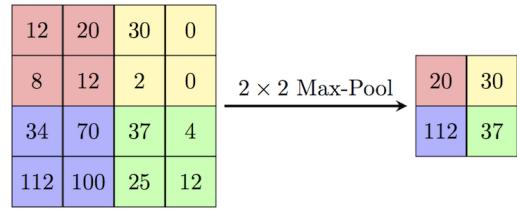


Rysunek 4: Schemat działania pojedynczego filtra z warstwy konwolucyjnej (operacja splotu).

344 wejścia podawany jest obraz czarno-biały, filtry w kolejnych warstwach konwolucyjnych (oprócz
345 pierwszej) mają po trzy wymiary.



Rysunek 5: Działanie pojedynczego filtra (3D) na wejście o trzech wymiarach.



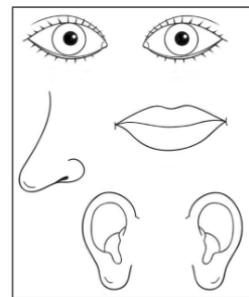
Rysunek 6: Działanie warstwy typu *max-pool*

346 Oprócz warstw konwolucyjnych, w sieciach tego typu stosowane są także tzw. warstwy typu
347 *max-pooling*. Zasada jej działania jest bardzo prosta: wykonuje funkcję *maksimum* na zadanym
348 fragmencie obrazu (por. Rys. 6). Ich rolą jest zmniejszanie rozmiaru przekazywanej w sieci
349 informacji.

350 Typowa architektura stosowana w przypadku sieci konwolucyjnych jest następująca: naj-
351 pierw warstwy konwolucyjne (pomiędzy nimi czasem warstwy typu *max-pool*), następnie wszyst-
352 kie filtry są rozwijane i składane w długi jednowymiarowy wektor, który przekazywany jest do
353 warstw typu *FC*. W przypadku problemu klasyfikacji, na końcu znajduje się jeszcze warstwa
354 typu *softmax* normalizująca wyjście z sieci do jedynki (por. Rys. 7).

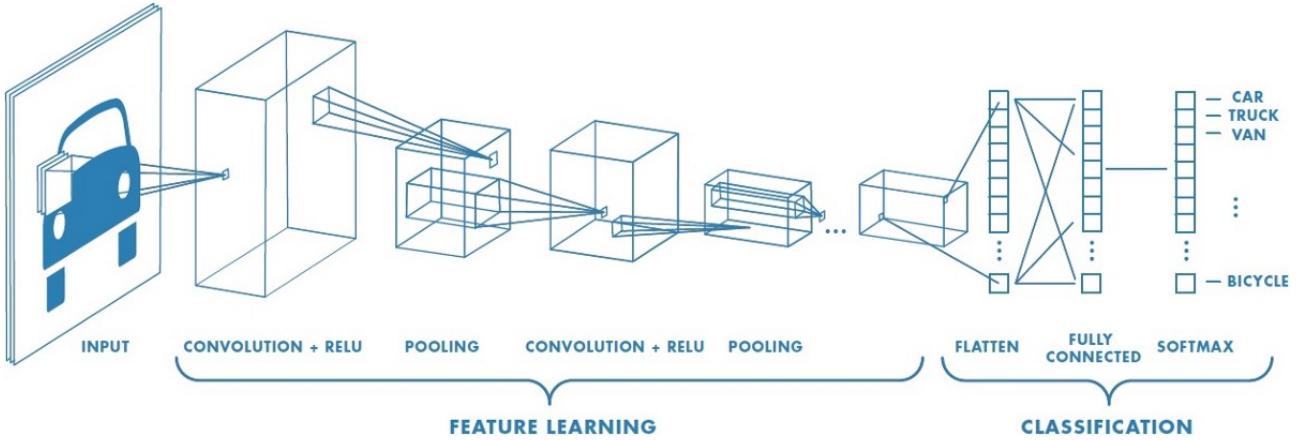
355 Sieci konwolucyjne posiadają dwie właściwości odróżniające je od
356 *MLP*:

- 357 • niezmienniczość względem przesunięcia (ang. *translation invariance*) – głównie za sprawą dzielenia wag oraz obecności warstw
358 typu *max-pool*, położenie danej cechy na obrazie jest niemal
359 bez znaczenia (obraz po prawej stronie byłby rozpoznany ja-
360 ko twarz)
- 362 • lokalność połączeń – filtry obejmują tylko kilka sąsiednich pikseli
363 (tam zwykle występują najsilniejsze zależności) nie są w stanie
364 dostrzec cechy rozciągniętej na obszar większy od rozmiaru filtra



365 Hiperparametry i trenowanie sieci neuronowych na analizowanych danych

366 Parametry sieci, których wartości są określone przez projektanta sieci, takie jak liczba warstw
367 ukrytych są nazywane hiperparametrami (dla odróżnienia od parametrów - wag połączeń).

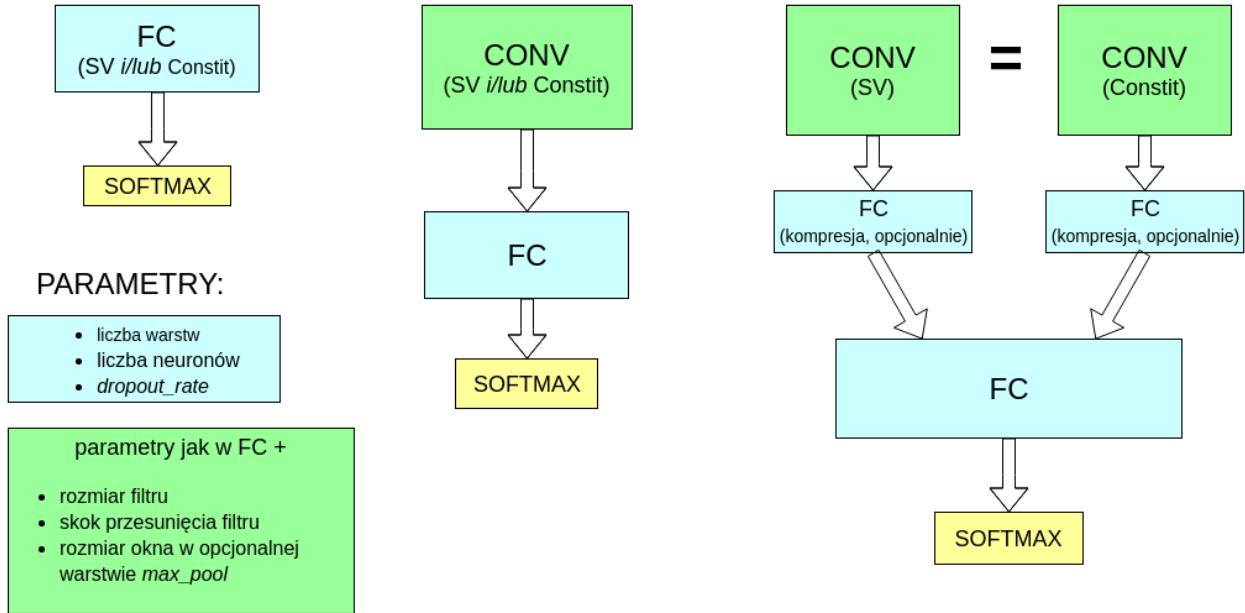


Rysunek 7: Typowa struktura stosowana w sieciach konwolucyjnych. Warstwy konwolucyjne mają za zadanie wydobywać cechy, na podstawie których późniejsze warstwy dokonują klasyfikacji. Widoczna jest charakterystyczna stopniowa zmiana rozmiaru przekazywanej macierzy: rozmiar poprzeczny maleje kosztem głębokości, co odpowiada rosnącej liczbie filtrów i malejącym rozmiarowi obszaru po jakim są one przesuwane.

368 Testowane były trzy architektury: sieci w pełni połączone, sieci konwolucyjne oraz sieć
 369 złożona z dwóch gałęzi, osobnych dla wtórnych wierzchołków i częstek tworzących dżet (por.
 370 Rozdz. 3) przedstawione schematycznie na Rys. 8. Do trenowania sieci wykorzystano wysoko-
 371 poziomową bibliotekę Keras [13] korzystającą z silnika obliczeniowego zaimplementowanego w
 372 TensorFlow [14].

373 Zestaw hiperparametrów definiujący działanie sieci w pełni połączonej:

- 374 • liczba warstw ukrytych
- 375 • liczba neuronów w każdej warstwie
- 376 • funkcja aktywacji – nielinowa funkcja aplikowana przed zwróceniem wartości w każdym
 377 neuronie, najpopularniejsze to *tanh*, *ReLU* ($f(x) = \max(0, x)$) oraz funkcja sigmoidalna
 378 ($f(x) = \frac{1}{1+\exp(x)}$)
- 379 • algorytm optymalizacyjny – spadek gradientowy lub jego wariacje
- 380 • parametr szybkości uczenia i jego modyfikacje w trakcie uczenia
- 381 • liczba przykładów trenujących przetwarzanych w jednym kroku uczenia (ang. *batch_size*)
 382 – im większy tym szersze jest trenowanie sieci (dzięki wydajnym operacjom macierzo-
 383 wym), natomiast może się to odbywać kosztem precyzji
- 384 • liczba epok uczenia – ile razy będzie pokazywany sieci każdy przykład
- 385 • opcjonalnie: warunki stopu (ang. *early stopping*) w razie osiągnięcia *plateau* (wysycenia
 386 procesu uczenia)
- 387 • opcjonalnie: regularyzacja przy pomocy różnych technik (zwykle konieczna)
- 388 • ponadto dla sieci konwolucyjnych:
 - 389 – liczba warstw konwolucyjnych i liczba filtrów w każdej warstwie
 - 390 – obecność lub brak warstw *max-pool* i rozmiar ich okna



Rysunek 8: Schematyczne przedstawienie trzech testowanych rodzin architektur sieci. Każdy blok odpowiada kilku warstwom danego typu. Bloki warstw typu *FC* i opisane jako "kompresja" składały się z kilku neuronów i miały za zadanie zredukować całą informację z danej gałęzi do wektora kilku liczb. Obie gałęzie miały taką samą strukturę.

391 – rozmiar filtrów i długość skoku przy ich przesuwaniu

392 Same dwie pierwsze wielkości dają nieograniczoną liczbę konfiguracji. Czas trenowania sieci
 393 neuronowych jest rzędu wielkości większej niż drzew decyzyjnych, dlatego przyjęto szereg kro-
 394 ków mających na celu zmniejszenie przeszukiwanej przestrzeni hiperparametrów. Na podstawie
 395 wstępnych testów oraz różnych wskazówek dostępnych w literaturze przyjęto:

- 396 • *batch_size* zawsze równy 64 (inne testowane wartości: 16, 32, 128)
- 397 • za algorytm optymalizacyjny przyjęto algorytm o nazwie *Nadam* [15], tj. rozwinięcie al-
 398 gorytmu *Adam* [16] o parametr Nesterova (inne testowane to zwykły spadek gradientowy
 399 oraz *Adam*)
- 400 • funkcję aktywacji: *ReLU*
- 401 • liczba epok równa 100 lub 200 (lub mniej do testów), zrezygnowano z *early stopping*
- 402 • stałe w trakcie treningu wartości parametru szybkości uczenia
- 403 • spośród technik regularizacyjnych testowano wyłącznie *dropout* [17] z parametrami 0.1,
 404 0.2, 0.5
- 405 • kilka wybranych kombinacji dla zestawu parametrów: rozmiar filtra, długość skoku i roz-
 406 miar okna w warstwach *max-pool* – takie same w kolejnych warstwach
- 407 • liczby neuronów/filtrów w warstwach będące zawsze potęgami dwójki oraz stałą liczbę w
 408 kolejnych warstwach lub zmieniającą się o stały czynnik, np. 256-128-64, 128-128-128 lub
 409 16-32-64
- 410 • liczba warstw *FC*: 2-8, konwolucyjnych 2-6

411 Nawet po przyjęciu powyższych uproszczeń nie sposób sprawdzić wszystkich możliwych
412 zestawów hiperparametrów, dlatego sposób ich dobierania w kolejnych testach był mocno em-
413 piryczny. Dostępne dane dzielone były na trzy zbiory: trenujący, walidacyjny i testowy. Wobec
414 braku warunków stopu, zbiór walidacyjny użyty był wyłącznie do porównywania różnych ze-
415 stawów parametrów, tak aby wynik testowy pozostał nieobciążony.

416 Zgodnie z zasadą ortogonalizacji działań, proces doboru hiperparametrów dzielono na dwie
417 części: najpierw starano się uzyskać jak najlepsze wyniki na zbiorze uczącym, a dopiero później
418 zmusić algorytm do lepszej generalizacji na zbiorze testowym przez zwiększoną regularyzację i
419 modyfikację parametru szybkości uczenia.

420 **2.3 Dyskusja użycia dwóch algorytmów**

421 Użycie więcej niż jednego algorytmu ma wiele zalet. Po pierwsze daje możliwość porównania
422 wyników. Pozwala to na oszacowanie błędu *Bayesowskiego* (najniższego możliwego do osiągnię-
423 cia przez jakikolwiek algorytm błędu). Jest to bardzo ważne w sytuacji, gdy nie dysponuje
424 się innym oszacowaniem tego błędu (w wielu problemach naturalnych dla człowieka jak roz-
425 poznawanie obiektów na obrazkach jest nim błąd ludzki lub też błąd popełniany przez zespół
426 ekspertów w bardziej zaawansowanych zastosowaniach).

427 Po drugie, wykorzystane zostały dwa algorytmy mocno różniące się w swojej naturze, co po-
428 zwala wykorzystać cechy każdego z nich w analizie: przykładowo sieci neuronowe dobrze radzą
429 sobie z niestrukturyzowanymi danymi – potrafią tworzyć wysoko poziomowe cechy na podsta-
430 wie niskopoziomowego wejścia (np. położenia oka na zdjęciu twarzy na podstawie pixeli). Są
431 natomiast trudne w interpretacji i często traktowane są jako tzw. "czarne skrzynki" (ang. *black*
432 *box*). Oprócz tego, liczba możliwych konfiguracji sieci jest ogromna i przez to niemożliwe jest
433 stwierdzenie czy wykorzystane zostały pełne ich możliwości.

434 Z kolei drzewa decyzyjne posiadają stosunkowo niewielką liczbę parametrów, a ich trenowa-
435 nie jest bardzo szybkie co pozwala na ich ekstensywne przeszukiwanie i otrzymanie wyników,
436 które można uznać za optymalne dla tego algorytmu. Ponadto, w przypadku drzew istnieją
437 niewymagające dodatkowych obliczeń miary użyteczności poszczególnych zmiennych, co daje
438 wgląd w działanie algorytmu i poprawia intuicyjne zrozumienie jego predykcji.

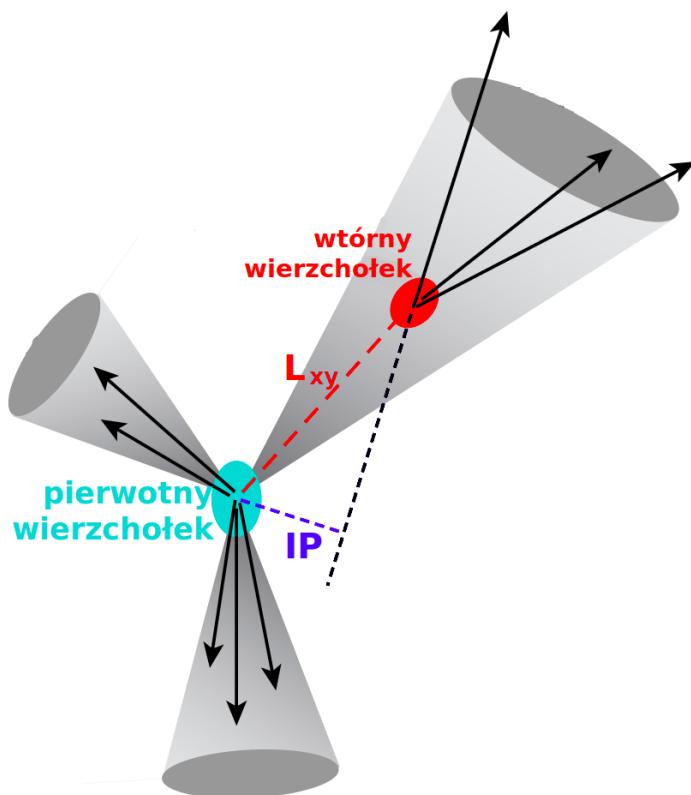
439 3 Dane

440 Dane użyte w analizie pochodzą z symulacji Monte Carlo zderzeń proton-proton przy energii
441 w układzie środka masy równej $\sqrt{s} = 13$ TeV dostępnych na serwerach eksperymentu ALICE.
442 Są to pełne symulacje detektora ALICE, wykorzystujące generator zderzeń Pythia8 [18] (*tune:*
443 *Pythia8Jets_Monash2013* [19]) oraz pakiet Geant3 [20] do transportu cząstek przez detektor.

444 Do rekonstrukcji dżetów wykorzystany został algorytm *anti-kt* z parametrem $R = 0.4$ za-
445 implementowany w pakiecie FASTJET [21]. Dżetów poszukiwano tylko wyłącznie wśród cząstek
446 naładowanych (ang. *charged jets*) ze względu na słabe pokrycie przestrzeni fazowej przez ka-
447 lorymetry w eksperymencie ALICE.

448 Do analizy wybrano dżety o p_T większym niż 15 GeV i mieszczących się w całości w akcep-
449 tancji detektora *TPC*, tj. $|\eta| < 0.9$, co przy użytym parametrze rozmiaru dżetu $R = 0.4$, daje
450 ograniczenie na pseudopospieszność $|\eta| < 0.5$ dla osi dżetu.

451 Dla każdego dżetu obliczony został szereg wielkości, które można podzielić na zmienne
452 na poziomie dżetu, związane z wtórnymi wierzchołkami oraz cząstkami tworzącymi dżet. Za
453 potencjalne wtórne wierzchołki uznaje się wszystkie kombinacje trzech cząstek spełniających
454 pewne dosyć luźne kryteria jak $p_T > 0.15$ GeV (rozważane są wyłącznie trzy-cząstkowe wtórne
455 wierzchołki), stąd ich liczba może być dużo większa od liczby cząstek tworzących dżet.



457 Rysunek 9: Rysunek ilustrujący znaczenie używanych wielkości: L_{xy} oraz parametru zderzenia
458 IP .

456 Lista używanych zmiennych:

457 • Zmienne na poziomie dżetu:

458 – η, ϕ – pseudopospieszność (ang. *pseudorapidity*) i kąt azymutalny

- p_T – pęd poprzeczny dżetu
- masa dżetu
- powierzchnia dżetu – liczona w płaszczyźnie (η, ϕ) , do powierzchni dżetu zaliczany jest element w powierzchni w którym dodanie cząstki o nieskończonym małym pędzie poprzecznym sprawi, że zostanie ona zaliczona do tego dżetu [#REF <https://arxiv.org/pdf/0707.0952.pdf>]
- gęstość tła (w danym zdarzeniu)
- N_{SV} – liczba wtórych wierzchołków
- $N_{Constit}$ – liczba cząstek tworzących dżet

• Zmienne opisujące cząstki tworzące dżet:

- η, ϕ – pseudopospieszność i kąt azymutalny cząstki względem osi dżetu
- p_T – pęd poprzeczny cząstki
- IP_D – rzut na kierunek poprzeczny wektora parametru zderzenia
- IP_Z – rzut na oś z wektora parametru zderzenia

• Zmienne opisujące wtórne wierzchołki:

- L_{xy} – odległość między pierwotnym a wtórnym wierzchołkiem (ang. *decay length*)
- $\sigma_{L_{xy}}$ – niepewność wyznaczenia L_{xy}
- $\sigma_{vertex} = \sqrt{d_1^2 + d_2^2 + d_3^2}$ – rozrzut śladów (ang. *tracks*) wokół wtórnego wierzchołka, gdzie d_i to odległość najbliższego zbliżenia śladu / odległość najbliższego przelotu do wtórnego wierzchołka (ang. *distance of closest approach – DCA*)
- $M_{inv} = \sqrt{(E_1 + E_2 + E_3)^2 - (\vec{p}_1 + \vec{p}_2 + \vec{p}_3)^2}$ – masa niezmiennica wierzchołka, gdzie E_i, p_i to energia i pęd i -tej cząstki tworzącej wierzchołek
- χ^2/Ndf dopasowania wtórnego wierzchołka

TBD: tabelaryczna postać, liczba SV i Nconstit, jak posortowane, wypełniania zerami korespondencja między użytymi danymi a algorytmami z sekcji "identyfikacja dżetów b"

4 Analiza

4.1 Dobór metryki

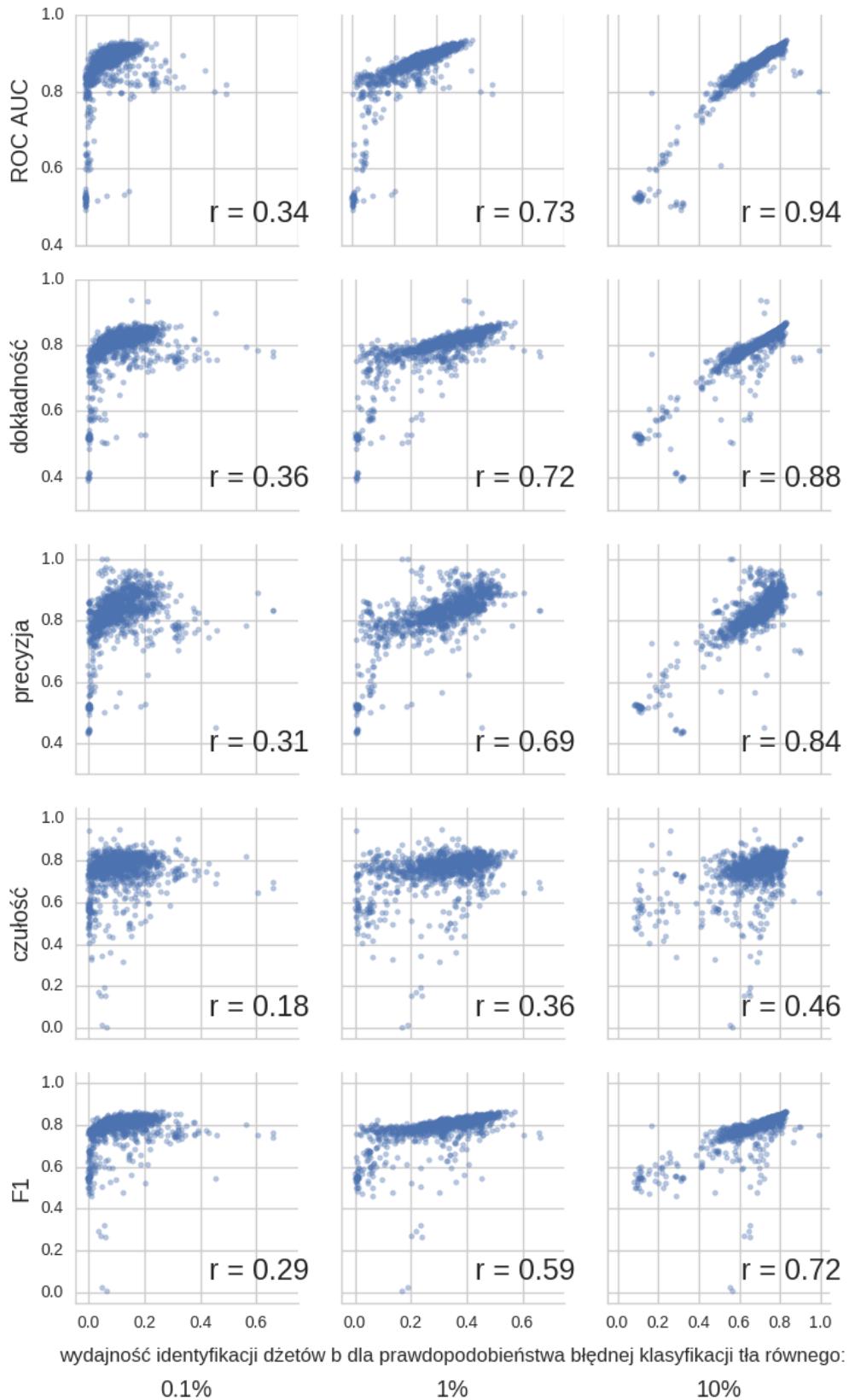
Bardzo ważnym elementem w trenowaniu algorytmów uczenia maszynowego jest dobór odpowiedniej metryki – klasycznym złym przykładem jest używanie dokładności (ang. *accuracy*) do oceniania klasyfikacji binarnej w przypadku dużego niezrównoważenia klas – algorytm przewidujący zawsze klasę większościową może osiągnąć dużą wartość dokładności będąc jednocześnie bardzo słabym modelem.

Kilka najczęściej używanych metryk wymieniono w Tab. ???. Używanie i porównywanie kilku miar efektywności jest często niepraktyczne dlatego dobrze jest wybrać jedną metrykę. Przy jej wyborze należy kierować się potencjalnymi zastosowaniami modelu. W tym przypadku są to analizy fizyczne, które mogą mieć różne wymagania dotyczące czystości i liczebności otrzymywanych próbek a co za tym idzie, preferować inne punkty pracy zdefiniowane jako pary liczb: wydajność poprawnej klasyfikacji dżetów b (ang. *tagging efficiency = true positive rate = recall*), i ułamek niepoprawnie zaklasyfikowanych przypadków tła (ang. *mistagging rate = false positive rate*).

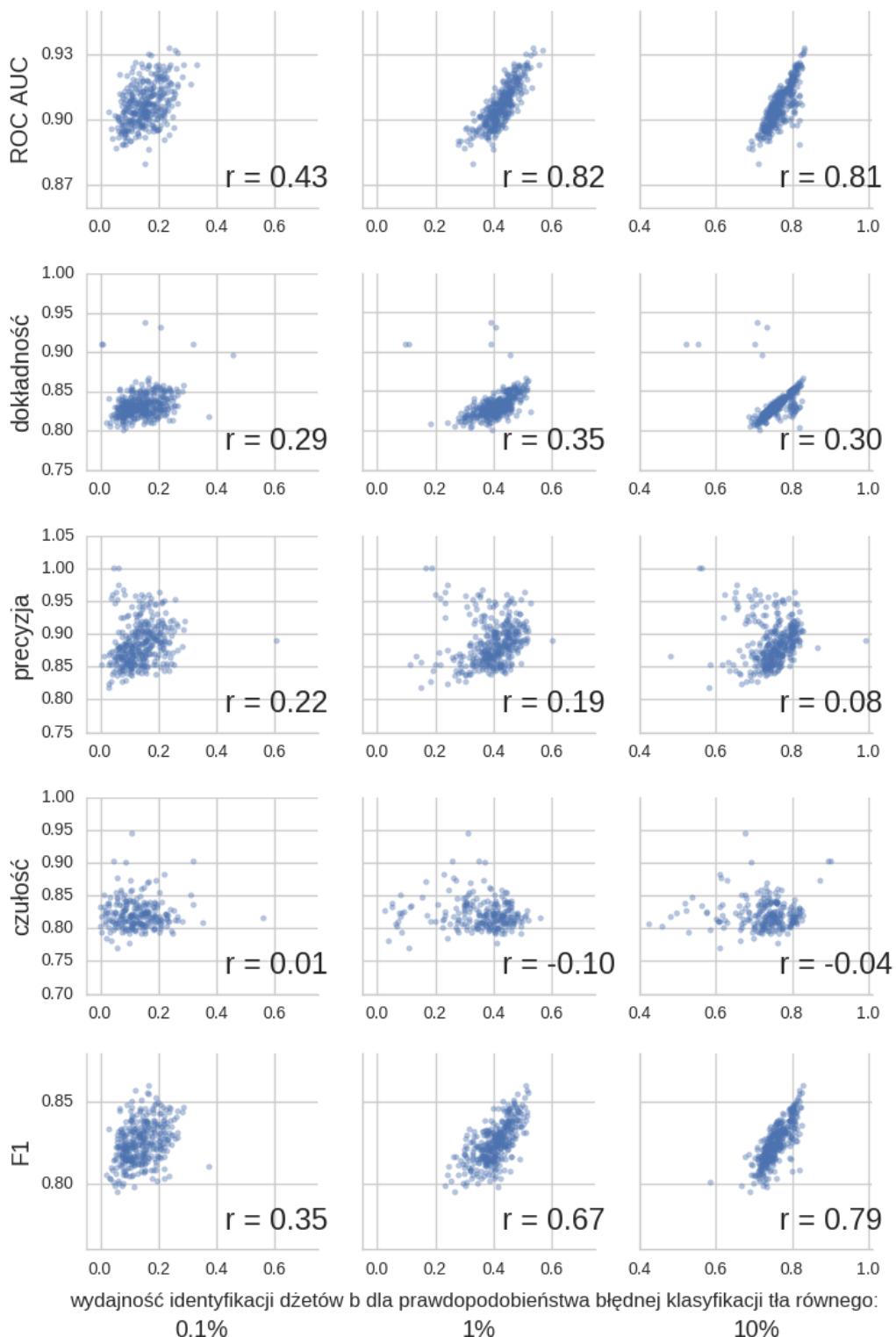
Naturalnym wyborem wydaje się pole pod powierzchnią krzywej ROC (ang. *ROC Area Under Curve – ROC AUC*). Potencjalną przeszkodą może być zakres rozsądnych wartości prawdopodobieństwa błędnej klasyfikacji przypadków tła: dżety b stanowią tylko kilka procent liczby wszystkich dżetów, zatem z punktu widzenia analizy dopuszczalne będą punkty pracy zapewniające wydajność identyfikacji dżetów b ok. 10 – 100 razy większą niż częstość niepoprawnego zaklasyfikowania przypadków tła. Oznacza to, że zdecydowana większość punktów pracy znajdujących się na krzywej ROC jest nie do zaakceptowania – interesujące są tylko te o najniższych częstotliwościach.

Aby ilościowo porównywać różne algorytmy wprowadzone zostaną trzy punkty pracy: o prawdopodobieństwie błędnej klasyfikacji tła równej 0.1%, 1% oraz 10%. Na Rys. 14 przedstawione zostały zależności poszczególnych metryk od wydajności identyfikacji dżetów b w tych trzech punktach pracy. Każdy punkt odpowiada jednemu eksperymentowi (dla dowolnego algorytmu) przeprowadzonemu w trakcie przygotowywania analizy. Daje to pogląd, używanie której metryki zapewni jednocześnie wysokie wartości wydajności na identyfikację dżetów b w wybranych punktach pracy. Najwyższe koreacje występują dla punktu pracy o najwyższym prawdopodobieństwie błędnej klasyfikacji tła – jak będzie to pokazane później wyniki dla tego punktu pracy są najbardziej stabilne. Spośród analizowanych metryk najwyższe wartości współczynnika Pearsona otrzymano dla pola pod powierzchnią krzywej ROC, dosyć wysokie również dla dokładności i precyzji. Widać, że wartości czułości są najsłabiej skorelowane z wydajnością identyfikacji dżetów b – jest zrozumiałe, że są one dużo niższe niż dla precyzji: wartości czułości maleją gdy błędnie klasyfikowane są dżety b stanowiące sygnał, podczas gdy wartości precyzji maleją, gdy błędnie klasyfikowane są przypadki tła. Z tych dwóch błędów, drugi jest bardziej kosztowny, gdyż błędna klasyfikacja nawet niewielkiej części tła znacznie pogarsza czystość otrzymywanej próbki.

Na Rys. 11 przedstawiono te same wykresy, ale tym razem wybrano tylko 25% najlepszych wartości dla każdej metryki – te punkty są bardziej znaczące, gdyż ostatecznie modele z eksperymentów dających najlepsze wyniki będą używane. Dla tych wykresów otrzymano zdecydowaną dominację *ROC AUC* – wybór modeli dających najwyższe pole pod powierzchnią krzywej ROC zapewnia jednocześnie otrzymanie wysokich wartości wydajności identyfikacji dżetów b dla wybranych punktów pracy.

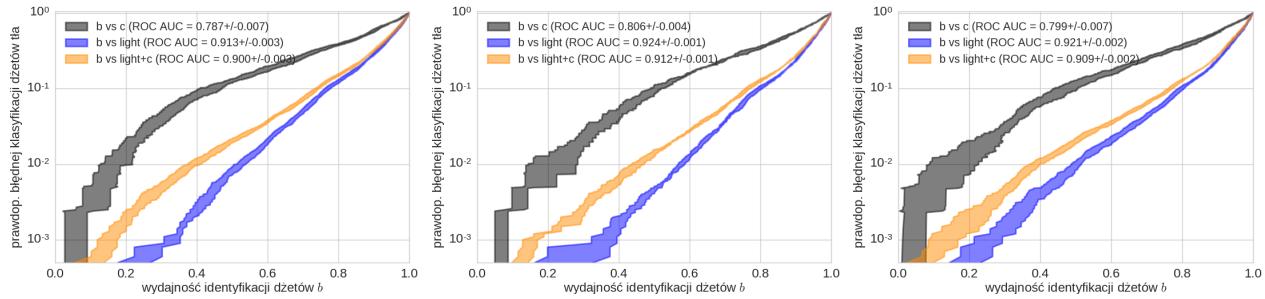


Rysunek 10: Zależność podstawowych metryk od wydajności identyfikacji dżetów b dla punktów pracy o prawdopodobieństwie błędnej klasyfikacji tła równej 0.1%, 1% oraz 10%. Dla każdego wykresu przedstawiono współczynnik korelacji r Pearsona.

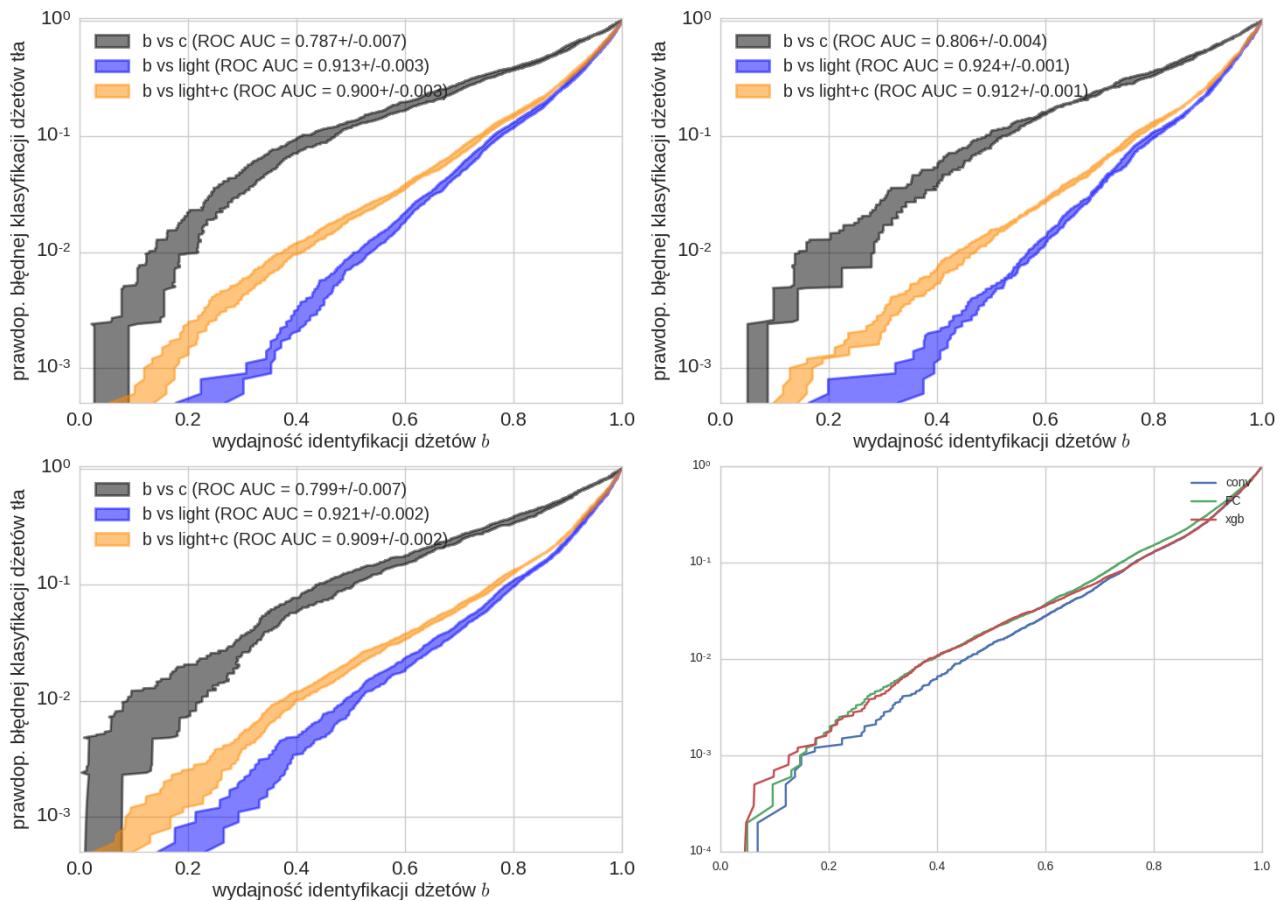


Rysunek 11: Rysunek podobny do Rys. 14, ale przedstawione zostały tylko punkty odpowiadające eksperymentom o wartościach metryki będących w górnym kwartylu wartości danej metryki dla wszystkich eksperymentów.

528 4.2 Wyniki dla zmiennych SV

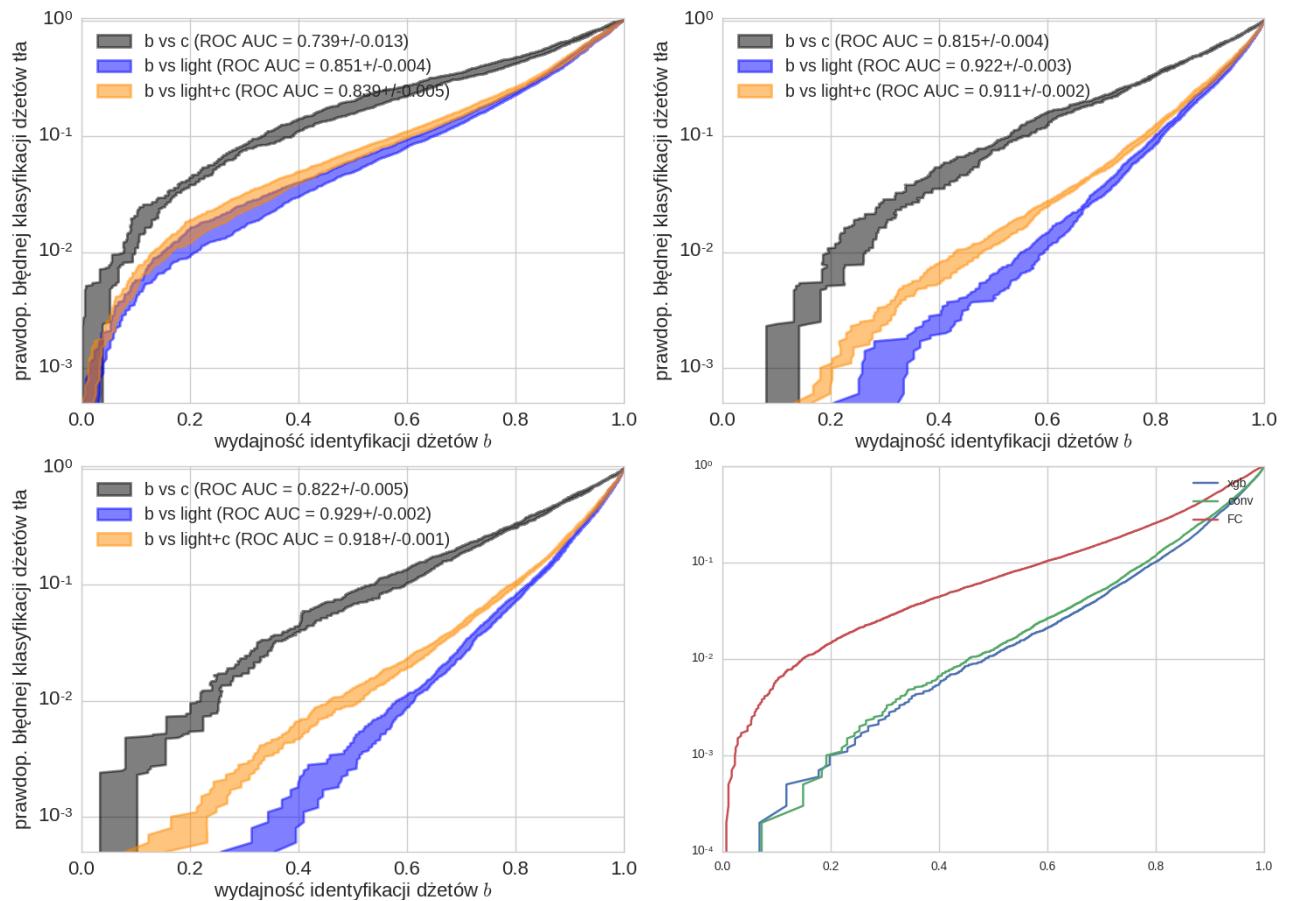


Rysunek 12: Zależności



Rysunek 13: Zależności

529 4.3 Wyniki dla zmiennych constit



Rysunek 14: Zależności

530 References

- 531 [1] Donald H. Perkins. "Oddziaływanie międzykwasowe i chromodynamika kwantowa". In:
532 *Wstęp do Fizyki Wysokich Energii*. 2nd ed. PWN, 2005, 171–193.
- 533 [2] Anton Andronic. "An overview of the experimental study of quark-gluon matter in high-
534 energy nucleus-nucleus collisions". In: *Int. J. Mod. Phys. A* 29 (2014), p. 1430047. DOI:
535 10.1142/S0217751X14300476. arXiv: 1407.5003 [nucl-ex].
- 536 [3] Vardan Khachatryan et al. "Evidence for collectivity in pp collisions at the LHC". In:
537 *Phys. Lett. B* 765 (2017), pp. 193–220. DOI: 10.1016/j.physletb.2016.12.009. arXiv:
538 1606.06198 [nucl-ex].
- 539 [4] Jaroslav Adam et al. "Enhanced production of multi-strange hadrons in high-multiplicity
540 proton-proton collisions". In: *Nature Phys.* 13 (2017), pp. 535–539. DOI: 10.1038/nphys4111.
541 arXiv: 1606.07424 [nucl-ex].
- 542 [5] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. "The Anti-k(t) jet clustering al-
543 gorithm". In: *JHEP* 04 (2008), p. 063. DOI: 10.1088/1126-6708/2008/04/063. arXiv:
544 0802.1189 [hep-ph].
- 545 [6] Carlos A. Salgado and Urs Achim Wiedemann. "Calculating quenching weights". In: *Phys.*
546 *Rev. D* 68 (2003), p. 014008. DOI: 10.1103/PhysRevD.68.014008. arXiv: hep-ph/0302184
547 [hep-ph].
- 548 [7] Yuri L. Dokshitzer and D. E. Kharzeev. "Heavy quark colorimetry of QCD matter". In:
549 *Phys. Lett. B* 519 (2001), pp. 199–206. DOI: 10.1016/S0370-2693(01)01130-3. arXiv:
550 hep-ph/0106202 [hep-ph].
- 551 [8] Georges Aad et al. "Performance of b-Jet Identification in the ATLAS Experiment". In:
552 *JINST* 11.04 (2016), P04008. DOI: 10.1088/1748-0221/11/04/P04008. arXiv: 1512.
553 01094 [hep-ex].
- 554 [9] A. M. Sirunyan et al. "Identification of heavy-flavour jets with the CMS detector in pp
555 collisions at 13 TeV". In: *JINST* 13.05 (2018), P05011. DOI: 10.1088/1748-0221/13/
556 05/P05011. arXiv: 1712.07158 [physics.ins-det].
- 557 [10] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In:
558 *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery
559 and Data Mining*. KDD '16. San Francisco, California, USA: ACM, 2016, pp. 785–794.
560 ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: <http://doi.acm.org/10.1145/2939672.2939785>.
- 562 [11] James Bergstra and Yoshua Bengio. "Random Search for Hyper-parameter Optimization".
563 In: *J. Mach. Learn. Res.* 13 (Feb. 2012), pp. 281–305. ISSN: 1532-4435. URL: <http://dl.acm.org/citation.cfm?id=2188385.2188395>.
- 565 [12] Kurt Hornik. "Approximation capabilities of multilayer feedforward networks". In: *Neural
566 Networks* 4.2 (1991), pp. 251–257. ISSN: 0893-6080. DOI: [https://doi.org/10.1016/0893-6080\(91\)90009-T](https://doi.org/10.1016/0893-6080(91)90009-T). URL: <http://www.sciencedirect.com/science/article/pii/089360809190009T>.
- 569 [13] François Chollet et al. *Keras*. <https://keras.io>. 2015.
- 570 [14] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Sys-
571 tems*. Software available from tensorflow.org. 2015. URL: <https://www.tensorflow.org/>.
- 572 [15] Timothy Dozat. *Incorporating Nesterov Momentum into Adam*. 2015. URL: {http://cs229.stanford.edu/proj2015/054_report.pdf}.

- 574 [16] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In:
575 *CoRR* abs/1412.6980 (2014). arXiv: 1412 . 6980. URL: <http://arxiv.org/abs/1412.6980>.
- 577 [17] Nitish Srivastava et al. “Dropout: A Simple Way to Prevent Neural Networks from Over-
578 fitting”. In: *J. Mach. Learn. Res.* 15.1 (Jan. 2014), pp. 1929–1958. ISSN: 1532-4435. URL:
579 <http://dl.acm.org/citation.cfm?id=2627435.2670313>.
- 580 [18] Torbjorn Sjostrand, Stephen Mrenna, and Peter Z. Skands. “A Brief Introduction to
581 PYTHIA 8.1”. In: *Comput. Phys. Commun.* 178 (2008), pp. 852–867. DOI: 10.1016/j.
582 cpc.2008.01.036. arXiv: 0710.3820 [hep-ph].
- 583 [19] Peter Skands, Stefano Carrazza, and Juan Rojo. “Tuning PYTHIA 8.1: the Monash 2013
584 Tune”. In: *Eur. Phys. J.* C74.8 (2014), p. 3024. DOI: 10.1140/epjc/s10052-014-3024-y.
585 arXiv: 1404.5630 [hep-ph].
- 586 [20] René Brun et al. “GEANT Detector Description and Simulation Tool”. In: (1994). DOI:
587 10.17181/CERN.MUHF.DMJ1.
- 588 [21] Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. “FastJet User Manual”. In: *Eur.*
589 *Phys. J.* C72 (2012), p. 1896. DOI: 10.1140/epjc/s10052-012-1896-2. arXiv: 1111.
590 6097 [hep-ph].