

Spis treści

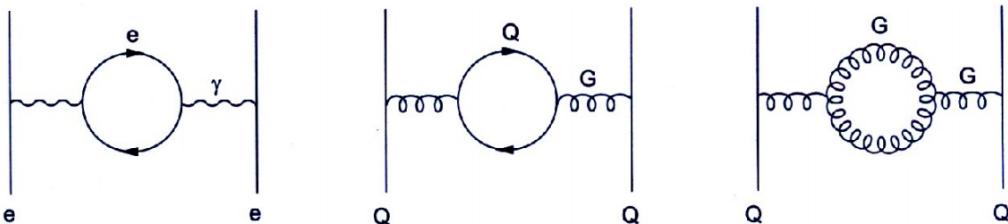
1	Fizyka dżetów cząstek	2
1.1	Chromodynamika kwantowa	2
1.2	Plazma kwarkowo-gluonowa	3
1.3	Dżety	4
1.4	Dżety b	4
1.5	Eksperyment ALICE	5
2	Uczenie maszynowe	9
2.1	Wzmacniane drzewa decyzyjne	9
2.2	Sieci neuronowe	10
2.3	Dyskusja użycia dwóch algorytmów	15
3	Dane	16
4	Analiza	19
4.1	Dobór metryki	19
4.2	Wyniki dla poszczególnych modeli	22
4.3	Korelacje predykcji modeli	27
4.4	Analiza istotności zmiennych w BDT	28
4.5	Analiza wpływu zmiennych na predykcje modeli	31
19	Dodatki	39
20	Dodatek A Metryki	39
21	Dodatek B Skróty i oznaczenia	40

22 1 Fizyka dżetów cząstek

23 1.1 Chromodynamika kwantowa

24 Chromodynamika kwantowa (ang. *Quantum Chromodynamics – QCD*) to kwantowa teoria
 25 pola opisująca oddziaływanie silne [1]. Wprowadza ona dla kwarków nową liczbę kwantową
 26 nazywaną kolorem lub ładunkiem kolorowym, który jest odpowiednikiem ładunku elektrycznego
 27 w elektrodynamice kwantowej (ang. *Quantum Electrodynamics – QED*), ale w przeciwnieństwie
 28 do niego może przyjmować 3 różne wartości (i trzy antywartości dla antykwarków). Elementarne
 29 oddziaływanie w obu teoriach przenoszone są przez bezmasowe bozony pośredniczące: w *QED*
 30 jest to elektrycznie obojętny foton a w *QCD* gluony, które występują w 8 odmianach i są
 31 kolorowo naładowane, przez co możliwe jest oddziaływanie zachodzące między dwoma gluonami.
 32 Kwarki i gluony zbiorczo nazywane są partonami.

33 Próżnia, w rozumieniu klasycznym będąca zupełnie pusta, w teoriach kwantowych wypeł-
 34 niona jest pojawiającymi i znikającymi wirtualnymi cząstками. Cząstki te ekranują ładunek
 35 próbkowy umieszczony w kwantowej próżni, wywołując zjawisko polaryzacji próżni (analogiczne
 36 do polaryzacji dielektryków), które efektywnie zmniejsza pole wytwarzane przez ten ładunek.
 37 Siła tego efektu zależy od liczby ekranujących cząstek, czyli pośrednio od skali odległości. Skala
 38 ta wyznaczona jest przez długości fali próbującej cząstki, zatem także jej energię (im więk-
 39 sza energia, tym mniejsza długość fali i mniejsza ilość ekranujących cząstek obserwowanych
 40 w pobliżu rzeczywistego ładunku, zatem tym słabszy efekt ekranowania i większy efektywny
 41 ładunek). Prowadzi to do zależnej od energii stałej sprężenia α , którą nazywamy efektywną
 42 lub biegącą stałą sprężenia (ang. *running coupling constant*).



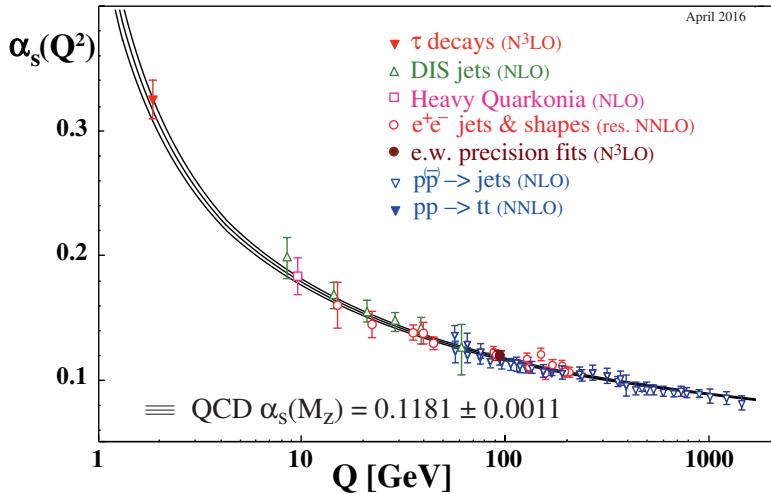
Rysunek 1: Diagramy Feynmana opisujące polaryzację próżni w *QED* (lewy) i *QCD* (środkowy i prawy). Rysunki lewy i środkowy są swoimi odpowiednikami w tych dwóch teoriach, natomiast prawy, w którym oddziałują jedynie bozony pośredniczące nie ma swojego odpowiednika w *QED*. Źródło: [1]

43 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
 44 jest także samooddziałanie gluonów, przez co dopuszczalne są diagramy Feynmana jak ten
 45 przedstawiony na Rys. 1 po prawej. Pętle gluonowe działają anty-ekranującą – zwiększą efek-
 46 tywną wartość silnej stałej sprężenia, ponadto jest to efekt dominujący nad przyczynkiem od
 47 par kwark-antykwark, co sprawia że zależność biegącej stałej sprężenia w *QCD* jest odwrotna
 48 i dużo silniejsza niż w przypadku *QED*. Wartość α_{em} maleje od wartości $\frac{1}{128}$ przy energiach ok.
 49 90 GeV do $\frac{1}{137}$ przy energii bliskiej zeru, co oznacza zmianę o kilka procent. Tymczasem α_S
 50 rośnie w miarę zbliżania się do niskich energii od wartości $\alpha_S \lesssim 0.1$ dla $E \gtrsim 100$ GeV do $\alpha_S > 1$
 51 dla energii poniżej 200 MeV (por. Rys. 2). Prowadzi to do dwóch zjawisk charakterystycznych
 52 dla chromodynamiki kwantowej:
 53

- 54 • asymptotyczna swoboda (ang. *asymptotic freedom*) [2], [3] – dla wysokich energii ($\gtrsim 100$
 55 GeV) silna stała sprężenia jest mała (w tym zakresie energii możliwe jest stosowanie

56 rachunku perturbacyjnego) i kwarki wewnątrz hadronów zachowują się jak cząstki quasi-
 57 swobodne.

- 58 • uwiecznienie koloru (ang. *colour confinement*) – przy zwiększaniu odległości między par-
 59 tonami siła oddziaływanego rośnie do nieskończoności, dlatego nigdy nie obserwuje się
 60 swobodnych cząstek obdarzonych ładunkiem kolorowym a jedynie związane w kolorowo
 61 obojętne hadrony.



Rysunek 2: Zależność silnej stałej sprzężenia od przekazu czteropędu. Źródło: [4].

62 1.2 Plazma kwarkowo-gluonowa

63 Odkrycie asymptotycznej swobody pozwoliło na sprawdzenie przewidywań *QCD* w warun-
 64 kach bardzo wysokich temperatur oraz gęstości. Dla wystarczająco dużych gęstości hadrony
 65 zaczynają na siebie zatoczyć, prowadząc do stworzenia stanu, w którym poszczególne hadrony
 66 przestają być odróżnialne [5]. Zasugerowane zostało także istnienie przejścia fazowego w tem-
 67 peraturze porównywalnej z masą pionów oraz w temperaturze niższej, ale przy odpowiednio
 68 dużych gęstościach [6]. Stan materii powstały po osiągnięciu, któregoś z tych warunków nazy-
 69 wany jest plazmą kwarkowo-gluonową (ang. *quark-gluon plasma – QGP*). Obecnie przewiduje
 70 się, że materia w takim stanie istniała w pierwszych ułamkach sekund po Wielkim Wybuchu
 71 [7] oraz, że może się znajdować w jądrach gwiazd neutronowych [8].

72 Obecnie aby uzyskać dostęp do materii w stanie plazmy kwarkowo-gluonowej potrzebne są
 73 wysokoenergetyczne zderzenia cząstek. Powszechnie mówi się o niej w kontekście zderzeń cięż-
 74 kich jonów, chociaż istnieją także prace doszukujące się obecności *QGP* w mniejszych systemach
 75 np. w zderzeniach proton-proton [9], [10].

76 Cechą charakterystyczną *QGP* jest obecność wolnych kwarków i gluonów. Ze względu na
 77 uwiecznienie koloru w każdym innym stanie materii są one zawsze związane i tworzą hadrony.
 78 Wolne kwarki i gluony powstające w zderzeniach muszą zatem przejść przez proces hadroni-
 79 zacji, w którym rekombinują one ze spontanicznie wytworzonymi nowymi partonami, tworząc
 80 hadrony. W wyniku tego procesu, z każdego partonu obecnego w początkowym etapie zderzenia
 81 może powstać wiele cząstek poruszających się podobnym kierunkiem, tworząc stożek z wierzchoł-
 82 kiem blisko punktu interakcji wiązek. Taki stożek skolimowanych cząstek nazywany jest dżetem
 83 cząstek.

84 1.3 Dżety

85 Przedstawiona powyżej definicja dżetu nie jest precyzyjna z punktu widzenia pracy ekspery-
86 mentalnej. W detektorze obserwuje się tylko cząstki w stanie końcowym, nie jest zatem możliwe
87 przyporządkowanie cząstki według jej pochodzenia. W związku z tym, konieczne jest użycie al-
88 gorytmu klasteryzującego, dostającego na wejściu tylko obserwowalne eksperymentalnie cząstki.
89 To jakie dżety zostaną zaobserwowane w danym zdarzeniu zależy od użytego algorytmu. Ozna-
90 cza to, że precyzyjną definicję dżetu stanowi algorytm klasteryzujący wraz z zestawem para-
91 metrów. Obecnie najpowszechniej stosowanym algorytmem jest algorytm *anti-kt* [11].

92 Eksperimentalne ograniczenia związane z obserwacją tylko końcowego stanu oddziaływań
93 nie występują w analizie danych z symulacji Monte Carlo (MC), gdzie ma się dostęp do peł-
94 nej informacji na temat historii każdej cząstki. Nie należy jednak używać jej do klasteryzacji
95 dżetów, gdyż utracona została cecha odpowiedniości między obiektami nazywanymi dżetami
96 w symulacji i w eksperymencie, która to cecha jest niewątpliwie jedną z podstawowych wyma-
97 gań stawianych przed dobrą symulacją. Właściwym podejściem jest rekonstrukcja dżetów przy
98 pomocy takiego samego algorytmu jak w przypadku danych eksperimentalnych.

99 Dżety wykorzystuje się w badaniach plazmy kwarkowo-gluonowej. Dają one pośredni wgląd
100 we właściwości *QGP* na podstawie wpływu jaki wywiera na oddziałujące z nią partony. Przy-
101 kładową obserwabłą mierzoną w zderzeniach ciężkich jonów jest czynnik modyfikacji jądrowej
102 R_{AA} (ang. *nuclear modification factor*), który jest miarą strat energii przez parton przechodzący
103 przez medium. Jest to stosunek pędowych rozkładów dżetów cząstek zmierzonych w
104 zderzeniach ciężkich jąder oraz w zderzeniach pp (przemnożonych przez liczbę binarnych zde-
105 rzeń nukleon-nukleon przewidywanych przez model teoretyczny). Odchylenia od wartości 1 dla
106 wysokich p_T są oznaką modyfikacji pędów dżetów przez gęste medium (w stosunku do zderzeń
107 pp gdzie *QGP* nie powstaje), jest to tzw. tłumienie dżetów (ang. *jet quenching*). Rezultaty
108 pomiarów R_{AA} w CMS: [12] i ALICE: [13].

109 Oprócz globalnego wpływu medium na dżety, analizuje się także różnice między dżetami
110 pochodząymi z gluonów oraz kwarków o różnych zapachach (ang. *flavours*). Modele teore-
111 tyyczne przewidują między innymi większe straty energii w wyniku interakcji z *QGP* dla dżetów
112 gluonowych niż kwarkowych [14] oraz zależność strat energii od masy partonu [15] – w tym
113 przypadku precyzyjne pomiary rozróżniające typy dżetów pozwalają lepiej zrozumieć mecha-
114 nizm odpowiadający za straty energii przez partony. Zagadnienie rozpoznania z jakiego rodzaju
115 partonu powstał dany dżet, nazywane jest identyfikacją lub tagowaniem dżetu. Ważną rolę
116 w studiowaniu tego problemu odgrywają symulacje MC, które pozwalają określić wydajności
117 poszczególnych technik tagowania dżetów na podstawie znajomości kanału produkcji każdej
118 symulowanej cząstki.

119 1.4 Dżety *b*

120 1.4.1 Właściwości

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

126 Identyfikacja dżetów *b* jest sporym wyzwaniem ze względu na zdecydowanie częściej wy-
127 stępujące dżety lekkie, tj. powstałe z hadronizacji kwarków *u,d,s* lub gluonów. Rozpoznawanie
128 dżetów *b* bazuje na charakterystycznych właściwościach hadronów zawierających kwark piękny:
129 relatywnie długim czasie życia oraz (w mniejszym stopniu) na ich pół-leptonowym rozpadach

130 o względnej częstotliwości rozpadu w tym kanale (ang. *branching ratio*) na poziomie 10%.

131 1.4.2 Przegląd algorytmów używanych do identyfikacji dżetów b na LHC

132 Używane w eksperymentach na LHC: ATLAS, CMS i ALICE algorytmy można podzielić na
133 trzy kategorie: wykorzystujące wtórne wierzchołki, informację o odległości najbliższego zbliżenia
134 (parametrze zderzenia) częstek tworzących dżet (ang. *Distance of Closest Approach – DCA*,
135 *Impact Parameter – IP*) oraz identyfikujące produkty półleptonowych rozpadów pięknych lub
136 powabnych hadronów. Dokładne opisy omawianych algorytmów można znaleźć w: [16], [17]
137 (ATLAS), [18], [19] (CMS), [20], [21] (ALICE).

138 Najprostszym algorytmem jest dyskryminacja na podstawie istotności statystycznej (wynik
139 pomiaru podzielony przez jego niepewność) odległości wtórnego wierzchołka od wierzchołka
140 pierwotnego L . Jest to metoda wykorzystywana w każdym z trzech wymienionych eksperymentów (por. ATLAS: algorytm SV0, CMS: algorytm SSV, ALICE). Może być ona rozszerzona
141 poprzez użycie dodatkowych zmiennych opisujących wtórny wierzchołek jak na przykład jego
142 masa, ułamek niesionej przez niego całkowitej energii dżetu (por. ATLAS: SV1) lub użycie
143 "pseudowierzchołków" (kombinacji dwóch częstek o dużych *DCA*) w celu poprawienia wydaj-
144 ności detekcji o przypadki, w których wtórny wierzchołek nie został zrekonstruowany (por.
145 CMS: CSV).

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

152 Do wykorzystania półleptonowego kanału rozpadu ciężkich hadronów do identyfikacji dżetów
153 b w eksperymentach ATLAS (SMT) i CMS (SE, SM) użyto wzmacnianych drzew decyzyjnych
154 trenowanych na kilku ręcznie zdefiniowanych w tym celu zmiennych takich jak pęd leptonu
155 transwersalny względem osi dżetu.

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

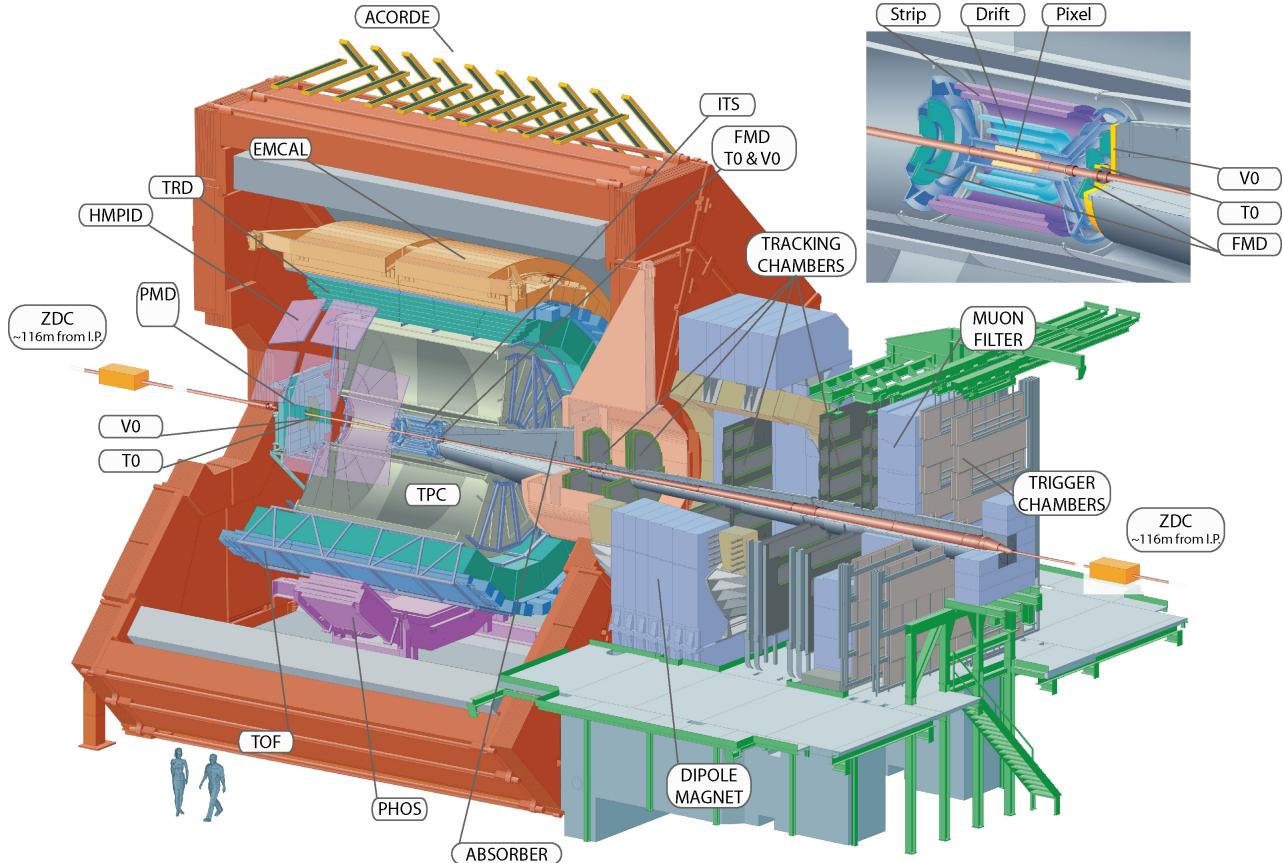
160 Przykład innego podejścia zaprezentowała współpraca przy eksperymencie LHCb, gdzie
161 wykorzystano dwa zestawy wzmacnianych drzew decyzyjnych operujących na zmiennych zwią-
162 zanych z wtórnymi wierzchołkami. Pierwszy zapewnia separację dżetów lekkich od ciężkich a
163 drugi odróżnia dżety b od c . Do wyboru punktu pracy, zamiast jednowymiarowego rozkładu
164 predykcji używany jest dwuwymiarowy rozkład wag przypisany przez oba klasyfikatory [22].

166 1.5 Eksperyment ALICE

167 Eksperyment ALICE [23], [24] jest jednym z czterech największych eksperymentów na LHC.
168 Jest on dedykowany zderzeniom ciężkich jonów (w LHC są to jony ołowiu PbPb), ale mierzone
169 są także mniejsze systemy, tj. proton-proton pp (głównie jako referencję dla pomiarów PbPb)
170 oraz proton-ołów p+Pb, które dostarczają także okazji do badania asymetrycznych zderzeń.
171 Cechą charakterystyczną pomiarów ciężkojonowych jest ich znacznie większa niż w przypadku
172 zderzeń pp krotność, tj. liczba częstek wyprodukowana w pojedynczym zderzeniu. W przypadku
173 zderzeń PbPb może powstawać nawet do 8000 naładowanych częstek na jednostkę pseudorapi-
174 dyty η (ang. *pseudorapidity*)¹. Detektor ALICE został zoptymalizowany do mierzeniach takich

¹ $\eta = -\ln[\tan(\frac{\theta}{2})]$, gdzie θ jest kątem między wektorem pędu częstki a osią wiązki

175 przypadków, jak również pod kątem rekonstrukcji i identyfikacji cząstek o szerokim zakresie
 176 pędów (100 MeV – 100 GeV).



Rysunek 3: Schemat detektora ALICE. Źródło: [25]

177 Detektor ALICE jest urządzeniem złożonym z wielu subdetektorów, schematycznie przed-
 178stawionych na Rys. 3. Można je podzielić według pełnionej w pomiarach roli. ITS, TPC, TRD
 179 oraz TOF pokrywają pełen kąt azymutalny oraz zakres pseudopospieszności $|\eta| < 0.9$.

- 180 • Detektory śladowe – mierzące trajektorie cząstek zakrzywiane w polu magnetycznym o
 181 wartości $B = 0.5$ T.
 - 182 – Inner Tracking System (ITS) – zespół krzemowych detektorów śladowych znajdujący
 183 się najbliżej miejsca interakcji wiązek. Składa się on z 6 cylindrycznych warstw o pro-
 184 mieniach od 4 do 43 cm, wykonanych w trzech różnych technologiach. Jego główną
 185 rolą jest rekonstrukcja pierwotnego oraz wtórnego wierzchołków. Bierze także udział
 186 w rekonstrukcji trajektorii i strat energetycznych cząstek, szczególnie tych niskope-
 187 dowych, które nie docierają do dalej położonych detektorów.
 - 188 – Time Projection Chamber (TPC) – dłuża na 5m i o takiej średnicy komora projekcji
 189 czasowej. Jest to główny detektor śladowy ALICE, wraz z ITS służy do wyznaczania
 190 trajektorii cząstek i na ich podstawie również wierzchołków zderzenia. Elektrony
 191 uwolnione ze zjonizowanego przez poruszające się w nim naładowane cząstki gazu
 192 dryfują wzdłuż kierunku wiązki w stronę końcowych elektrod. Następnie są tam
 193 zbierane dostarczając informacji o dwóch współrzędnych toru cząstki: odległości od
 194 wiązki i kącie azymutalnym. Trzecia składowa trajektorii jest otrzymywana na pod-
 195 stawie czasu dotarcia elektronów do elektrod. TPC jest najwolniejszym detektorem
 196 ALICE (ze względu na ograniczający czas dryfu elektronów wynoszący $\sim 90 \mu\text{s}$),

197 użycie detektora tego typu podyktowane jest jego zdolnością do rozwikłania śladów
198 tysięcy cząstek spodziewanych w centralnych zderzeniach PbPb.

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

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

- 204 – Transition Radiation Detector (TRD) – detektor wykrywający promieniowanie przej-
205 scia, służy głównie do odróżniania wysokopędowych ($p_T > 1$ GeV) elektronów od
206 pionów. Promieniowanie przejścia emitowane jest podczas przechodzenia relatywi-
207 stycznych cząstek przez granicę ośrodków, jego intensywność jest proporcjonalna do
208 czynnika Lorentza γ , co pozwala na odróżnienie cząstek o tym samym pędzie na pod-
209 stawie różnicy mas (elektrony są ponad 250 razy lżejsze od pionów). TRD oprócz
210 identyfikacji elektronów uczestniczy także w rekonstrukcji śladów wysokopędowych
211 cząstek i może być użyty w systemach wyzwalania (ang. *trigger*).
212 – Time-Of-Flight (TOF) – detektor czasu przelotu o zdolności rozdzielczej ~ 80 ps.
213 Pozwala na separację pionów i kaonów o pędach do ok. 2.5 GeV i protonów do 4
214 GeV.
215 – High-Momentum Particle Identification Detector (HMPID) – detektor typu RICH
216 (ang. *ring-imaging Cherenkov*), wykrywający fotony emitowane podczas przejścia
217 przez ośrodek naładowanej cząstki o prędkości większej od prędkości fazowej światła
218 w tym ośrodku (promieniowanie Cherenkowa). Na podstawie kąta pod jakim emitow-
219 ane są fotonów określana jest prędkość cząstki. HMPID pozwala na identyfikację pio-
220 nów, kaonów i protonów o $p_T > 1$ GeV. Pokrywa przestrzeń kątów: $1.2^\circ < \phi < 58.8^\circ$
221 oraz $|\eta| < 0.6$ (5% akceptancji TPC).

222 • kalorymetry

- 223 – Photon Spectrometer (PHOS) – elektromagnetyczny kalorymetr o wysokiej rozdziel-
224 czości energetycznej i przestrzennej (podzielony na kryształy o rozmiarze poprzecznym
225 2.2×2.2 cm, co odpowiada rozmiarowi w dziedzinie η , ϕ 0.004×0.004). Pokrywa
226 zakres pseudopospieszności $|\eta| < 0.12$ i kąta azymutalnego równy 100° . PHOS ma
227 za zadanie identyfikację i pomiar czteropędów fotonów, w szczególności tych niepo-
228 chodzących z rozpadu innych cząstek (ang. *direct photons*) oraz lekkich mezonów
229 neutralnych (np. π^0) przez dwufotonowy kanał rozpadu.
230 – Electromagnetic Calorimeter (EMCal) – drugi elektromagnetyczny kalorytmeter ALICE
231 o mniejszej ziarnistości ($\Delta\eta, \Delta\phi = 0.014 \times 0.014$), ale dużo większej akceptancji
232 ($|\eta| < 0.7$, $\Delta\phi = 107^\circ$). EMCal poprawia możliwości ALICE w zakresie pomiarów
233 tłumienia dżetów, pozwalając na wyznaczanie neutralnej składowej energii dżetów
234 (energii niesionej przez neutralne cząstki). Dzięki innej charakterystyce dla elektro-
235 nów i hadronów (elektrony typowo deponują niemal całą energię a hadrony tylko
236 niewielką część) pozwala je odróżnić na podstawie stosunku zmierzonej w nim ener-
237 gii do wyznaczonego wcześniej pędu E/p . EMCal może być użyty także w szybkim
238 systemie wyzwalania, do selekcji przypadków z dżetami oraz wysokoenergetycznymi
239 fotonami i elektronami.

240 • Muon spectrometer – spektrometr mionowy, złożony z dwóch pasywnych absorberów,
241 znajdujących się między nimi 10 warstw detektora śladowego oraz komór systemu wy-
242 zwalającego na końcu. Przedni absorber, gruby na 4 metry ($\sim 60X_0$) wykonany z betonu

i grafitu, zatrzymuje hadrony oraz miony o niższych energiach (np. z rozpadów pionów i kaonów). Jest on zoptymalizowany aby minimalizować rozpraszanie mionów i zapewnić ochronę pozostałych detektorów ALICE przed wtórnymi cząstkami powstały w jego materiale. Komory pozycjoczułe mają zdolność rozdzielczą ok. $100 \mu\text{m}$, co pozwala osiągnąć wysoką rozdzielczość przy wyznaczaniu masy niezmienniczej rzędu $100 \text{ MeV}/c^2$. Spektrometr mionowy służy głównie do mierzenia mezonów wektorowych (ω , ϕ , J/Ψ , Υ) rozpadających się w kanale $\mu^+ \mu^-$.

- Detektory przednie, wyznaczające min. centralność zderzeń oraz płaszczyznę reakcji.

- ZDC – zespół czterech kalorymetrów (po dwa do pomiaru protonów i neutronów, gdyż ich tory są rozdzielane przez pole magnetyczne) mierzących energię nukleonów nieuczestniczących w zderzeniu tzw. obserwatorów, co pozwala na określenie liczby nukleonów oddziałujących, tzw. uczestników. Znajdują się one 116 m od miejsca interakcji.
- PMD – detektor mierzący krotności oraz rozkład przestrzenny fotonów
- FMD – krzemowy detektor paskowy mierzący precyjnie liczbę naładowanych cząstek w zakresie pseudopospieszności wykraczającym poza akceptancję detektora ITS.
- V0 – liczniki scyntylacyjne położone po obu stronach detektora, używane w systemie wyzwalania o minimalnym obciążeniu (ang. *minimum bias trigger*) – wymóg obecności sygnału w obu detektorach pozwala na odrzucenie przypadków tła z oddziaływania wiązki protonów z reszkami gazu obecnymi w rurach próżniowych.
- T0 – dostarcza dokładny czas interakcji potrzebny dla detektora TOF, pozwala także na śledzenie światłości w czasie rzeczywistym.

2 Uczenie maszynowe

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

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*).

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

Występuje wiele algorytmów uczenia maszynowego takich jak regresja liniowa i logistyczna, drzewa decyzyjne i ich wariacje, maszyny wektorów wspierających, sztuczne sieci neuronowe oraz wiele innych [26], [27]. Uczenie polega na znalezieniu pewnej funkcji dopasowującej do przyjmowanego na wejściu zestawu (wektora) cech (zmiennych, kolumn) pewną odpowiedź (predykcję), która minimalizuje zadaną funkcję straty. Jej rolę w przypadku regresji często pełni błąd średniokwadratowy a w przypadku klasyfikacji np. entropia krzyżowa (ang. *cross entropy*)². Różne algorytmy szukają przy tym funkcji dopasowującej należącej do różnych klas funkcji: przykładowo klasyczne drzewa decyzyjne przeszukują tylko przestrzeń funkcji dających się opisać skończonym zbiorem reguł "jeśli – to" (ang. *if – else*).

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

2.1 Wzmacniane drzewa decyzyjne

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 ³.

Drzewa decyzyjne są często łączone w komitety klasyfikatorów (ang. *ensemble methods*). Wiele "słabych" klasyfikatorów jest łączonych w jeden silny dwa sposoby: workowania (ang. *bagging*) [28] oraz wzmacniania (ang. *boosting*) [29], które są często ze sobą porównywane.

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

² $J = -\frac{1}{N} \sum_{i=1}^N [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)]$, gdzie y_i to prawidłowa klasa i -tego przykładu a \hat{y}_i to predykcja algorytmu

³ $G_L = 1 - \sum_k p_k^2$ 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

306 kowo, do uczenia każdego drzewa używa się tylko podzbioru wszystkich cech uczących. Końcową
307 predykcję algorytmu otrzymuje się poprzez "głosowanie" wszystkich drzew z odpowiednimi wa-
308 gami.

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

315 **Parametry i sposób trenowania drzew decyzyjnych na analizowanych danych**

316 W niniejszej pracy wykorzystano wzmacniane drzewa decyzyjne zaimplementowane w wydaj-
317 nej bibliotece **XGBoost** [30]. Szybkość obliczeń jest bardzo ważna, gdyż oprócz komfortu pracy
318 z algorytmem, przekłada się na jakość otrzymanych wyników – krótszy czas obliczeń oznacza
319 możliwość przeprowadzenia większej ilości eksperymentów i lepsze dobranie parametrów oraz
320 danych. Implementacja wzmacnianych drzew decyzyjnych w **XGBoost** wykorzystuje wszystkie
321 rdzenie procesora, pomimo że sam algorytm ma charakter sekwencyjny – jest to możliwe dzięki
322 paralelizacji procesu tworzenia każdego drzewa (przed każdym podziałem konieczne jest spraw-
323 dzenie pewnej ilości możliwych zmiennych i wartości progowych i ten proces jest wykonywany
324 równolegle).

325 Dzięki szybkiemu uczeniu się algorytmu, możliwe było użycie kosztownego obliczeniowo au-
326 tomatycznego przeszukiwania przestrzeni parametrów przy pomocy przeszukiwania losowego
327 (ang. *random search*), które jest zwykle preferowane nad przeszukiwanie sieciowe [31]. W tym
328 celu cały zbiór danych dzielony był na dwie części: trenującą oraz testową (80/20%). Następ-
329 nie algorytm był trenowany i oceniany z użyciem trzy- lub pięciokrotnej walidacji krzyżowej
330 (ang. *cross-validation*) na zbiorze trenującym dla różnych zestawów parametrów. Model z naj-
331 lepszym wynikiem uzyskanym w walidacji krzyżowej był sprawdzany na zbiorze testowym.

332 Parametry optymalizowane w opisanym procesie to:

- 333 • *max_depth* – maksymalna głębokość każdego drzewa (niekoniecznie osiągana)
- 334 • *n_estimators* – liczba drzew
- 335 • *learning_rate* – parametr szybkości uczenia, komplementarny do *n_estimators*, w praktyce
336 można ustalić liczbę drzew i szukać optymalnej szybkości uczenia
- 337 • *subsample*, *colsample_bytree*, *colsample_bylevel* – parametry regularyzacyjne określające
338 ułamek kolejno: wierszy użytych do trenowania każdego drzewa, kolumn użytych w ka-
339 dym drzewie (cechy losowane raz dla danego drzewa), kolumn użytych przy każdym po-
340 dziale (cechy losowane przy każdym podziale)
- 341 • γ – minimalny zysk w postaci zmniejszenia wartości funkcji straty konieczny do wykonania
342 podziału

343 **2.2 Sieci neuronowe**

344 Sieci neuronowe (ang. *neural networks* – *NN*) są szczególnym algorytmem uczenia maszyno-
345 wego. Występują w bardzo wielu odmianach i są wykorzystywane w rozwiązywaniu szerokiej
346 gamy problemów. Nawet bardzo pobiczny opis sieci neuronowych wymaga dużo więcej miejsca
347 niż może być temu poświęcone w tej pracy. Wprowadzenia do sieci neuronowych od podstaw
348 można znaleźć m.in. w [32] lub [33]. Tu przedstawione zostaną wyłącznie wybrane zagadnienia

349 mające ścisłejšzy związek z pracą. Używane mogą być terminy, których znaczenie wyjaśniane
350 jest w podanych źródłach.

351 W niniejszej pracy, wykorzystane zostały dwa rodzaje sieci neuronowych: sieci w pełni połącz-
352 czone (ang. *fully connected NN – FC NN*), nazywane także wielowarstwowymi perceptronami
353 (ang. *multi-layer perceptron – MLP*) oraz sieci konwolucyjne (ang. *convolutional NN – Co-*
354 *nvNets, CNN*).

355 Sieci w pełni połączone

356 W nierekurencyjnych sieciach neuronowych (tylko takie są używane w tej pracy), informacja jest
357 przekazywana kolejno od warstw wejściowych, poprzez warstwy ukryte aż do wyjściowej. W sie-
358 ciach typu *FC* wszystkie warstwy składają się z identycznych neuronów – każdy neuron dostaje
359 na wejściu wektor, natomiast zwraca skalar – wartość pewnej zadanej, nielinowej funkcji, jako
360 argument podając średnią ważoną z elementów wektora wejściowego. Wartości zwracane przez
361 neurony w danej warstwie składają się na wektor wejściowy dla neuronów kolejnej warstwy.
362 Wejściem dla pierwszej warstwy są natomiast kolejne przykłady ze zbioru uczącego. Trenowa-
363 nie sieci neuronowych polega na zmienianiu wag (parametrów) w liczonej w każdym neuronie
364 średniej, każdy neuron posiada własny, niezależny zestaw wag.

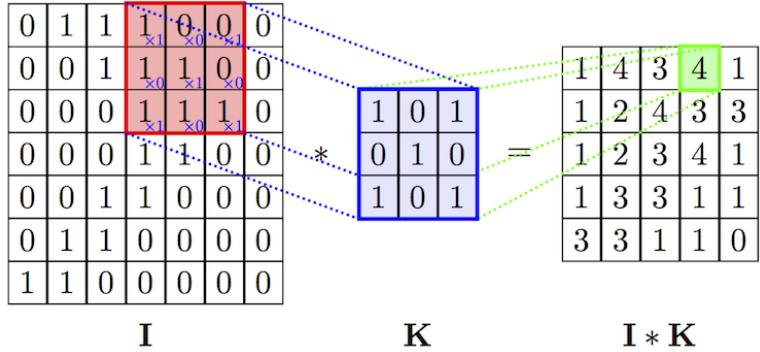
365 Istnieje twierdzenie o sieciach neuronowych jako uniwersalnych aproksymatorach funkcji
366 (ang. *universal approximation theorem*) [34], mówiące, że już sieć neuronowa o jednej warstwie
367 ukrytej jest zdolna do przybliżenia dowolnej funkcji z dowolną dokładnością. Twierdzenie to
368 nie podaje niestety liczby potrzebnych neuronów a przede wszystkim – sposobu ich trenowania.
369 Trenowanie jest prostsze w przypadku zastosowania wielu warstw, które odpowiadają kolejnym
370 poziomu abstrakcji jednak nadal jest dużym wyzwaniem ze względu na fakt, że nawet stosun-
371 kowo niewielka sieć może posiadać bardzo dużą liczbę parametrów, przykładowo sieć o czterech
372 warstwach, w każdej po 128 neuronów ma ich ponad 65 tysięcy.

373 Sieci konwolucyjne

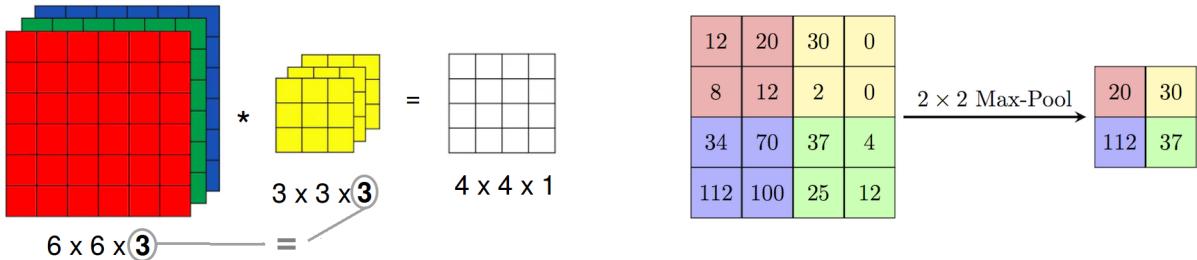
374 Jednym ze sposobów na ograniczenie liczby trenowanych parametrów jest użycie konwolucyj-
375 nych sieci neuronowych [35] (bardziej poprawną choć rzadko używaną nazwą w języku polskim
376 jest sieć splotowa). Są one inspirowane połączeniami w korze wzrokowej zwierząt i wywodzą
377 się z badań w obszarze widzenia komputerowego, gdzie liczby parametrów są szczególnie duże
378 (wektor wejściowy ma wymiar równy liczbie pikseli w obrazie), na takim przykładzie również
379 najłatwiej zrozumieć ich działanie.

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

387 W przypadku gdy zamiast wejścia dwuwymiarowego (jak np. obraz czarno-biały), mamy do
388 czynienia z wejściem trójwymiarowym (np. trzeci wymiar to kolejne kolory w kodowaniu RGB),
389 filtry również muszą mieć trzy wymiary, przy czym rozmiar w ostatnim wymiarze musi być
390 równy rozmiarowi w tym kierunku wektora wejściowego. Wynik operacji splotu jest ponownie
391 dwuwymiarowy, gdyż filtr przesuwany jest tylko w dwóch pierwszych wymiarach. Trzeci wymiar
392 powstaje przez składanie kolejnych filtrów. Widać zatem, że również w przypadku gdy na
393 wejścia podawany jest obraz czarno-biały, filtry w kolejnych warstwach konwolucyjnych (oprócz
394 pierwszej) mają po trzy wymiary.



Rysunek 4: Schemat działania pojedynczego filtra z warstwy konwolucyjnej (operacja splotu).
Źródło: [36].



Rysunek 5: Działanie pojedynczego filtra (3D) na wejście o trzech wymiarach.
Źródło: [37].

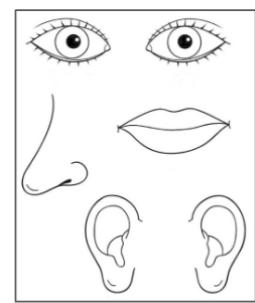
Rysunek 6: Działanie warstwy typu *max-pool*. Źródło: [36].

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

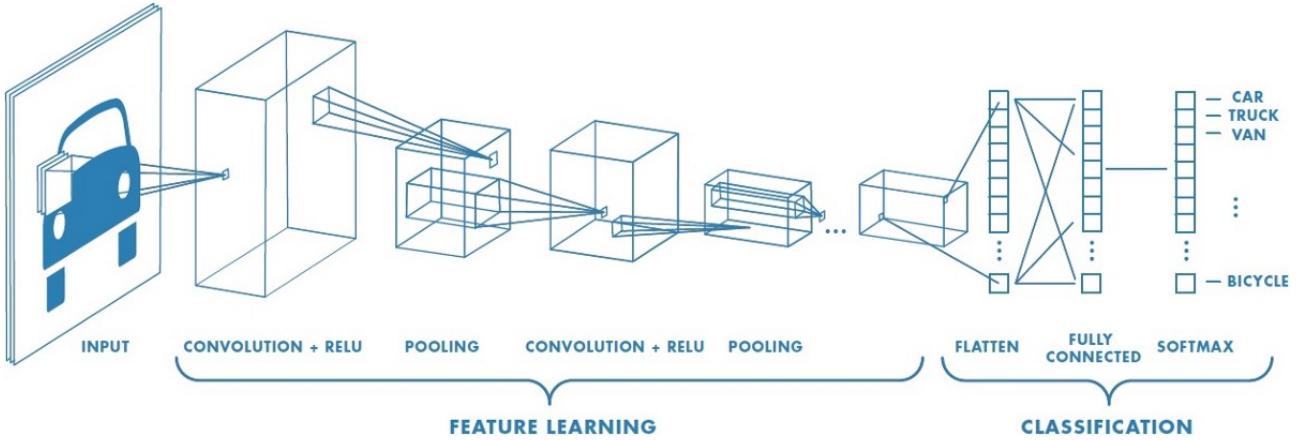
Typowa architektura stosowana w przypadku sieci konwolucyjnych jest następująca: najpierw warstwy konwolucyjne (pomiędzy nimi czasem warstwy typu *max-pool*), następnie wszystkie filtry są rozwijane i składane w długi jednowymiarowy wektor, który przekazywany jest do warstw typu *FC*. W przypadku problemu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

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

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



Rysunek 8: Źródło: [39].

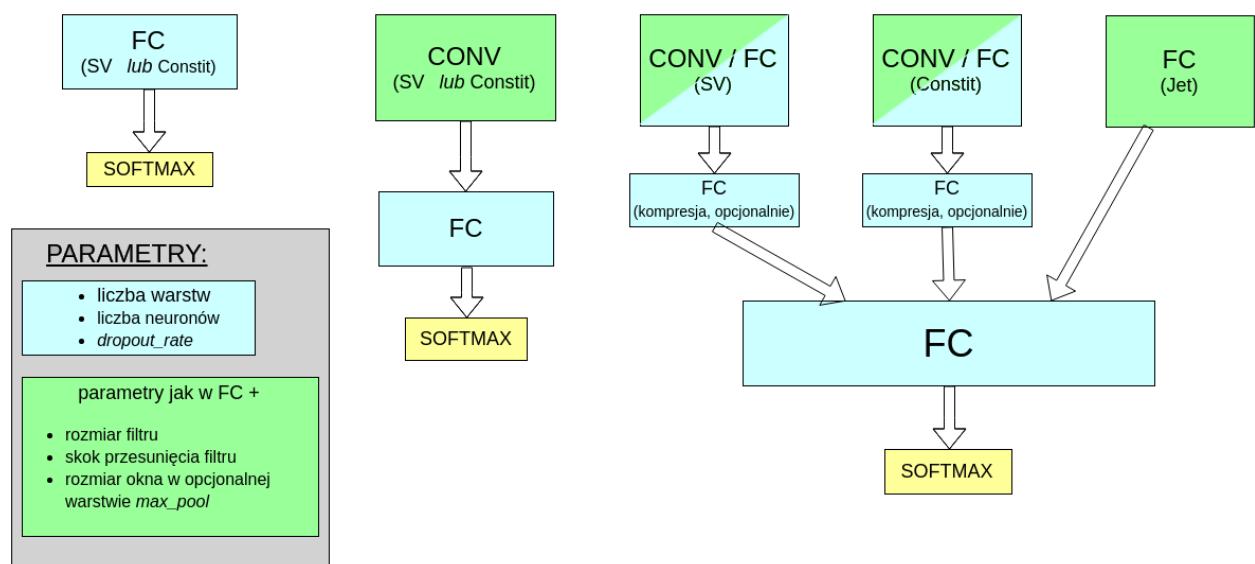


Rysunek 7: Typowa struktura stosowana w sieciach konwolucyjnych. Warstwy konwolucyjne mają za zadanie wydobywać cechy, na podstawie których późniejsze warstwy dokonują klasyfikacji. Widoczna jest charakterystyczna stopniowa zmiana rozmiaru przekazywanej macierzy: rozmiar poprzeczny maleje kosztem głębokości, co odpowiada rosnącej liczbie filtrów i malejącym rozmiarowi obszaru po jakim są one przesuwane. Źródło: [38].

415 Hiperparametry i trenowanie sieci neuronowych na analizowanych danych

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

418 Testowane były trzy architektury: sieci w pełni połączone, sieci konwolucyjne oraz sieć
 419 złożona z dwóch gałęzi, osobnych dla wtórnego wierzchołków i częstek tworzących dżet (por.
 420 Rozdz. 3) przedstawione schematycznie na Rys. 9. Do trenowania sieci wykorzystano wysoko-
 421 poziomową bibliotekę Keras [40] korzystającą z silnika obliczeniowego zaimplementowanego w
 422 TensorFlow [41].



Rysunek 9: Schematyczne przedstawienie trzech testowanych rodzin architektur sieci. Każdy blok odpowiada kilku warstwom danego typu. Bloki warstw typu *FC* i opisane jako ”kompre-
 sja” składają się z kilku neuronów i miały za zadanie zredukować całą informację z danej gałęzi
 do wektora kilku liczb.

423 Zestaw hiperparametrów definiujący działanie sieci w pełni połączonej:

- 424 • liczba warstw ukrytych

- 425 • liczba neuronów w każdej warstwie

426 • funkcja aktywacji – nielinowa funkcja aplikowana przed zwróceniem wartości w każdym
427 neuronie, najpopularniejsze to *tanh*, *ReLU* ($f(x) = \max(0, x)$) oraz funkcja sigmoidalna
428 ($f(x) = \frac{1}{1+\exp(-x)}$)

- 429 • algorytm optymalizacyjny – spadek gradientowy lub jego wariacje

- 430 • parametr szybkości uczenia i jego modyfikacje w trakcie uczenia

431 • liczba przykładów trenujących przetwarzanych w jednym kroku uczenia (ang. *batch_size*)
432 – im większy tym szybsze jest trenowanie sieci (dzięki wydajnym operacjom macierzo-
433 wym), natomiast może się to odbywać kosztem precyzji

- 434 • liczba epok uczenia – ile razy będzie pokazywany sieci każdy przykład

435 • opcjonalnie: warunki stopu (ang. *early stopping*) w razie osiągnięcia *plateau* (wysycenia
436 procesu uczenia)

- 437 • opcjonalnie: regularyzacja przy pomocy różnych technik (zwykle konieczna)

438 Ponadto dla sieci konwolucyjnych:

- 439 • liczba warstw konwolucyjnych i liczba filtrów w każdej warstwie

- 440 • obecność lub brak warstw *max-pool* i rozmiar ich okna

- 441 • rozmiar filtrów i długość skoku przy ich przesuwaniu

442 Same dwie pierwsze wielkości dają nieograniczoną liczbę konfiguracji. Czas trenowania sieci
443 neuronowych jest rzędu wielkości większej niż drzew decyzyjnych, dlatego przyjęto szereg kro-
444ków mających na celu zmniejszenie przeszukiwanej przestrzeni hiperparametrów. Na podstawie
445 wstępnych testów oraz różnych wskazówek dostępnych w literaturze przyjęto:

- 446 • *batch_size* zawsze równy 64 (inne testowane wartości: 16, 32, 128)

447 • za algorytm optymalizacyjny przyjęto algorytm o nazwie *Nadam* [42], tj. rozwinięcie al-
448 gorytmu *Adam* [43] o parametr Nesterova (inne testowane to zwykły spadek gradientowy
449 oraz *Adam*)

- 450 • funkcję aktywacji: *ReLU*

- 451 • liczba epok równa 50, 100 lub 200, zrezygnowano z *early stopping*

- 452 • stałe w trakcie treningu wartości parametru szybkości uczenia

453 • spośród technik regularizacyjnych testowano wyłącznie *dropout* [44] z prawdopodobień-
454 stwem odrzucenia równym 0.1, 0.2 lub 0.5

- 455 • kilka wybranych kombinacji dla zestawu parametrów: rozmiar filtra, długość skoku i roz-
456 miar okna w warstwach *max-pool* – takie same w kolejnych warstwach

- 457 ● liczby neuronów/filtrów w warstwach będące zawsze potęgami dwójki oraz stałą liczbę w
458 kolejnych warstwach lub zmieniającą się o stały czynnik, np. 256-128-64, 128-128-128 lub
459 16-32-64
- 460 ● liczba warstw *FC*: 2-8, konwolucyjnych 2-6

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

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

470 **2.3 Dyskusja użycia dwóch algorytmów**

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

477 Po drugie, wykorzystane zostały dwa algorytmy mocno różniące się w swojej naturze, co
478 pozwala wykorzystać cechy każdego z nich w analizie: przykładowo sieci neuronowe dobrze
479 radzą sobie z nieustrukturyzowanymi danymi – potrafią tworzyć wysoko poziomowe cechy na
480 podstawie niskopoziomowego wejścia (np. położenia oka na zdjęciu twarzy na podstawie pixeli).
481 Są natomiast trudne w interpretacji i często traktowane są jako tzw. „czarne skrzynki” (ang. *black*
482 *box*). Oprócz tego, liczba możliwych konfiguracji sieci jest ogromna i przez to niemożliwe jest
483 stwierdzenie czy wykorzystane zostały pełne ich możliwości.

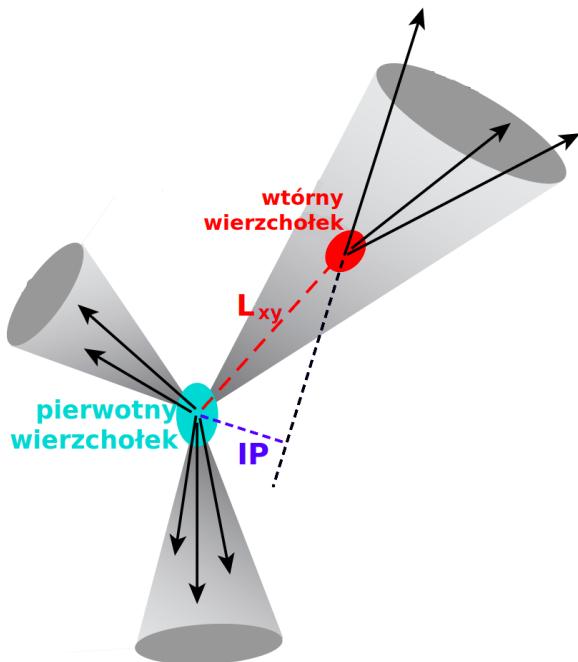
484 Z kolei drzewa decyzyjne posiadają stosunkowo niewielką liczbę parametrów, a ich trenowa-
485 nie jest bardzo szybkie co pozwala na ich ekstensywne przeszukiwanie i otrzymanie wyników,
486 które można uznać za optymalne dla tego algorytmu. Ponadto, w przypadku drzew istnieją
487 niewymagające dodatkowych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

489 3 Dane

490 Dane użyte w analizie pochodzą z symulacji Monte Carlo zderzeń proton-proton przy energii
 491 w układzie środka masy równej $\sqrt{s} = 13$ TeV dostępnych na serwerach eksperymentu ALICE.
 492 Są to pełne symulacje detektora ALICE, wykorzystujące generator zderzeń *Pythia8* [45] (*tune:*
 493 *Pythia8Jets_Monash2013* [46]) oraz pakiet *Geant3* [47] do transportu cząstek przez materiał
 494 detektora.

495 Do rekonstrukcji dżetów wykorzystany został algorytm *anti-kt* z parametrem $R = 0.4$ zaim-
 496 plementowany w pakiecie *FASTJET* [48]. Dżetów poszukiwano wyłącznie wśród cząstek naładowa-
 497 nych (ang. *charged jets*) ze względu na słabe pokrycie przestrzeni fazowej przez kalorymetry
 498 w eksperymencie ALICE.

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



Rysunek 10: Rysunek ilustrujący znaczenie używanych wielkości: L_{xy} oraz parametru zderzenia IP . Źródło: [49].

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

507 Lista używanych zmiennych:

- 508 • Zmienne na poziomie dżetu:
 - 509 – η_{jet}, ϕ_{jet} – pseudopospieszność i kąt azymalny osi dżetu
 - 510 – $p_{T,jet}$ – pęd poprzeczny dżetu
 - 511 – M_{jet} – masa dżetu [#REF]

- 512 – A_{jet} – powierzchnia dżetu – liczona w płaszczyźnie (η, ϕ), do powierzchni dżetu zaliczany jest element w powierzchni w którym dodanie części o nieskończenie małym pędzie poprzecznym sprawi, że zostanie ona zaliczona do tego dżetu [50]
- 513 – ρ_{bckg} – gęstość tła w danym zdarzeniu

516 • Zmienne opisujące cząstki tworzące dżet:

- 517 – η, ϕ – pseudopospieszność i kąt azymutalny cząstki względem osi dżetu
- 518 – p_T – pęd poprzeczny cząstki
- 519 – IP_D – rzut na kierunek poprzeczny wektora parametru zderzenia
- 520 – IP_Z – rzut na oś z wektora parametru zderzenia
- 521 – $N_{Constit}$ – liczba cząstek tworzących dżet

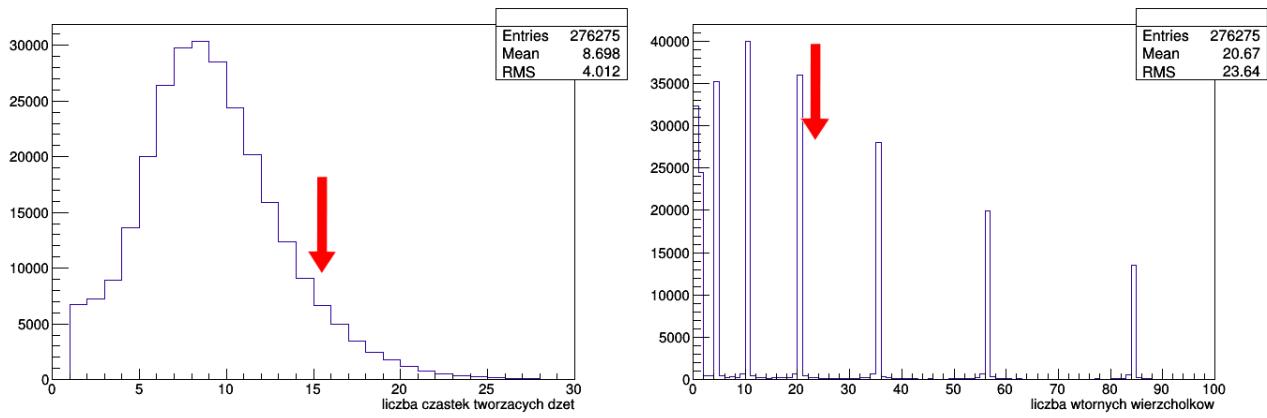
522 • Zmienne opisujące wtórne wierzchołki:

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

531 Dżety różnią się liczbą cząstek je tworzących oraz liczbą wtórnych wierzchołków. Większość algorytmów uczenia maszynowego wymaga natomiast dostarczenia danych w postaci tabelarycznej (macierzowej), ze stałą liczbą kolumn (wiersze stanowią kolejne dżety). Aby spełnić to wymaganie konieczne jest przyjęcie pewnej ustalonej liczby wtórnych wierzchołków oraz cząstek tworzących dżet – w przypadku gdy dżet ma więcej elementów tego typu są one odrzucane, natomiast puste pola są wypełniane zerami w przypadku gdy ma ich mniej. Po przeanalizowaniu rozkładów liczby wtórnych wierzchołków i cząstek tworzących dżet (Rys. 11) oraz wstępny sprawdzeniu jak dodawanie kolejnych elementów wpływa na otrzymywane wyniki (na podstawie wzmacnianych drzew decyzyjnych ze względu na wspomnianą w 2.3 szybkość i stabilność) ustalono liczbę cząstek tworzących dżet równą 15 a wtórnych wierzchołków równą 20.

541 Istotnym zagadnieniem jest także kolejność w jakiej ułożone będą zmienne. Dla sieci konwolucyjnych szukających lokalnych zależności rozsądne jest ułożenie obok siebie tych samych zmiennych, np. $L_{xy,1}, L_{xy,2}, L_{xy,3} \dots \sigma_{vertex,1}, \sigma_{vertex,2}, \sigma_{vertex,3} \dots$. Dla sieci w pełni połączonych oraz drzew decyzyjnych kolejność zmiennych nie ma znaczenia, ale ważne jest aby ich położenie było stałe, np. aby L_{xy} i $\sigma_{L_{xy}}$ danego wtórnego wierzchołka były w tych samych miejscach, tak aby możliwe było szukanie zależności między nimi.

547 Nastecną kwestią jest wybór wielkości decydującej o kolejności ułożenia elementów, tj. która cząstka będzie cząstką nr 1 a która nr 5. Losowe ułożenie elementów sprawiłoby, że bezpośrednie porównywanie wielkości w danych kolumnach (co ma miejsce bezpośrednio w drzewach decyzyjnych a pośrednio w sieciach neuronowych) straciłoby sens. Z kolei dobry dobór tej kolejności pozwala na łatwe odtworzenie przez algorytm uczenia maszynowego motywów fizycznie algorytmów omówionych w sekcji 1.4.2. Przykładowo cięcie na wartość IP drugiej lub trzeciej cząstki (gdy są one posortowane wg malejących wartości IP) jest istotą algorytmu nazywanego *Track Counting – TC* stosowanego w CMS i ALICE. Kolejność w jakiej ułożone będą elementy,



Rysunek 11: Rozkłady liczby cząstek tworzących dżet i liczby wtórnych wierzchołków (część) wraz wartościami cięć.

wpływa także na to, które z nich będą odrzucone w przypadku gdy dżet zawiera więcej niż 15 cząstek i 20 wierzchołków. Ponownie posiłkowano się testami z użyciem drzew decyzyjnych.
Ostatecznie wtórne wierzchołki ułożono według malejącego L_{xy} a cząstki – malejącego p_T .

558 4 Analiza

559 4.1 Dobór metryki

560 Bardzo ważnym elementem w trenowaniu algorytmów uczenia maszynowego jest dobór odpowiadającej metryki – klasycznym złym przykładem jest używanie dokładności (ang. *accuracy*) do 561 oceniania klasyfikacji binarnej w przypadku dużego niezrównoważenia klas – algorytm przewidujący zawsze klasę większościową może osiągnąć dużą wartość dokładności będąc jednocześnie 562 bardziej słabym modelem.

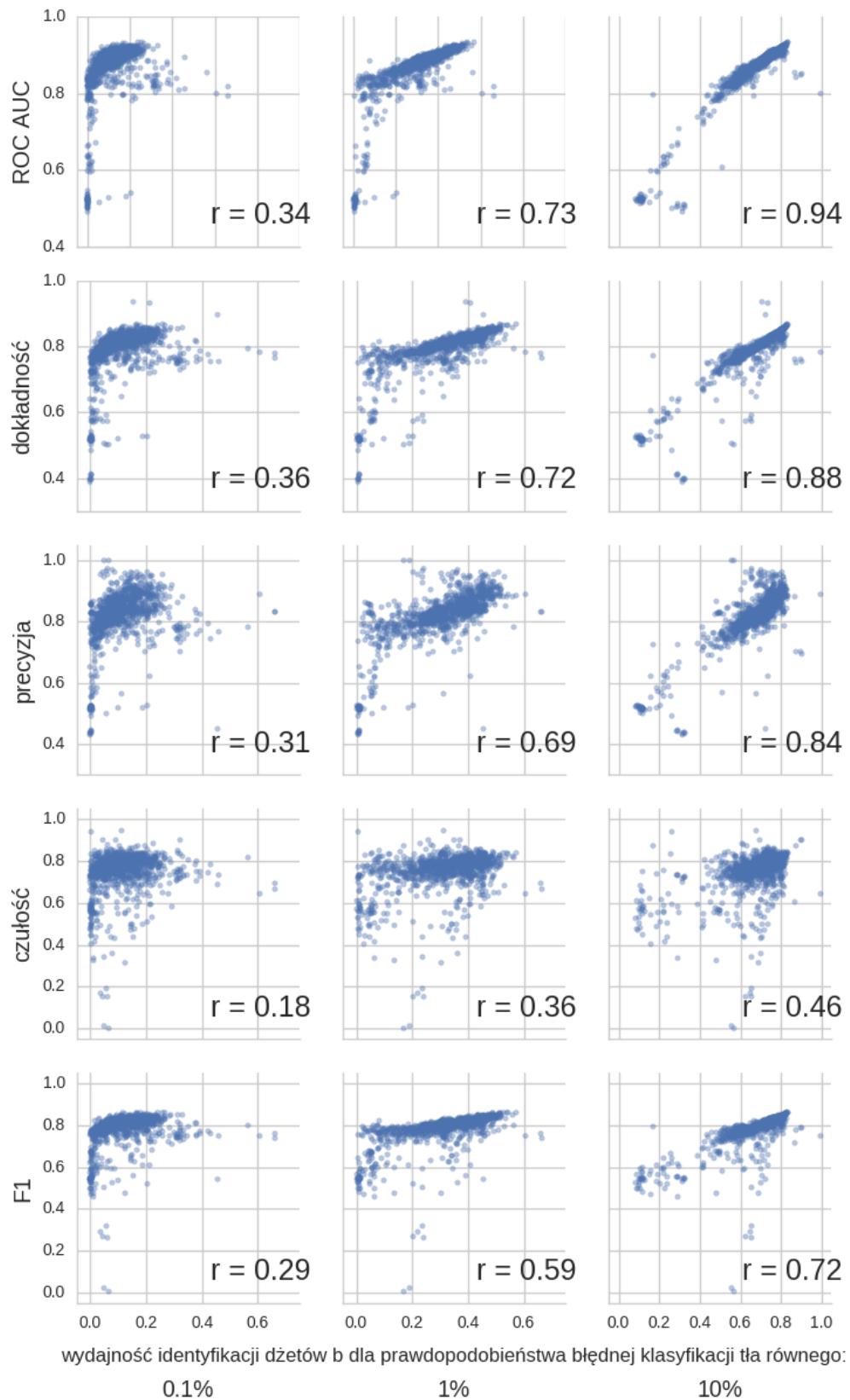
563 Kilka najczęściej używanych metryk wymieniono w Tab. A1. Używanie i porównywanie kilku 564 miar efektywności jest często niepraktyczne dlatego dobrze jest wybrać jedną metrykę. Przy jej 565 wyborze należy kierować się potencjalnymi zastosowaniami modelu. W tym przypadku 566 są to analizy fizyczne, które mogą mieć różne wymagania dotyczące czystości i liczebności 567 otrzymywanych próbek a co za tym idzie, preferować inne punkty pracy zdefiniowane jako pary 568 liczb: wydajność poprawnej klasyfikacji dżetów b (ang. *tagging efficiency = true positive rate* 569 $= recall$), i ułamek niepoprawnie zaklasyfikowanych przypadków tła (ang. *mistagging rate =* 570 *false positive rate*).

571 Naturalnym wyborem wydaje się pole pod powierzchnią krzywej ROC (ang. *ROC Area Under Curve – ROC AUC*) [51]. Potencjalną przeszkodą może być zakres rozsądnych wartości 572 prawdopodobieństwa błędnej klasyfikacji przypadków tła: dżety b stanowią tylko kilka procent 573 liczby wszystkich dżetów, zatem z punktu widzenia analizy dopuszczalne będą punkty pracy 574 zapewniające wydajność identyfikacji dżetów b ok. 10 – 100 razy większą niż częstość niepoprawnego 575 zaklasyfikowania przypadków tła. Oznacza to, że zdecydowana większość punktów pracy 576 znajdujących się na krzywej ROC jest nie do zaakceptowania – interesujące są tylko te o 577 najwyższych częstościach błędnej klasyfikacji.

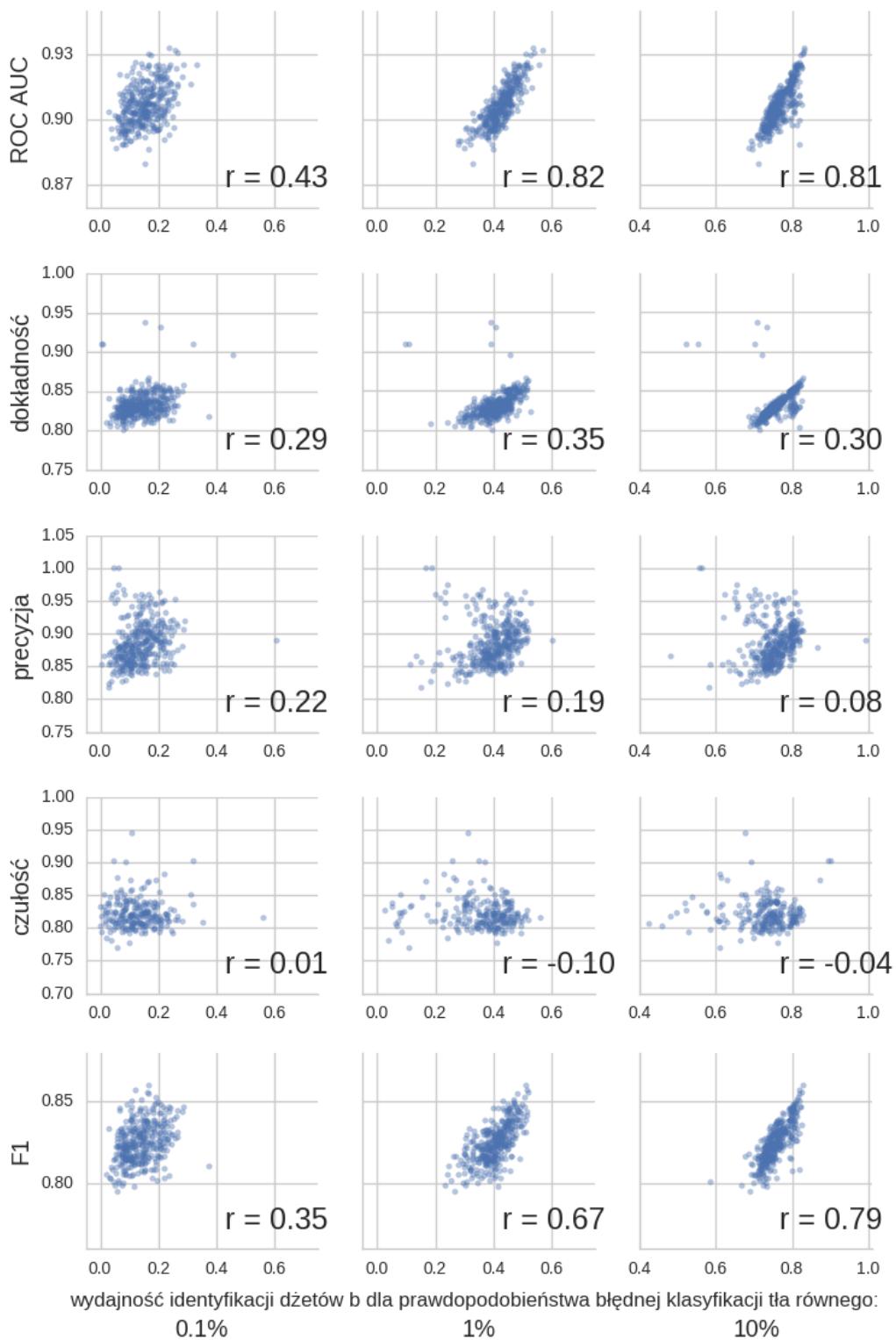
578 Aby ilościowo porównywać różne algorytmy wprowadzone zostaną trzy punkty pracy: o 579 prawdopodobieństwie błędnej klasyfikacji tła równej 0.1%, 1% oraz 10%. Na Rys. 12 przedstawione 580 zostały zależności poszczególnych metryk od wydajności identyfikacji dżetów b w tych 581 trzech punktach pracy. Każdy punkt odpowiada jednemu eksperymentowi (dla dowolnego algorytmu) 582 przeprowadzonemu w trakcie przygotowywania analizy. Daje to pogląd, na to która 583 metryka zapewni jednocześnie wysokie wartości wydajności na identyfikację dżetów b w wybranych 584 punktach pracy. Najwyższe koreacje występują dla punktu pracy o najwyższym prawdopodobieństwie 585 błędnej klasyfikacji tła – jak będzie to pokazane później wyniki dla tego punktu pracy 586 są najbardziej stabilne. Spośród analizowanych metryk najwyższe wartości współczynnika 587 Pearsona otrzymano dla pola pod powierzchnią krzywej ROC, dosyć wysokie również dla 588 dokładności i precyzji. Widać, że wartości czułości są najsłabiej skorelowane z wydajnością 589 identyfikacji dżetów b – jest zrozumiałe, że są one dużo niższe niż dla precyzji: wartości czułości 590 maleją gdy błędnie klasyfikowane są dżety b stanowiące sygnał, podczas gdy wartości precyzji 591 maleją, gdy błędnie klasyfikowane są przypadki tła. Z tych dwóch błędów, drugi jest bardziej 592 kosztowny, gdyż błędna klasyfikacja nawet niewielkiej części tła znacznie pogarsza czystość 593 otrzymywanej próbki.

594 Na Rys. 13 przedstawiono te same wykresy, ale tym razem wybrano tylko 25% najlepszych 595 wartości dla każdej metryki – te punkty są bardziej znaczące, gdyż ostatecznie modele z 596 eksperymentów dających najlepsze wyniki będą używane. Dla tych wykresów otrzymano zdecydowaną 597 dominację *ROC AUC* – wybór modeli dających najwyższe pole pod powierzchnią krzywej ROC 598 zapewnia jednocześnie otrzymanie wysokich wartości wydajności identyfikacji dżetów b dla 599 wybranych punktów pracy.

600 Pole pod powierzchnią krzywej *ROC* zostało wybrane jako główna metryka używana w 601 procesie dobierania parametrów modeli oraz przy prezentacji wyników.



Rysunek 12: 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 13: Rysunek podobny do Rys. 12, ale przedstawione zostały tylko punkty odpowiadające eksperymentom o wartościach metryki będących w górnym kwartylu wartości danej metryki dla wszystkich eksperymentów.

605 **4.2 Wyniki dla poszczególnych modeli**

606 W następnych podrozdziałach przedstawione zostały wyniki uzyskane dla poszczególnych mo-
607 deli. Jako model rozumiana jest para: algorytm (wraz z jego hiperparametrami) oraz zestaw
608 danych użyty do jego trenowania. Oznaczenia używane w prezentacji wyników zebrane zostały
609 w Dodatku B.

610 Każdy przedstawiony wynik jest rezultatem uśrednienia pięciu powtórzeń procedury, na
611 którą składa się: losowy podział zbioru danych na dane treningowe, walidacyjne i testowe oraz
612 uczenie algorytmu (randomizacji podlega także inicjalizacja wag połączeń w przypadku sieci
613 neuronowych).

614 Na Rys. 14, 15, 16 przedstawiono odmianę krzywej ROC – wykresy przedstawiające zależ-
615 ności wydajności identyfikacji dżetów b z niepewnościami (na osi poziomej) dla danego praw-
616 dopodobieństwa błędnej klasyfikacji dżetów tła (na osi pionowej).

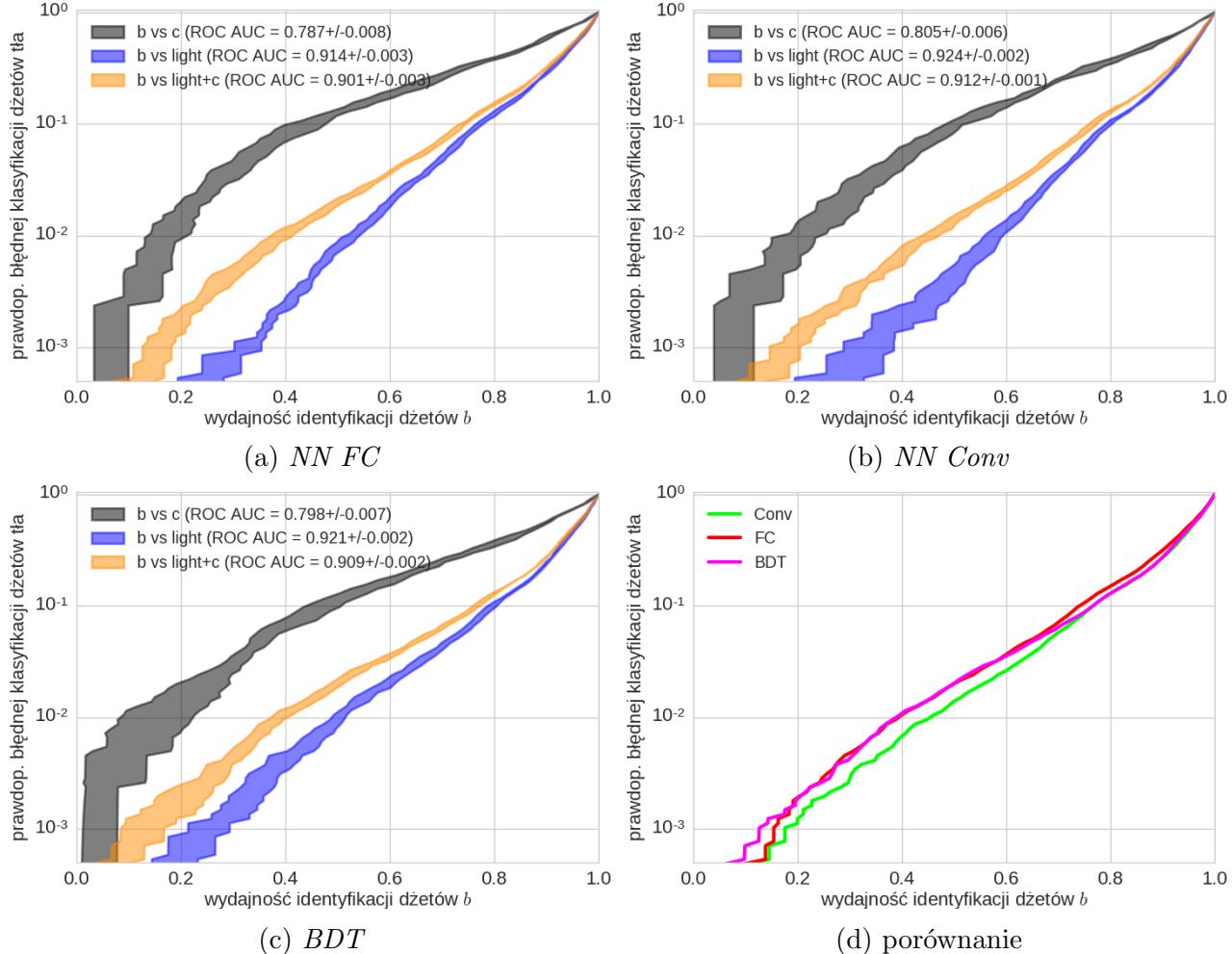
617 Trzy krzywe na każdym wykresie odpowiadają separacji dżetów b od dżetów: lekkich, c
618 oraz mieszanej próbki złożonej w 90% z dżetów lekkich i w 10% z dżetów powabnych. Cztery
619 wykresy na każdym rysunku odpowiadają trzem użytym algorytmom: sieciom neuronowym
620 typu FC (na górze po lewej), konwolucyjnym sieciom neuronowych (na górze po prawej) i
621 wzmacnianym drzewom decyzyjnym (na dole po lewej) oraz porównaniu wszystkich trzech
622 (na dole po prawej). Porównane zostały tylko krzywe odpowiadające separacji dżetów b od
623 mieszanego tła (bez niepewności).

624 Jeśli nie zaznaczono inaczej, prezentowane wyniki dotyczą separacji dżetów b od mieszanego
625 tła (90% dżetów lekkich + 10% c).

626 W Tab. 1 przedstawiono podsumowanie wyników.

627 4.2.1 Wyniki dla zmiennych związanych z wtórnymi wierzchołkami

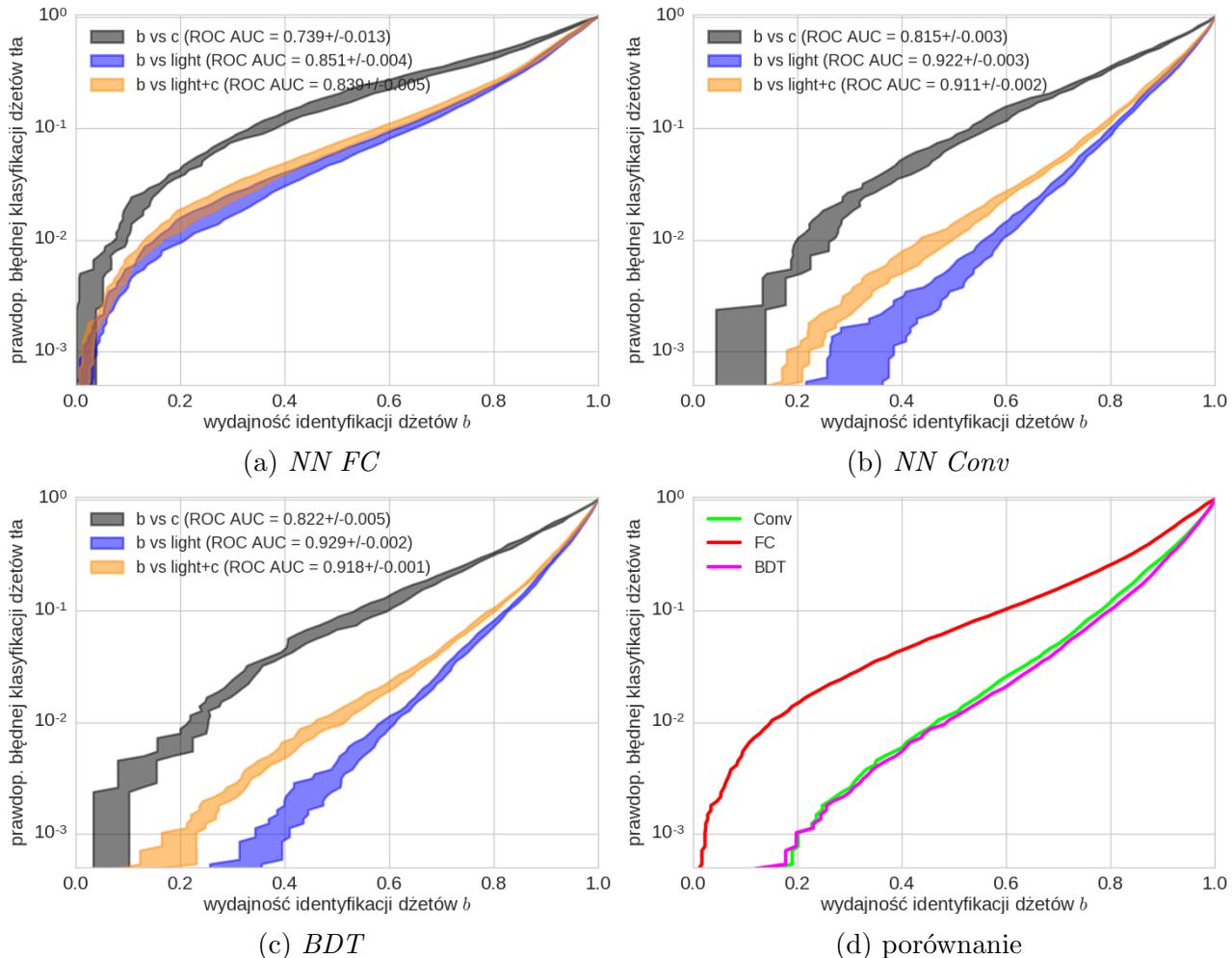
628 Na Rys. 14 przedstawiono rezultaty uzyskane przy trenowaniu na zbiorze danych *SV*. Uzyskano
 629 bardzo zbliżone wyniki dla wszystkich trzech algorytmów, sieci konwolucyjne były nieznacznie
 630 skuteczniejsze przy wydajnościach na identyfikację dżetów b poniżej 70% – dla tych samych
 631 prawdopodobieństw błędnej klasyfikacji tła uzyskiwały wydajności identyfikacji lepsze o ok.
 632 5%.



Rysunek 14: Zależności wydajności identyfikacji dżetów b od prawdopodobieństwa błędnej klasyfikacji dżetów tła dla poszczególnych algorytmów oraz ich porównanie. Algorytmy wytrenowane na zmiennych *SV*.

633 4.2.2 Wyniki dla zmiennych związanych z częstками tworzącymi dżet

634 Na Rys. 15 przedstawiono rezultaty uzyskane przy trenowaniu na zbiorze danych *constit*. Wyniki
 635 dla *BDT* oraz *Conv* są niemal identyczne, dla *BDT* trochę lepsze niż w przypadku zestawu da-
 636 nych *SV*, natomiast te uzyskane dla sieci neuronowych typu *FC* są znacznie gorsze. Stosunkowo
 637 najmniejsze różnice pomiędzy zestawami danych *SV* i *constit* dla algorytmu *FC* otrzymano w
 638 przypadku separacji dżetów *b* od *c*.

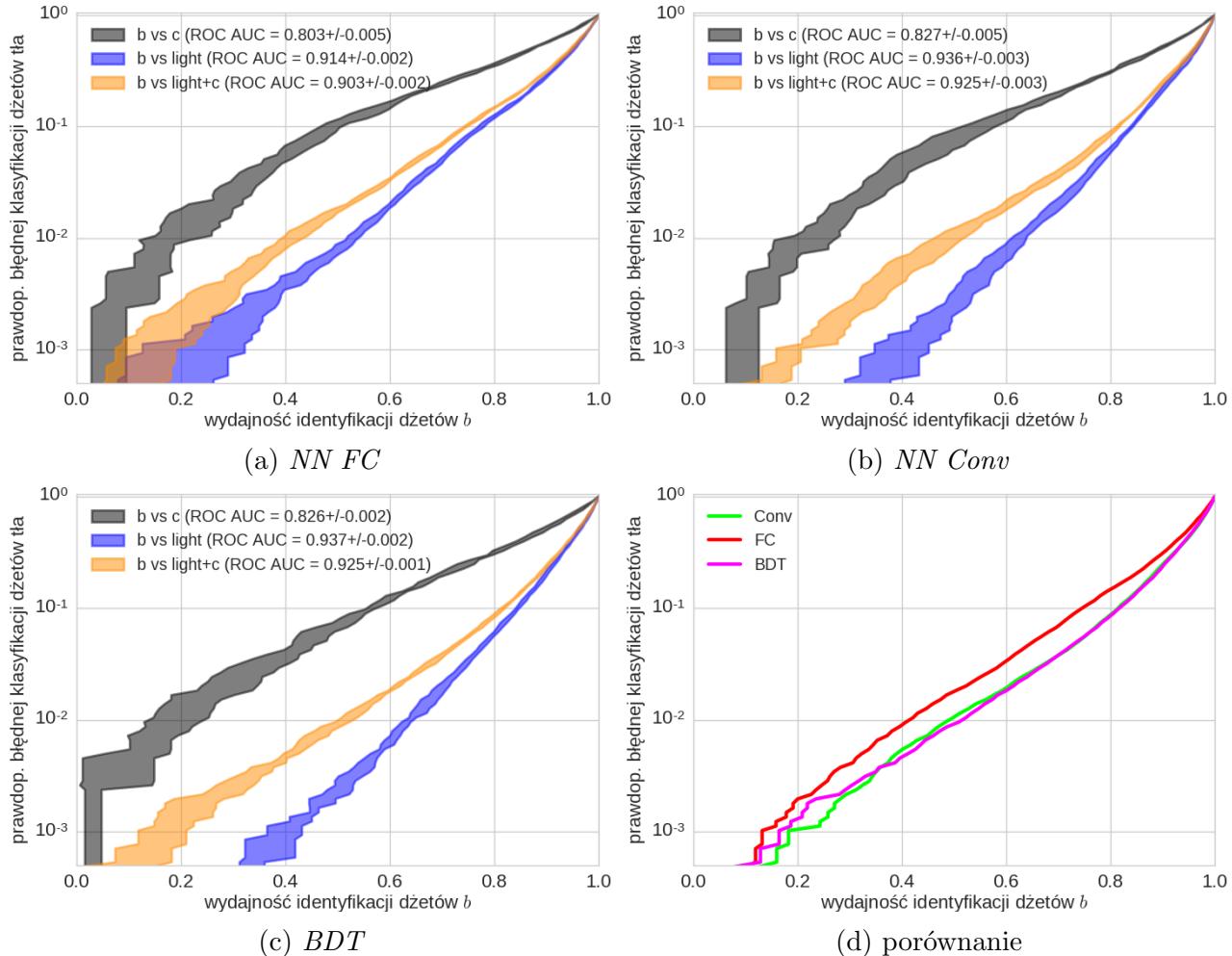


Rysunek 15: Zależności wydajności identyfikacji dżetów *b* od prawdopodobieństwa błędnej klasyfikacji dżetów tła dla poszczególnych algorytmów oraz ich porównanie. Algorytmy wytreno-
 wane na zmiennych *constit*.

639 4.2.3 Wyniki dla wszystkich zmiennych

640 Na Rys. 16 przedstawiono rezultaty uzyskane przy trenowaniu na zbiorze danych *merged*. Po-
 641 nownie krzywe ROC wzmacnianych drzew decyzyjnych i sieci konwolucyjnych prawie się nie
 642 różnią, natomiast dla sieci w pełni połączonych krzywa jest przesunięta o ok. 0.1 #### w stronę
 643 niższych wydajności.

644 W porównaniu do poprzednich zbiorów danych wszystkie algorytmy poprawiły swoje pre-
 645 dykce, najmniej *FC*, którego zdolności separacyjne są prawie takie same jak w przypadku *SV*
 646 (trochę lepsza separacja *b* od *c*).



Rysunek 16: Zależności wydajności identyfikacji dżetów *b* od prawdopodobieństwa błędnej klasyfikacji dżetów tła dla poszczególnych algorytmów oraz ich porównanie. Algorytmy wytrenowane na zmiennych *merged*.

647 4.2.4 Podsumowanie

648 Uzyskane wartości $ROC AUC$ oraz wydajności identyfikacji dżetów b dla trzech punktów pracy
 649 zebrano w Tab. 1.

650 Tak jak było to widać na Rys. 13, z $ROC AUC$ najbardziej skorelowane są wydajności dla
 651 punktu pracy o najwyższych wydajnościach. Wynika to z faktu, że pierwsze dwa punkty pracy
 652 leżą na samym początku krzywej ROC (dają niewielki wkład do pola pod krzywą).

653 Algorytm wytrenowany na połączonym zbiorze danych daje trochę lepsze wyniki niż trenowa-
 654 ny na zbiorach SV i $constit$, co jest oczywiście oczekiwane. Można było natomiast spodziewać
 655 się większej poprawy, co pokazuje, że predykcje algorytmów trenowanych na SV oraz $constit$
 656 muszą być skorelowane (przy założeniu, że model trenowany na dwóch połączonych zbiorach
 657 danych daje wyniki niegorsze niż trywialne połączenie dwóch modeli trenowanych na osobnych
 658 zbiorach danych).

659 W każdym przypadku najgorsze wyniki uzyskano dla sieci w pełni połączonych. Z tabeli
 660 wynika, że duże znaczenie mają zarówno użyty algorytm jak i zestaw danych. Wysokie po-
 661 dobieżstwo wyników uzyskiwanych dla BDT oraz $Conv$ pozwala przypuszczać, że uzyskiwane
 662 przez nie poziom błędu jest zbliżony do minimum osiągalnego przy tych zestawach danych
 663 (błędu *Bayesowskiego*).

model	$ROC AUC$	wydajność identyfikacji dżetów b [%]		
		dla prawd. błędnej klas. tła równej	0.1%	1%
$SV-FC$	0.901 ± 0.003	15 ± 3	39 ± 2	73.7 ± 0.6
$SV-Conv$	0.912 ± 0.001	18 ± 3	45 ± 2	76.4 ± 0.8
$SV-BDT$	0.909 ± 0.002	13 ± 4	38 ± 1	76.3 ± 0.6
$constit-FC$	0.839 ± 0.005	2 ± 1	15 ± 2	58.6 ± 1.5
$constit-Conv$	0.911 ± 0.002	20 ± 2	46 ± 3	77.8 ± 0.6
$constit-BDT$	0.918 ± 0.001	20 ± 3	48 ± 3	79.4 ± 0.7
$merged-FC$	0.903 ± 0.002	13 ± 6	41 ± 2	74.2 ± 0.3
$merged-Conv$	0.925 ± 0.003	18 ± 2	48 ± 3	81.3 ± 0.4
$merged-BDT$	0.925 ± 0.001	16 ± 5	51 ± 1	81.4 ± 0.4

Tablica 1: Tabela podsumowująca wyniki uzyskane przez poszczególne modele.

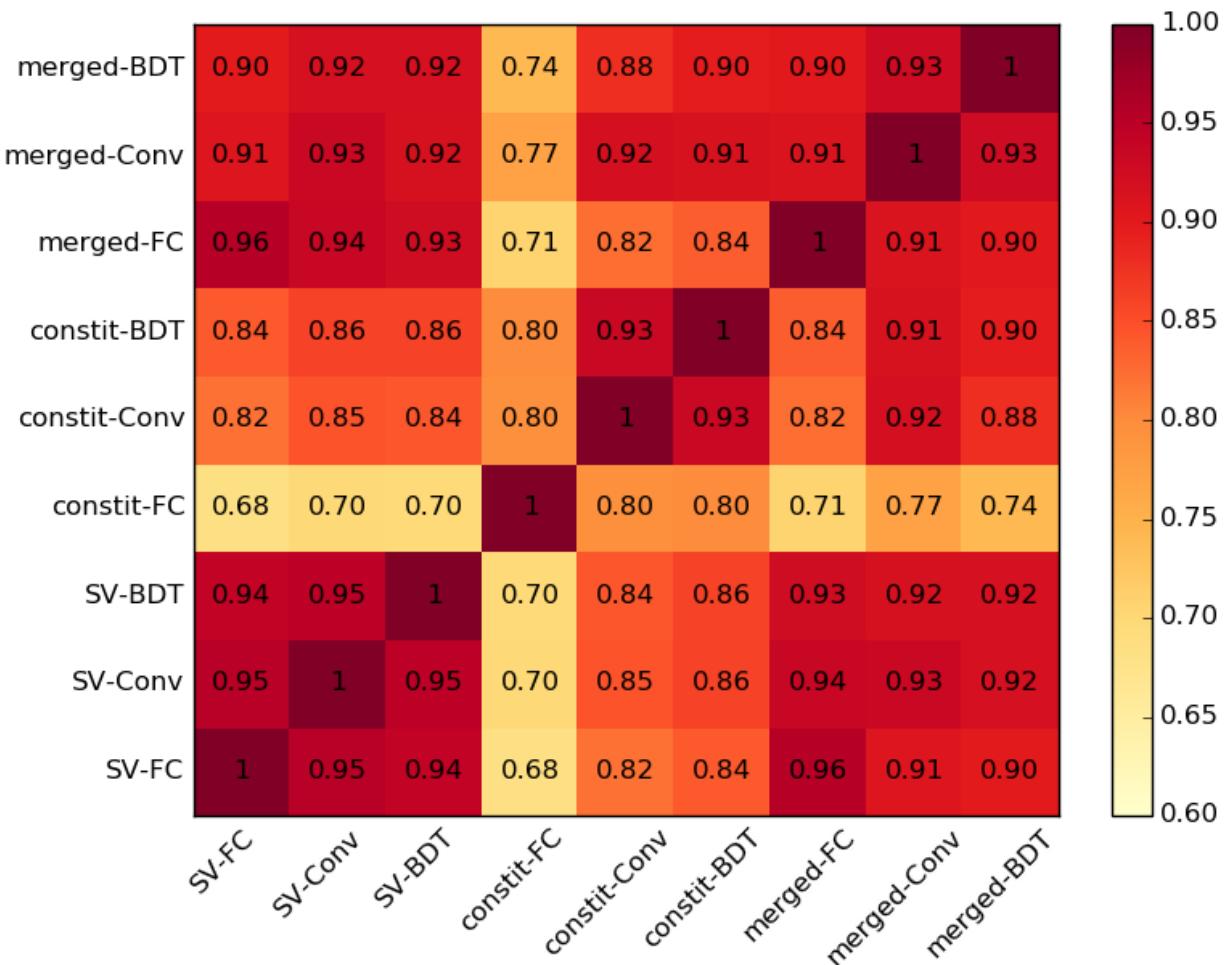
664 4.3 Korelacje predykcji modeli

665 Na Rys. 17 przedstawiono korelacje pomiędzy poszczególnymi modelami. Korelacje obliczano
 666 pomiędzy wynikami zwracanym bezpośrednio przez algorytmy (liczba rzeczywista $\in [0, 1]$), nie
 667 na podstawie odpowiadających im klas ($\{0, 1\}$).

668 Pierwszy wniosek płynący z wykresu korelacji to fakt, że wszystkie modele są ze sobą silnie
 669 skorelowane. Tłumaczy to niewielki zysk płynący z połączenia zestawów danych *SV* i *constit*
 670 co wspomniano w podrozdziale 4.2.4.

671 Najmniejszymi korelacjami z innymi wyróżnia się model *constit-FC*, który dawał zdecydowanie
 672 najsłabsze wyniki.

673 Widać wyraźny wzrost korelacji w przypadku modeli trenowanych na tych samych zbiorach
 674 danych (szczególnie dla modeli trenowanych na zmiennych związanych z wtórnymi wierzchołkami). Modele trenowane na połączonym zbiorze danych są nieco silniej skorelowane z modelami
 676 *SV-X* niż *constit-X*. Nie obserwuje się natomiast istotnie większych korelacji między modelami
 677 wykorzystującymi ten sam algorytm.



Rysunek 17: Średnie współczynniki korelacji Pearsona pomiędzy predykcjami poszczególnych modeli. Odchylenia standardowe wynosiły 0.001 – 0.009.

678 4.4 Analiza istotności zmiennych w *BDT*

679 Jedną z zalet drzew decyzyjnych jest możliwość dokładnego prześledzenia procesu decyzyjnego
680 poprzedzającego klasyfikację danego przypadku, co pozwala na lepsze zrozumienie działania
681 modelu. Cecha ta zanika w przypadku stosowania klasyfikatora złożonego z dużej ilości drzew,
682 jak *BDT* jednak nadal istnieją dosyć proste sposoby na zrozumienie, które cechy miały naj-
683 większy wpływ na predykcję.

684 Wykorzystano trzy miary istotności zmiennych zaimplementowane w bibliotece **XGBoost**:⁴
685 *weight* informuje ile razy dana zmienna została wykorzystana w podziałach drzew. *total_gain*
686 jest sumą po spadkach funkcji straty uzyskiwanych dzięki podziałom wykorzystującym daną
687 zmienną, natomiast *total_cover* to suma liczb przykładów trenujących, na które miały wpływ
688 podziały wykorzystujące daną zmienną.

689 Zasadniczo najważniejszą miarą jest *total_gain*, gdyż uwzględnia ona zarówno częstość wyko-
690 rzystania danej zmiennej (*weight*) jak i średni zysk w postaci spadku funkcji straty uzyskiwany
691 w poszczególnych podziałach. Zmienna binarna wykorzystana już w jednym podziale drzewa,
692 nie może zostać użyta ponownie w żadnej gałęzi będącej kontynuacją danego podziału, stąd
693 zmienne binarne zwykle osiągają mniejsze wartości miary *weight*. *total_cover* jest z kolei zwią-
694 zane z głębokościami drzewa na jakiej wykorzystywana była zmienna – pierwszy podział w
695 danym drzewie wpływa zawsze na wszystkie przykłady trenujące, natomiast kolejne – na stop-
696 niowo coraz mniejszy ułamek. Zatem niskie wartości *total_cover* sugerują, że dana zmienna
697 wykorzystywana była raczej na dużych głębokościach (na już mocno podzielonym drzewie).

698 W Tab. 2 przedstawiono wartości miar istotności zmiennych dla 15 najważniejszych cech
699 (wg *total_gain*) dla modeli *SV-BDT* oraz *constit-BDT*. Z kolei w Tab. 3 przedstawiono wartości
700 tych miar pogrupowane wg wielkości fizycznych. Wartości każdej miary zostały unormowane
701 do 100 (suma dla wszystkich elementów, nie tylko tych przedstawionych w tabelach jest równa
702 100). System oznaczeń opisany jest w Dodatku B.

703 Zdecydowanie najważniejszymi zmiennymi okazały się: L_{xy} w przypadku wtórnych wierz-
704 chołków oraz IP_D w przypadku części składowych dżetu, czyli zmienne związane z długim
705 czasem życia pięknych hadronów. Wyraźna jest także tendencja, że im wyższa pozycja elementu
706 na liście, tym istotniejsze są jego właściwości (najistotniejsze są elementy nr 0).

707 O ile miary *total_gain* i *total_cover* są dosyć silnie skorelowane (co widać w obu tabelach)
708 to *weight*, czyli częstość ich użycia wydaje się być niezależna i przyjmować zbliżone wartości
709 dla wszystkich zmiennych. Okazuje się, że nie ma zmiennych, które byłyby wyraźnie rzadziej
710 wykorzystywane od innych.

711 Pojawiają się także niespodziewane wyniki, przykładowo zmienna $\sigma_{L_{xy}}$ pojawia się częściej
712 niż L_{xy} , której niepewność opisuje, co jest zaskakujące tym bardziej, że jest $\sigma_{L_{xy}}$ tylko jednym z
713 trzech rodzajów zmiennych związanych z jakością wtórnego wierzchołka ($\sigma_{L_{xy}}$, σ_{vertex} i χ^2/Ndf).

714 Wyniki uzyskane dla modelu *merged-BDT* (prawa strona Tab. 3) pokazują, że przy treно-
715 waniu na pełnym zestawie danych zmienne ze zbioru *SV* są bardziej znaczące niż te ze zbioru
716 *constit*, co jest zgodne z odnotowaną wyższą korelacją modeli *merged-X* z modelami *SV-X* niż
717 z modelami *constit-X*.

⁴nazwy miar pochodzą z wersji biblioteki dla języka Python, w wersji dla innych języków, np. R nazwy miar oraz ich znaczenie są różne

	<i>weight</i>	trening osobno <i>total_gain</i>	<i>total_cover</i>
L_{xy} - SV0	2.76	13.37	4.68
L_{xy} - SV2	1.88	9.83	3.80
L_{xy} - SV3	1.66	7.68	3.51
L_{xy} - SV1	1.34	6.79	2.84
L_{xy} - SV5	0.77	3.79	1.78
L_{xy} - SV4	1.03	2.79	2.06
χ^2/Ndf - SV0	3.30	2.30	2.56
L_{xy} - SV6	0.96	2.21	1.75
L_{xy} - SV8	0.96	2.14	1.81
L_{xy} - SV7	0.89	1.99	1.59
N_{SV}	0.73	1.73	1.33
σ_{vertex} - SV0	2.70	1.40	2.08
σ_{Lxy} - SV9	0.91	1.36	1.38
χ^2/Ndf - SV3	1.91	1.33	2.14
M_{inv} - SV9	0.77	1.31	1.24
IP_D - C0	2.73	19.29	9.68
IP_D - C1	2.76	15.06	8.74
IP_D - C2	3.39	11.37	8.69
IP_D - C3	2.84	6.56	6.16
IP_Z - C0	3.01	5.26	5.86
IP_D - C4	3.02	3.90	5.37
IP_Z - C1	2.50	3.44	4.63
IP_Z - C2	2.29	2.28	3.55
IP_D - C5	1.95	1.67	3.15
IP_Z - C3	2.14	1.50	2.65
IP_D - C6	2.07	1.33	2.96
IP_Z - C4	1.95	1.10	2.16
p_T - C0	2.22	1.00	1.79
p_T - C2	2.14	0.98	2.04
ϕ - C0	2.31	0.98	1.27

Tablica 2: Tabela zawierająca wartości miar istotności 15 najważniejszych (wg *total_gain*) cech, osobno dla zmiennych związanych z wtórnymi wierzchołkami (górsza połowa) oraz z częstotliwościami składowymi (dolna połowa).

	trening osobno			trening razem		
	<i>weight</i>	<i>total_gain</i>	<i>total_cover</i>	<i>weight</i>	<i>total_gain</i>	<i>total_cover</i>
L_{xy}	17.2	54.5	28.9	6.9	21.1	12.9
σ_{Lxy}	19.4	13.1	18.1	7.2	8.5	7.9
M_{inv}	18.3	8.0	14.1	7.6	6.8	6.8
σ_{vertex}	17.0	8.2	14.2	6.6	6.0	6.9
χ^2/Ndf	19.3	12.8	19.5	7.1	6.6	7.9
IP_D	22.4	60.6	47.5	11.3	12.3	16.1
IP_Z	18.3	16.1	23.3	9.9	7.7	9.3
p_T	14.0	7.1	14.1	7.0	5.7	6.2
ϕ	23.2	8.7	8.2	12.4	8.4	8.2
η	21.7	7.3	6.4	13.0	8.8	8.9

Tablica 3: Tabela zawierająca wartości miar istotności cech, posumowane według rodzaju zmiennej (sumy po wszystkich wtórnych wierzchołkach / wszystkich cząstkach). Wartości w lewej części odpowiadają modelom *SV-BDT* i *constit-BDT* natomiast z prawej – *merged-BDT*.

718 4.5 Analiza wpływu zmiennych na predykcje modeli

719 Dalszą analizę wpływu poszczególnych zmiennych przeprowadzono przy użyciu wykresów zależności cząstkowych (ang. *partial dependence plots*). Ze względu na wysoki koszt obliczeniowy tej metody, przedstawione wyniki wyjątkowo nie są uśrednieniem kilka powtórzeń całej procedury.

720 Metoda wykresów zależności cząstkowych to metoda polegająca na sprawdzeniu zachowania się predykcji już wytrenowanego modelu na skutek zmiany wartości wybranej zmiennej. W tym celu przeprowadzane są predykcje na normalnym zbiorze danych, ze zmienioną jedną kolumną, zawierającą wartości analizowanej zmiennej. Wszystkie wartości w tej kolumnie ustawiane są na tę samą wartość, która jest stopniowo zmieniana od minimum do maksimum wartości przyjmowanych przez daną zmienną, za każdym razem przeprowadzana jest predykcja. Współrzędnymi punktów na wykresie zależności cząstkowej są: kolejne wartości ustawiane w analizowanej kolumnie (oś pozioma) i średnia wartość predykcji modelu (oś pionowa). Im wyższe wartości tej drugiej tym większa część przykładów zostały zaliczona przez algorytm do klasy pozytywnej (uznana za dżet b).

721 Taka analiza służy kilku celom. Po pierwsze, pozwala na weryfikację, tego czy modele działają zgodnie z naszą wiedzą, w przypadkach gdy wiadomo jak powinien zachować się algorytm. Odwrotna sytuacja (brak wiedzy na temat pożądanego zachowania) pozwala na wyciągnięcie wniosków na podstawie sposobu działania algorytmu co wzbogaca intuicję i zrozumienie analizowanych danych. Po trzecie, jest to jeden ze sposobów na zrozumienie działania algorytmów, w przypadku których w przeciwieństwie do drzew decyzyjnych czy regresji liniowej nie istnieją proste metryki pozwalające na oszacowanie wpływu poszczególnych zmiennych na predykcję. W przypadku sieci neuronowych często jest to dużym wyzwaniem.

722 Na Rys. 18 i 19 przedstawiono wykresy zależności cząstkowej dla wybranych zmiennych. Przy wyborze zmiennych posługiwano się wartościami miar istotności cech z sekcji 4.4, kierowano się także wartością ilustracyjną dla omawianych zagadnień. Skala osi pionowej jest inna dla każdego wykresu. Poniżej każdego wykresu przedstawiono rozkłady prawdopodobieństwa analizowanej zmiennej dla dżetów b oraz dżetów tła. Warto zaznaczyć na wstępie, że najbardziej istotne są obszary wykresów w okolicach maksimum rozkładów prawdopodobieństwa zmiennych.

723 Wykresy na Rys. 18 a) i b) oraz 19 a), b) i c) pokazują, że znane podstawowe cechy dżetów b pozwalające odróżniać je od tła, tj. duże wartości L_{xy} wtórych wierzchołków oraz duże wartości bezwzględne parametrów zderzenia cząstek (IP_D , IP_Z) są intensywnie wykorzystywane przez wszystkie algorytmy. Także wartości przy jakich najsilniej zmieniają się predykcje modeli znajdują odzwierciedlenie w rozkładach poniżej.

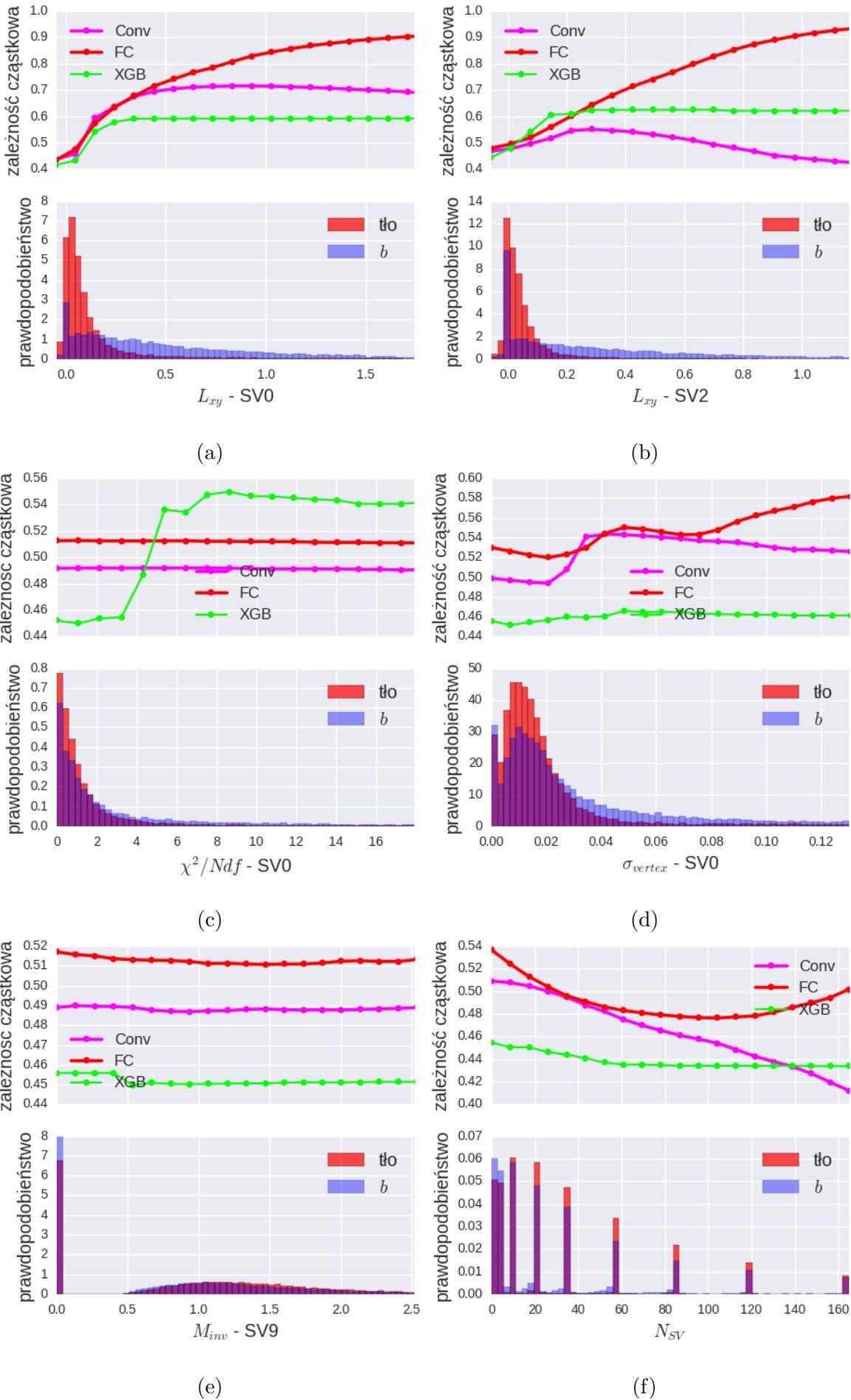
724 Analiza tych pięciu wykresów pozwala na zaobserwowanie ciekawych różnic między algorytmami. Pierwsza z nich: zależności cząstkowe wydają się mieć mniejszą amplitudę w przypadku BDT niż w przypadku sieci neuronowych. Może wynikać to z kształtu funkcji – w przypadku BDT jest to złożenie kilku funkcji schodkowych, podczas gdy sieci neuronowe dają bardziej gładkie zależności. Efektem tego jest inne zachowanie w obszarze ekstrapolacji (obszary gdzie rozkłady prawdopodobieństwa mają niską gęstość). Drzewa decyzyjne nie mają potrzeby dalszych podziałów np. w zakresie $L_{xy} - SV0 > 0.3$ lub $|IP_D| > 5$ dlatego ich predykcje są w tych zakresach niemal stałe. Z kolei w przypadku sieci neuronowych, z powodu braku wystarczającej liczby przykładów o takich wartościach zmiennych, wykres jest w przybliżeniu ekstrapolacją zachowania z zagościęzonego zakresu, co jest bardzo wyraźnie w przypadku sieci w pełni połączonych.

725 Druga obserwacja to to asymetria w predykcji BDT na wykresie $IP_Z - C1$. Pomimo wyjątkowo symetrycznego rozłożenia wartości tej zmiennej wokół zera, predykcje BDT są wyraźnie przesunięte, z minimum zlokalizowanym wokół wartości -1. Przyczyną takiego zachowania może

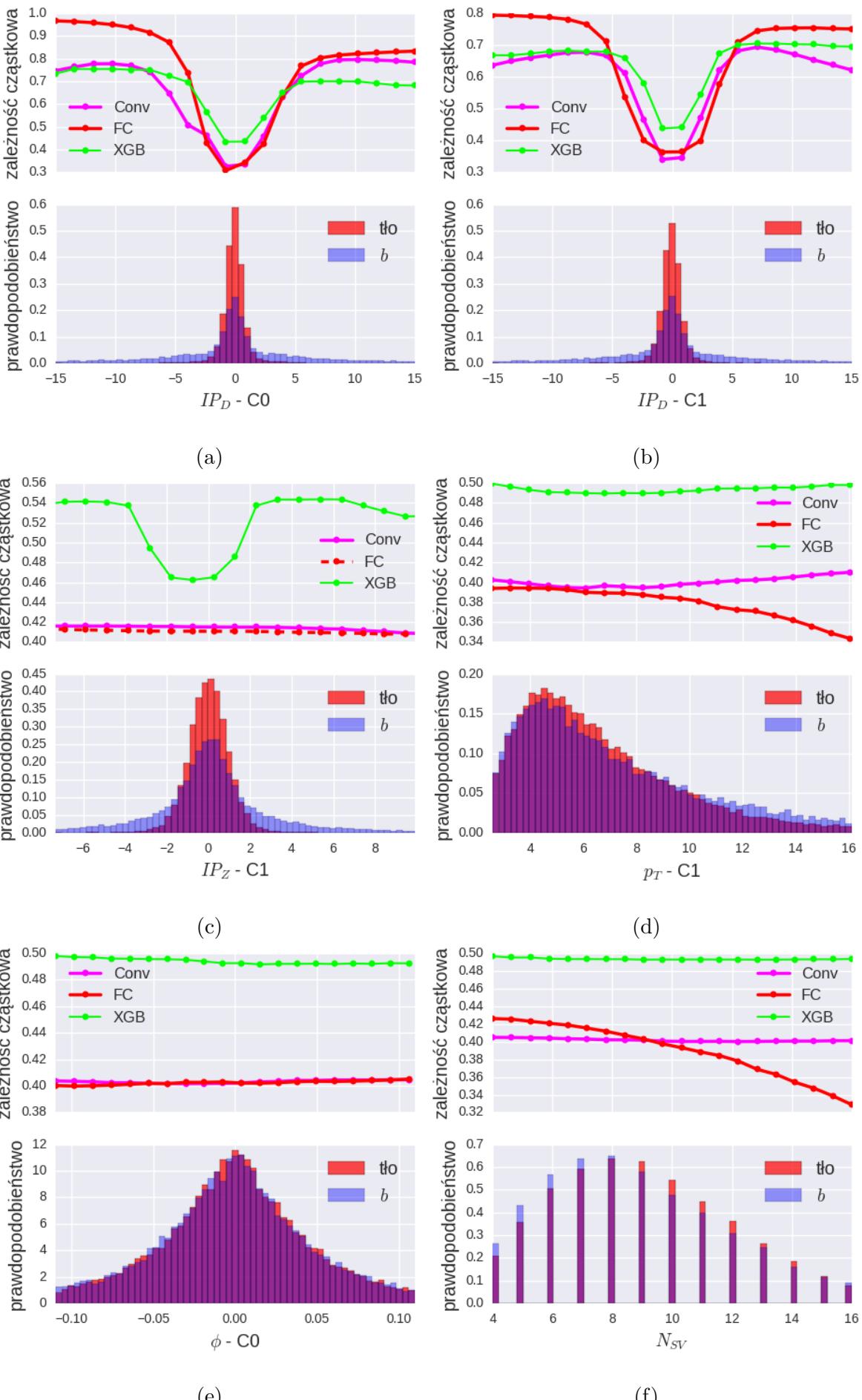
766 być specyficzny sposób trenowania drzew decyzyjnych, które w przypadku takiego rozkładu nie
767 mogą podzielić przestrzeń fazowej na 3 części w jednym kroku, co byłoby najbardziej korzystne,
768 ale zmuszone są do dwóch kolejnych podziałów, np. wzduż wartości -2 i 2, z czego pierwszy
769 podział musi niejako złamać symetrię rozkładu.

770 Wykresy takie jak te na Rys. 18 c) i f) lub 19 c), d) i f) pokazują, że predykcje algorytmów
771 mogą wykazywać nie tylko ilościowe, ale także jakościowe różnice w reakcji na wariancje wartości
772 jednej zmiennej.

773 Wreszcie wykresy przedstawione na Rys. 18 e) oraz Rys. 19 e) pokazują, że nawet w
774 przypadku zmiennych intensywnie wykorzystywane przy podziałach drzewa (o czym świadczy
775 Tab. 2) zmiana wyłącznie ich wartości ma często znikomy wpływ na predykcje całego modelu.
776 Są to zmienne, których rozkłady dla sygnału i tła prawie się nie różnią. Ich wpływ na pre-
777 dykcje algorytmu ujawnia się dopiero w kombinacji z innymi zmiennymi. Wykorzystanie tych
778 zmiennych jest dużo trudniejsze dla ludzi i między innymi na tym polega przewaga podejścia
779 analizy wielowymiarowej i uczenia maszynowego. Wykresy te sygnalizują jednocześnie ograni-
780 czenie metody wykresów zależności cząstkowych, która jak każda metoda wizualna nie radzi
781 sobie z przestrzeniami zmiennych o większej niż 3-4 liczbie wymiarów.



Rysunek 18: Zależności cząstkowe dla wybranych cech przedstawione dla modeli: *SV-Conv*, *SV-FC* i *SV-BDT*.



Rysunek 19: Zależności cząstkowe dla wybranych cech przedstawione dla modeli: *constit-Conv*, *constit-FC* i *constit-BDT*.

782 Bibliografia

- 783 [1] Donald H. Perkins. "Oddziaływanie międzykarkowe i chromodynamika kwantowa". W:
784 *Wstęp do Fizyki Wysokich Energii*. 2 wydr. PWN, 2005, 171–193.
- 785 [2] David J. Gross i Frank Wilczek. "Ultraviolet Behavior of Nonabelian Gauge Theories".
786 W: *Phys. Rev. Lett.* 30 (1973). [,271(1973)], s. 1343–1346. DOI: 10.1103/PhysRevLett.
787 30.1343.
- 788 [3] H. David Politzer. "Reliable Perturbative Results for Strong Interactions?" W: *Phys. Rev.*
789 *Lett.* 30 (1973). [,274(1973)], s. 1346–1349. DOI: 10.1103/PhysRevLett.30.1346.
- 790 [4] C. Patrignani i in. "Review of Particle Physics". W: *Chin. Phys.* C40.10 (2016), s. 100001.
791 DOI: 10.1088/1674-1137/40/10/100001.
- 792 [5] John C. Collins i M. J. Perry. "Superdense Matter: Neutrons Or Asymptotically Free
793 Quarks?" W: *Phys. Rev. Lett.* 34 (1975), s. 1353. DOI: 10.1103/PhysRevLett.34.1353.
- 794 [6] N. Cabibbo i G. Parisi. "Exponential Hadronic Spectrum and Quark Liberation". W:
795 *Phys. Lett.* 59B (1975), s. 67–69. DOI: 10.1016/0370-2693(75)90158-6.
- 796 [7] D. Boyanovsky, H. J. de Vega i D. J. Schwarz. "Phase transitions in the early and the
797 present universe". W: *Ann. Rev. Nucl. Part. Sci.* 56 (2006), s. 441–500. DOI: 10.1146/
798 annurev.nucl.56.080805.140539. arXiv: hep-ph/0602002 [hep-ph].
- 799 [8] Mark G. Alford i Kai Schwenzer. "What the Timing of Millisecond Pulsars Can Teach
800 us about Their Interior". W: *Phys. Rev. Lett.* 113.25 (2014), s. 251102. DOI: 10.1103/
801 PhysRevLett.113.251102. arXiv: 1310.3524 [astro-ph.HE].
- 802 [9] Vardan Khachatryan i in. "Evidence for collectivity in pp collisions at the LHC". W:
803 *Phys. Lett.* B765 (2017), s. 193–220. DOI: 10.1016/j.physletb.2016.12.009. arXiv:
804 1606.06198 [nucl-ex].
- 805 [10] Jaroslav Adam i in. "Enhanced production of multi-strange hadrons in high-multiplicity
806 proton-proton collisions". W: *Nature Phys.* 13 (2017), s. 535–539. DOI: 10.1038/nphys4111.
807 arXiv: 1606.07424 [nucl-ex].
- 808 [11] Matteo Cacciari, Gavin P. Salam i Gregory Soyez. "The Anti-k(t) jet clustering algori-
809 thm". W: *JHEP* 04 (2008), s. 063. DOI: 10.1088/1126-6708/2008/04/063. arXiv:
810 0802.1189 [hep-ph].
- 811 [12] Vardan Khachatryan i in. "Charged-particle nuclear modification factors in PbPb and
812 pPb collisions at $\sqrt{s_{NN}} = 5.02$ TeV". W: *JHEP* 04 (2017), s. 039. DOI: 10.1007/
813 JHEP04(2017)039. arXiv: 1611.01664 [nucl-ex].
- 814 [13] Betty Abelev i in. "Centrality Dependence of Charged Particle Production at Large Trans-
815 verse Momentum in Pb–Pb Collisions at $\sqrt{s_{NN}} = 2.76$ TeV". W: *Phys. Lett.* B720 (2013),
816 s. 52–62. DOI: 10.1016/j.physletb.2013.01.051. arXiv: 1208.2711 [hep-ex].
- 817 [14] Carlos A. Salgado i Urs Achim Wiedemann. "Calculating quenching weights". W: *Phys.*
818 *Rev.* D68 (2003), s. 014008. DOI: 10.1103/PhysRevD.68.014008. arXiv: hep-ph/0302184
819 [hep-ph].
- 820 [15] Yuri L. Dokshitzer i D. E. Kharzeev. "Heavy quark colorimetry of QCD matter". W:
821 *Phys. Lett.* B519 (2001), s. 199–206. DOI: 10.1016/S0370-2693(01)01130-3. arXiv:
822 hep-ph/0106202 [hep-ph].
- 823 [16] Georges Aad i in. "Performance of b-Jet Identification in the ATLAS Experiment". W:
824 *JINST* 11.04 (2016), P04008. DOI: 10.1088/1748-0221/11/04/P04008. arXiv: 1512.
825 01094 [hep-ex].

- 826 [17] M. Aaboud i in. “Measurements of b-jet tagging efficiency with the ATLAS detector using
 827 $t\bar{t}$ events at $\sqrt{s} = 13$ TeV”. W: *JHEP* 08 (2018), s. 089. DOI: 10.1007/JHEP08(2018)089.
 828 arXiv: 1805.01845 [hep-ex].
- 829 [18] Serguei Chatrchyan i in. “Identification of b-quark jets with the CMS experiment”. W:
 830 *JINST* 8 (2013), P04013. DOI: 10.1088/1748-0221/8/04/P04013. arXiv: 1211.4462
 831 [hep-ex].
- 832 [19] A. M. Sirunyan i in. “Identification of heavy-flavour jets with the CMS detector in pp
 833 collisions at 13 TeV”. W: *JINST* 13.05 (2018), P05011. DOI: 10.1088/1748-0221/13/
 834 05/P05011. arXiv: 1712.07158 [physics.ins-det].
- 835 [20] Linus Feldkamp. “Study of b-jet tagging performance in ALICE”. W: *J. Phys. Conf. Ser.*
 836 509 (2014), s. 012061. DOI: 10.1088/1742-6596/509/1/012061. arXiv: 1310.2817
 837 [hep-ex].
- 838 [21] Rüdiger Haake. “Machine and deep learning techniques in heavy-ion collisions with ALICE”.
 839 W: *Proceedings, 2017 European Physical Society Conference on High Energy Physics*
 840 (*EPS-HEP 2017*): *Venice, Italy, July 5-12, 2017*. T. EPS-HEP2017. 2017. DOI: 10.22323/
 841 1.314.0498. arXiv: 1709.08497 [physics.data-an]. URL: <https://pos.sissa.it/314/498/pdf>.
- 843 [22] Roel Aaij i in. “Identification of beauty and charm quark jets at LHCb”. W: *JINST* 10.06
 844 (2015), P06013. DOI: 10.1088/1748-0221/10/06/P06013. arXiv: 1504.07670 [hep-ex].
- 845 [23] K. Aamodt i in. “The ALICE experiment at the CERN LHC”. W: *JINST* 3 (2008),
 846 S08002. DOI: 10.1088/1748-0221/3/08/S08002.
- 847 [24] Betty Bezverkhny Abelev i in. “Performance of the ALICE Experiment at the CERN
 848 LHC”. W: *Int. J. Mod. Phys.* A29 (2014), s. 1430044. DOI: 10.1142/S0217751X14300440.
 849 arXiv: 1402.4476 [nucl-ex].
- 850 [25] Wikimedia Commons. *Schematics of the ALICE subdetectors*. 2014. URL: [https://commons.wikimedia.org/wiki/File:2012-Aug-02-ALICE_3D_v0_with_Text_\(1\)_2.jpg](https://commons.wikimedia.org/wiki/File:2012-Aug-02-ALICE_3D_v0_with_Text_(1)_2.jpg).
- 853 [26] Sotiris B Kotsiantis, I Zaharakis i P Pintelas. “Supervised machine learning: A review
 854 of classification techniques”. W: *Emerging artificial intelligence applications in computer*
 855 *engineering* 160 (2007), s. 3–24.
- 856 [27] Marcin Wolter. “Metody analizy wielu zmiennych w fizyce wysokich energii”. Prac. dokt.
 857 IFJ PAN, 2012.
- 858 [28] Leo Breiman. “Bagging predictors”. W: *Machine learning* 24.2 (1996), s. 123–140.
- 859 [29] Yoav Freund i Robert E Schapire. “A decision-theoretic generalization of on-line learning
 860 and an application to boosting”. W: *Journal of computer and system sciences* 55.1 (1997),
 861 s. 119–139.
- 862 [30] Tianqi Chen i Carlos Guestrin. “XGBoost: A Scalable Tree Boosting System”. W: *Proce-*
 863 *edings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and*
 864 *Data Mining*. KDD ’16. San Francisco, California, USA: ACM, 2016, s. 785–794. ISBN:
 865 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: <http://doi.acm.org/10.1145/2939672.2939785>.
- 867 [31] James Bergstra i Yoshua Bengio. “Random Search for Hyper-parameter Optimization”.
 868 W: *J. Mach. Learn. Res.* 13 (lut. 2012), s. 281–305. ISSN: 1532-4435. URL: <http://dl.acm.org/citation.cfm?id=2188385.2188395>.

- 870 [32] Sandhya Samarasinghe. *Neural networks for applied sciences and engineering: from fun-*
 871 *damentals to complex pattern recognition*. Auerbach publications, 2016.
- 872 [33] Ian Goodfellow, Yoshua Bengio i Aaron Courville. *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press, 2016.
- 874 [34] Kurt Hornik. “Approximation capabilities of multilayer feedforward networks”. W: *Neural*
 875 *Networks* 4.2 (1991), s. 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>.
- 878 [35] Alex Krizhevsky, Ilya Sutskever i Geoffrey E Hinton. “Imagenet classification with deep
 879 convolutional neural networks”. W: *Advances in neural information processing systems*.
 880 2012, s. 1097–1105.
- 881 [36] Petar Veličković. *Deep learning for complete beginners: convolutional neural networks with*
 882 *keras*. 2017. URL: <https://cambridgespark.com/content/tutorials/convolutional-neural-networks-with-keras/index.html> (term. wiz. 15.07.2018).
- 884 [37] Andrew Ng. *Convolutional Neural Networks*. 2017. URL: <https://www.coursera.org/learn/convolutional-neural-networks> (term. wiz. 15.07.2018).
- 886 [38] MathWorks. *Convolutional Neural Network*. URL: <https://www.mathworks.com/solutions/deep-learning/convolutional-neural-network.html> (term. wiz. 15.07.2018).
- 888 [39] Kendrick Tan. *Capsule Networks Explained*. 2017. URL: https://kndrck.co/posts/capsule_networks_explained/ (term. wiz. 15.07.2018).
- 890 [40] François Chollet i in. *Keras*. <https://keras.io>. 2015.
- 891 [41] Martín Abadi i in. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*.
 892 Software available from tensorflow.org. 2015. URL: <https://www.tensorflow.org/>.
- 893 [42] Timothy Dozat. *Incorporating Nesterov Momentum into Adam*. 2015. URL: {http://cs229.stanford.edu/proj2015/054_report.pdf}.
- 895 [43] Diederik P. Kingma i Jimmy Ba. “Adam: A Method for Stochastic Optimization”. W:
 896 *CoRR* abs/1412.6980 (2014). arXiv: 1412.6980. URL: <http://arxiv.org/abs/1412.6980>.
- 898 [44] Nitish Srivastava i in. “Dropout: A Simple Way to Prevent Neural Networks from Over-
 899 fitting”. W: *J. Mach. Learn. Res.* 15.1 (sty. 2014), s. 1929–1958. ISSN: 1532-4435. URL:
 900 <http://dl.acm.org/citation.cfm?id=2627435.2670313>.
- 901 [45] Torbjorn Sjostrand, Stephen Mrenna i Peter Z. Skands. “A Brief Introduction to PYTHIA
 902 8.1”. W: *Comput. Phys. Commun.* 178 (2008), s. 852–867. DOI: 10.1016/j.cpc.2008.01.036. arXiv: 0710.3820 [hep-ph].
- 904 [46] Peter Skands, Stefano Carrazza i Juan Rojo. “Tuning PYTHIA 8.1: the Monash 2013
 905 Tune”. W: *Eur. Phys. J.* C74.8 (2014), s. 3024. DOI: 10.1140/epjc/s10052-014-3024-y. arXiv: 1404.5630 [hep-ph].
- 907 [47] René Brun i in. “GEANT Detector Description and Simulation Tool”. W: (1994). DOI:
 908 10.17181/CERN.MUHF.DMJ1.
- 909 [48] Matteo Cacciari, Gavin P. Salam i Gregory Soyez. “FastJet User Manual”. W: *Eur. Phys.*
 910 *J.* C72 (2012), s. 1896. DOI: 10.1140/epjc/s10052-012-1896-2. arXiv: 1111.6097
 911 [hep-ph].
- 912 [49] D0 Collaboration. *Observation of Single Top Quark Production*. 2009. URL: https://www-d0.fnal.gov/Run2Physics/top/singletop_observation/singletop_observation_updated.html (term. wiz. 15.07.2018).

- 915 [50] Matteo Cacciari i Gavin P. Salam. “Pileup subtraction using jet areas”. W: *Phys. Lett.*
916 B659 (2008), s. 119–126. DOI: 10.1016/j.physletb.2007.09.077. arXiv: 0707.1378
917 [[hep-ph](#)].
- 918 [51] Andrew P Bradley. “The use of the area under the ROC curve in the evaluation of
919 machine learning algorithms”. W: *Pattern recognition* 30.7 (1997), s. 1145–1159.

920 Dodatek A Metryki

921 W poniżej tabeli zebrano stosowane najczęściej miary jakości klasyfikatorów. We wzorach defi-
 922 niujących metryki wykorzystano następujące wielkości:

923 TP (ang. *true positives*) – liczba poprawnie zaklasyfikowanych przypadków klasy pozytywnej

924 TN (ang. *true negatives*) – liczba poprawnie zaklasyfikowanych przypadków klasy negatywnej

925 FP (ang. *false positives*) – liczba błędnie zaklasyfikowanych przypadków klasy negatywnej

926 FN (ang. *false negatives*) – liczba błędnie zaklasyfikowanych przypadków klasy pozytywnej

927

nazwa metryki	nazwa angielska	wzór
dokładność	accuracy	$(TP+TN) / (TP+TN+FP+FN)$
precyzja	precision	$TP / (TP+FP)$
czułość, wydajność id. sygnału	recall, sensitivity, TP Rate	$TP / (TP+FN)$
swoistość	specificity, TN Rate	$TN / (TN+FP)$
F1	F1	$2 \frac{precision \cdot recall}{precision + recall}$
prawd. błędnej klas. tła	mistagging rate, FP Rate	$FP / (FP+TN)$

Tablica A1: Tabela zawierająca nazwy i definicje popularnych metryk.

928 Podanie tylko jednej metryki jest zwykle niewystarczające i stosuje się pary metryk, np.
 929 precyzja - czułość. Jeśli predykcja klasyfikatora ma charakter ciągły, to poprzez zmienianie
 930 wartości progowej otrzymuje się różne punkty pracy, scharakteryzowane przez wartości obu
 931 metryk. Kolejne punkty pracy wykreślone np. na wykresie $TPR(FPR)$ dają krzywą nazywaną
 932 krzywą ROC. Pole pod tą krzywą (ang. *ROC Area Under Curve – ROC AUC*) jest kolejną
 933 metryką, bardzo często wykorzystywana w praktyce, gdyż łączy w sobie wszystkie możliwe
 934 punkty pracy.

935 W literaturze dot. klasyfikacji dżetów wyniki zwykle przedstawia się z użyciem dwóch po-
 936 wszechnie znanych metryk, ale o zmienionych nazwach: *True Positive Rate*, nazywanej *b jet*
 937 *tagging efficiency* – wydajności na identyfikację dżetów *b* oraz *False Positive Rate*, nazywanym
 938 *mistagging rate* – prawdopodobieństwa błędnej klasyfikacji dżetów tła jako dżety *b*.

939 Dodatek B Skróty i oznaczenia

940 W pracy używane są następujące skróty i oznaczenia:

- 941 • algorytmy:

942 FC – sieci neuronowe w pełni połączone,

943 $Conv$ – sieci neuronowe konwolucyjne,

944 BDT - wzmacniane drzewa decyzyjne

- 945 • zbiory danych:

946 SV – zestaw zawierający tylko zmienne związane z wtórnymi wierzchołkami,

947 $constit$ – zestaw zawierający tylko zmienne związane z częstками tworzącymi dżet,

948 $merged$ – zestaw zawierający wszystkie zmienne (tj. $SV + Constit +$ zmienne na poziomie
949 dżetu)

- 950 • modele (zestaw danych + algorytm): nazywane są wg wzoru:

951 (oznaczenie_zbioru_danych)-(oznaczenie_algorytmu),

952 np. $SV-Conv$ oznacza konwolucyjną sieć neuronową wytrenowaną na zmiennych związanych
953 z wtórnymi wierzchołkami a $merged-BDT$ - wzmacniane drzewa decyzyjne, do
954 treningu których użyte zostały wszystkie zmienne. Zapis $merged-X$ stanowi zbiorcze ozna-
955 czenie modeli $merged-SV$, $merged-Conv$ i $merged-BDT$.

- 956 • poszczególne zmienne (kolumny w zbiorze danych) oznaczane są według wzoru:

957 (nazwa_zmiennej) – (numer_obiektu),

958 np. $\sigma_{Lxy} - SV2$ oznacza niepewność wyznaczenia L_{xy} dla wtórnego wierzchołka nr 2, a
959 $IP_Z - C5$ – parametr zderzenia wzdłuż osi wiązki częstki nr 5 (numery na posortowanych
960 listach, por. Rozdz. 3, gdzie znajdują się także opisy wielkości fizycznych)