

Spis treści

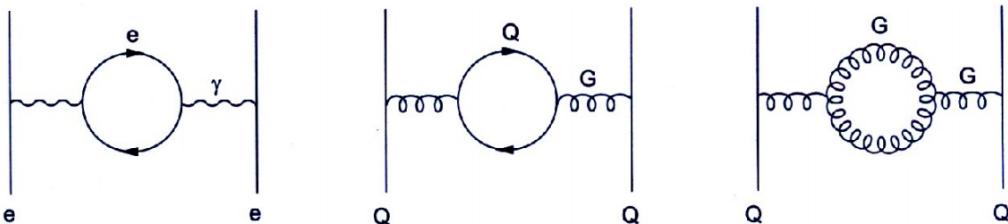
1 Fizyka dżetów cząstek	2
1.1 Chromodynamika kwantowa	2
1.2 Plazma kwarkowo-gluonowa	3
1.3 Dżety	4
1.4 Dżety b	5
1.5 Eksperyment ALICE	6
2 Uczenie maszynowe	9
2.1 Wzmacniane drzewa decyzyjne	9
2.2 Sieci neuronowe	10
2.3 Dyskusja użycia dwóch algorytmów	15
3 Dane	16
4 Analiza	19
4.1 Dobór metryki	19
4.2 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
5 Podsumowanie i wnioski	35
Dodatki	40
Dodatek A Metryki	40
Dodatek B Skróty i oznaczenia	41

23 1 Fizyka dżetów cząstek

24 1.1 Chromodynamika kwantowa

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

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



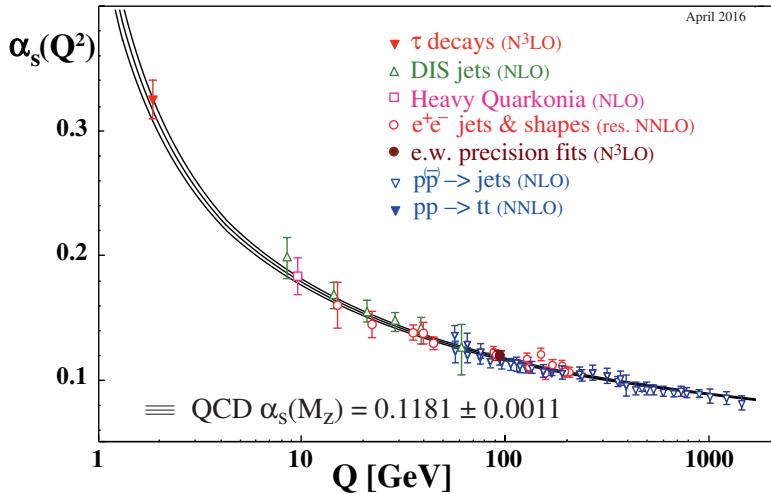
Rysunek 1: Diagramy Feynmana opisujące polaryzację próżni w *QED* (lewy) i *QCD* (środkowy i prawy). Rysunki lewy i środkowy są swoimi odpowiednikami w tych dwóch teoriach, natomiast prawy, w którym oddziałują jedynie bozony pośredniczące nie ma swojego odpowiednika w *QED*. Źródło: [1]

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

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

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

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



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

63 1.2 Plazma kwarkowo-gluonowa

64 Odkrycie asymptotycznej swobody pozwoliło na sprawdzenie przewidywań *QCD* w warun-
 65 kach bardzo wysokich temperatur oraz gęstości. Dla wystarczająco dużych gęstości hadrony
 66 zaczynają na siebie zatoczyć, prowadząc do stworzenia stanu, w którym poszczególne hadrony
 67 przestają być odróżnialne [5]. Zasugerowane zostało także istnienie przejścia fazowego w tem-
 68 peraturze porównywalnej z masą pionów oraz w temperaturze niższej, ale przy odpowiednio
 69 dużych gęstościach [6]. Stan materii powstały po osiągnięciu, któregoś z tych warunków nazy-
 70 wany jest plazmą kwarkowo-gluonową (ang. *quark-gluon plasma – QGP*). Obecnie przewiduje
 71 się, że materia w takim stanie istniała w pierwszych ułamkach sekund po Wielkim Wybuchu
 72 [7] oraz, że może się znajdować w jądrach gwiazd neutronowych [8].

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

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

85 1.3 Dżety

86 Przedstawiona powyżej definicja dżetu nie jest precyzyjna z punktu widzenia pracy ekspery-
87 mentalnej. W detektorze obserwuje się tylko cząstki w stanie końcowym, nie jest zatem możliwe
88 przyporządkowanie cząstki do dżetu według jej pochodzenia. W związku z tym, konieczne jest
89 użycie algorytmu klasteryzującego, dostającego na wejściu tylko obserwowalne eksperymentalnie
90 cząstki. To jakie dżety zostaną zaobserwowane w danym zdarzeniu zależy od użytego
91 algorytmu. Oznacza to, że precyzyjną definicję dżetu stanowi algorytm klasteryzujący wraz z
92 zestawem parametrów.

93 Obecnie najpowszechniej stosowanym algorytmem jest algorytm *anti-kt* [11]. Jest to algo-
94 rytm iteracyjny, który łączy kolejno pary cząstek na najmniejszej wartości wielkości
95 $d_{ij} = \min(1/k_{t,i}^2, 1/k_{t,j}) \frac{\Delta_{ij}^2}{R^2}$, gdzie $\Delta_{ij} = (\eta_i - \eta_j)^2 + (\phi_i - \phi_j)^2$, natomiast $k_{t,i}$, η_i , ϕ_i to kolejno
96 pęd poprzeczny, *pseudopospieszeńszość* (ang. *pseudorapidity*)¹ i kąt azymutalny i -tej cząstki. R
97 jest parametrem określającym rozmiar dżetu i zwykle przyjmuje wartości 0.2, 0.4 lub 0.7. Tak
98 określona miara, według której łączone są cząstki sprawia, że w pierwszej kolejności łączone są
99 cząstki z najbliższego otoczenia i koncentrują się one wokół do tzw. twardych cząstki (tj. tych
100 o dużym k_t).

101 Eksperymentalne ograniczenia związane z obserwacją tylko końcowego stanu oddziaływań
102 nie występują w analizie danych z symulacji Monte Carlo (MC), gdzie ma się dostęp do peł-
103 nej informacji na temat historii każdej cząstki. Nie należy jednak używać jej do klasteryzacji
104 dżetów, gdyż utracona została cecha odpowiedniości między obiektymi nazywanymi dżetami
105 w symulacji i w eksperymencie, która to cecha jest niewątpliwie jedną z podstawowych wyma-
106 gań stawianych przed dobrą symulacją. Właściwym podejściem jest rekonstrukcja dżetów przy
107 pomocy takiego samego algorytmu jak w przypadku danych eksperymentalnych.

108 Dżety wykorzystuje się w badaniach plazmy kwarkowo-gluonowej. Dają one pośredni wgląd
109 we właściwości *QGP* na podstawie wpływu jaki wywiera na oddziałujące z nią partony. Przy-
110 kładową obserwablą mierzoną w zderzeniach ciężkich jonów jest czynnik modyfikacji jądrowej
111 R_{AA} (ang. *nuclear modification factor*), który jest miarą strat energii przez parton przecho-
112 dzający przez medium. Jest to stosunek pędowych rozkładów dżetów cząstek zmierzonych w
113 zderzeniach ciężkich jąder oraz w zderzeniach pp (przemnożonych przez liczbę binarnych zde-
114 rzeń nukleon-nukleon przewidywanych przez model teoretyczny). Odchylenia od wartości 1 dla
115 wysokich p_T są oznaką modyfikacji pędów dżetów przez gęste medium (w stosunku do zderzeń
116 pp gdzie *QGP* nie powstaje), jest to tzw. tłumienie dżetów (ang. *jet quenching*). Rezultaty
117 pomiarów R_{AA} dostępne są w [12] (CSM) i [13] (ALICE).

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

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

128 1.4 Dżety b

129 1.4.1 Właściwości

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

135 Identyfikacja dżetów b jest sporym wyzwaniem ze względu na zdecydowanie częściej wy-
136 stępujące dżety lekkie, tj. powstałe z hadronizacji kwarków u,d,s lub gluonów. Rozpoznawanie
137 dżetów b bazuje na charakterystycznych właściwościach hadronów zawierających kwark piękny:
138 relatywnie długim czasie życia oraz (w mniejszym stopniu) na ich półleptonowych rozpadach o
139 względnej częstości rozpadu w tym kanale (ang. *branching ratio*) na poziomie 10% [4].

140 1.4.2 Przegląd algorytmów używanych do identyfikacji dżetów b na LHC

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

147 Najprostszym algorytmem jest dyskryminacja na podstawie istotności statystycznej (wynik
148 pomiaru podzielony przez jego niepewność) odległości wtórnego wierzchołka od wierzchołka
149 pierwotnego L . Jest to metoda wykorzystywana w każdym z trzech wymienionych ekspery-
150 mentów (por. ATLAS: algorytm SV0, CMS: algorytm SSV, ALICE). Może być ona rozsze-
151 rzona poprzez użycie dodatkowych zmiennych opisujących wtórnego wierzchołek jak na przy-
152 kład jego masy, ułamek niesionej przez niego całkowitej energii dżetu (por. ATLAS: SV1) lub
153 użycie "pseudowierzchołków" (kombinacji dwóch cząstek o dużych DCA) w celu poprawienia
154 wydajności detekcji o przypadki, w których wtórnego wierzchołek nie został zrekonstruowany
155 (por. CMS: CSV).

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

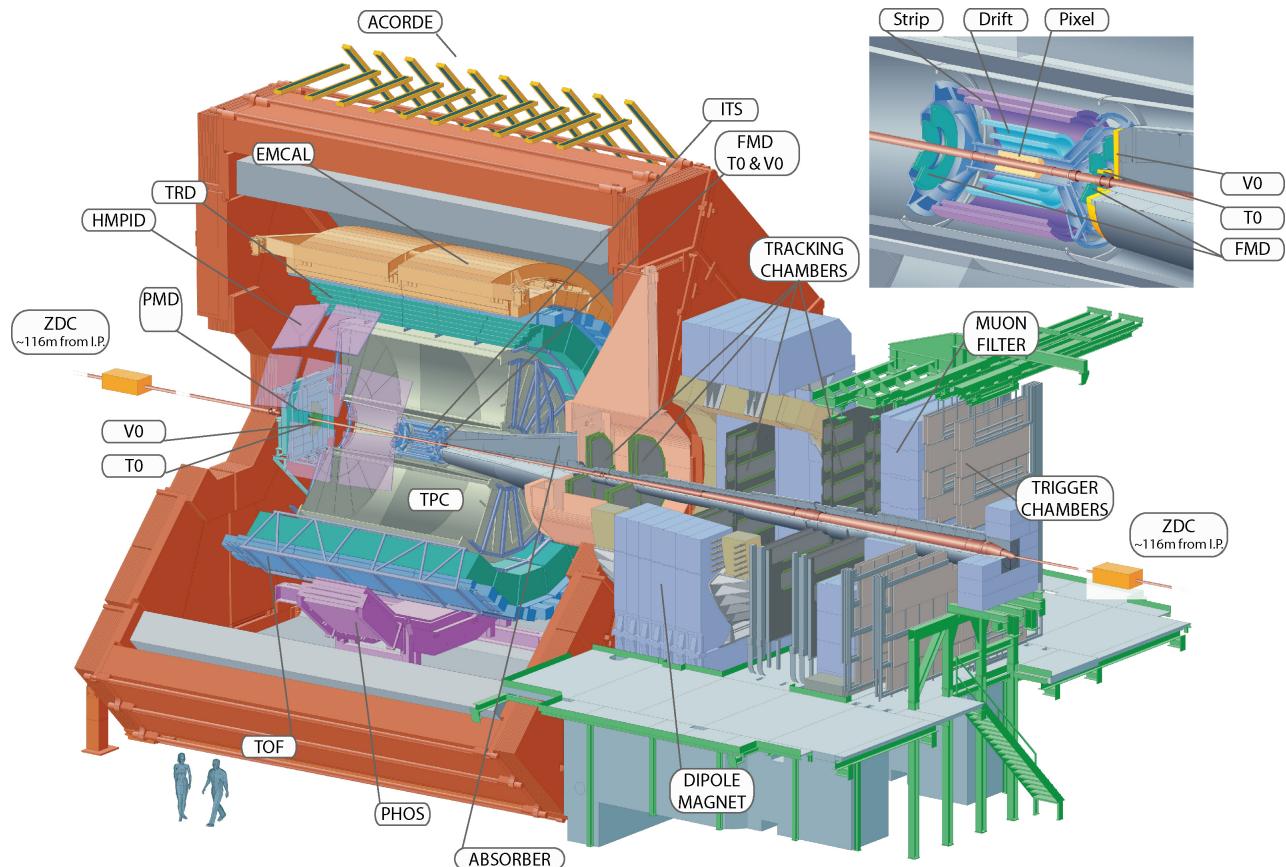
162 Do wykorzystania półleptonowego kanału rozpadu ciężkich hadronów do identyfikacji dżetów
163 b w eksperymentach ATLAS (SMT) i CMS (SE, SM) użyto wzmacnianych drzew decyzyjnych
164 trenowanych na kilku ręcznie zdefiniowanych w tym celu zmiennych takich jak pęd leptonu
165 transwersalny względem osi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

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

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

1.5 Eksperyment ALICE

Eksperyment ALICE [23], [24] jest jednym z czterech największych eksperymentów na LHC. Jest on dedykowany zderzeniom ciężkich jonów (w LHC są to jony ołowiu PbPb), ale mierzone są także mniejsze systemy, tj. proton-proton pp (głównie jako referencję dla pomiarów PbPb) oraz proton-ołów p+Pb, które dostarczają także okazji do badania asymetrycznych zderzeń. Cechą charakterystyczną pomiarów ciężkojonowych jest ich znacznie większa niż w przypadku zderzeń pp krotność, tj. liczba cząstek wyprodukowana w pojedynczym zderzeniu. W przypadku zderzeń PbPb może powstawać nawet do 8000 naładowanych cząstek na jednostkę pseudopospieszności η . Detektor ALICE został zoptymalizowany do mierzeniach takich przypadków, jak również pod kątem rekonstrukcji i identyfikacji cząstek o szerokim zakresie pędów (100 MeV/c – 100 GeV/c).



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

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

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

197 – Time Projection Chamber (TPC) – długa na 5 m i o takiej średnicy komora projekcji
198 czasowej. Jest to główny detektor śladowy ALICE, wraz z ITS służy do wyznaczania
199 trajektorii cząstek i na ich podstawie również wierzchołków zderzenia. Elektryny
200 uwolnione ze zjonizowanego przez poruszające się w nim naładowane cząstki gazu
201 dryfują wzdłuż kierunku wiązki w stronę końcowych elektrod. Następnie są tam
202 zbierane dostarczając informacji o dwóch współrzędnych toru cząstki: odległości od
203 wiązki i kącie azymutalnym. Trzecia składowa trajektorii jest otrzymywana na pod-
204 stawie czasu dotarcia elektronów do elektrod. TPC jest najwolniejszym detektorem
205 ALICE (ze względu na ograniczający czas dryfu elektronów wynoszący $\sim 90 \mu\text{s}$),
206 użycie detektora tego typu podyktowane jest jego zdolnością do mierzenia śladów
207 tysięcy cząstek spodziewanych w centralnych zderzeniach PbPb.

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

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

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

230 • kalorymetry

- 231 – Photon Spectrometer (PHOS) – elektromagnetyczny kalorymetr o wysokiej rozdziel-
232 czości energetycznej i przestrzennej (podzielony na kryształy o rozmiarze poprzecznym
233 $2.2 \times 2.2 \text{ cm}$, co odpowiada przestrzeni fazowej $\Delta\eta \times \Delta\phi = 0.004 \times 0.004$).
234 Pokrywa zakres pseudopospieszności $|\eta| < 0.12$ i kąta azymutalnego równy 100° .
235 PHOS ma za zadanie identyfikację i pomiar czteropędów fotonów, w szczególności
236 tych niepochodzących z rozpadu innych cząstek (ang. *direct photons*) oraz lekkich
237 mezonów neutralnych (np. π^0) przez dwufotonowy kanał rozpadu.
- 238 – Electromagnetic Calorimeter (EMCal) – drugi elektromagnetyczny kalorymetr ALICE
239 o mniejszej ziarnistości ($\Delta\eta \times \Delta\phi = 0.014 \times 0.014$), ale dużo większej akceptancji
240 ($|\eta| < 0.7$, $\Delta\phi = 107^\circ$). EMCal poprawia możliwości ALICE w zakresie pomiarów
241 tłumienia dżetów, pozwalając na wyznaczanie neutralnej składowej energii dżetów

(energii niesionej przez neutralne cząstki). Dzięki innej charakterystyce dla elektrownów i hadronów (elektrony typowo deponują niemal całą energię a hadrony tylko niewielką część) pozwala je odróżnić na podstawie stosunku zmierzonej w nim energii do wyznaczonego wcześniej pędu E/p . EMCAL może być użyty także w szybkim systemie wyzwalania, do selekcji przypadków z dżetami oraz wysokoenergetycznymi fotonami i elektronami.

- Muon spectrometer – spektrometr mionowy, złożony z dwóch pasywnych absorberów, znajdujących się między nimi 10 warstw detektora śladowego oraz komór systemu wyzwalającego 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 cząstками powstałymi w jego materiale. Komory śladowe mają zdolność rozdzielczą ok. $100 \mu\text{m}$, co pozwala osiągnąć wysoką rozdzielcość 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) pokrywających inną przestrzeń fazową ponieważ tory protonów są odchylane 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 gazowy mierzący krotkości oraz rozkład przestrzenny fotonów
 - FMD – krzemowy detektor paskowy mierzący precyzyjnie 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 późniowych.
 - T0 – zespół dwóch liczników Cherenkova, który dostarcza dokładny czas interakcji potrzebny dla detektora TOF, pozwala także na śledzenie świetlności w czasie rzeczywistym.

274 2 Uczenie maszynowe

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

281 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*).

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

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

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

300 2.1 Wzmacniane drzewa decyzyjne

301 Wzmacniane drzewa decyzyjne są jednym z rozwinięć klasycznego algorytmu drzewa decyzyjnego. Pojedyncze drzewo decyzyjne dzieli przestrzeń cech uczących przy pomocy prostopadłych
302 cięć, na mniejsze/większe niż zadana wartość w przypadku zmiennej ciągłej lub na należące/nie
303 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
304 były jak najbardziej jednorodne. Stosuje się różne miary nieporządku takie tak: indeks Gini G_L
305 lub entropia S_L ³.

306 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.

307 *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

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

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

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

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

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

341 Parametry optymalizowane w opisanym procesie to:

- 342 • *max_depth* – maksymalna głębokość każdego drzewa (niekoniecznie osiągana)
- 343 • *n_estimators* – liczba drzew
- 344 • *learning_rate* – parametr szybkości uczenia, komplementarny do *n_estimators*, w praktyce
345 można ustalić liczbę drzew i szukać optymalnej szybkości uczenia
- 346 • *subsample*, *colsample_bytree*, *colsample_bylevel* – parametry regularyzacyjne określające
347 ułamek kolejno: wierszy użytych do trenowania każdego drzewa, kolumn użytych w ka-
348 dym drzewie (cechy losowane raz dla danego drzewa), kolumn użytych przy każdym po-
349 dziale (cechy losowane przy każdym podziale)
- 350 • γ – minimalny zysk w postaci zmniejszenia wartości funkcji straty konieczny do wykonania
351 podziału

352 **2.2 Sieci neuronowe**

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

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

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

364 Sieci w pełni połączone

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

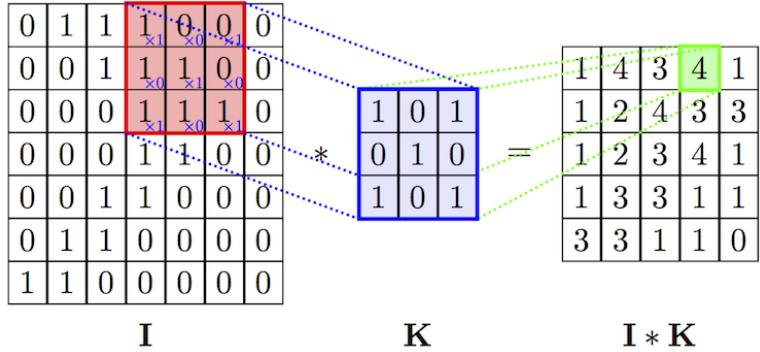
374 Istnieje twierdzenie o sieciach neuronowych jako uniwersalnych aproksymatorach funkcji
375 (ang. *universal approximation theorem*) [34], mówiące, że już sieć neuronowa o jednej warstwie
376 ukrytej jest zdolna do przybliżenia dowolnej funkcji z dowolną dokładnością. Twierdzenie to
377 nie podaje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

382 Sieci konwolucyjne

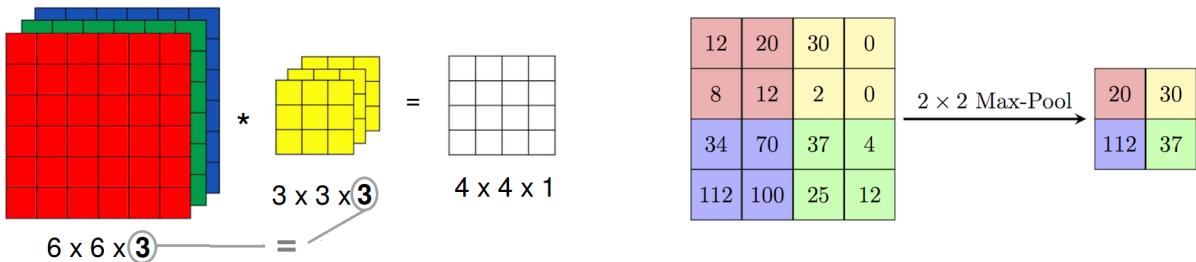
383 Jednym ze sposobów na ograniczenie liczby trenowanych parametrów jest użycie konwolucyj-
384 nych sieci neuronowych [35] (bardziej poprawną choć rzadko używaną nazwą w języku polskim
385 jest sieć splotowa). Są one inspirowane połączeniami w korze wzrokowej zwierząt i wywodzą
386 się z badań w obszarze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

389 Sieci konwolucyjne różnią się od sieci typu *FC* tym, że część wag połączeń między warstwami
390 jest dzielona. Występuje w nich nowy rodzaj warstwy, nazywany warstwą konwolucyjną. Każda
391 jednostka w warstwie konwolucyjnej (filtr) ma pewną stałą (niewielką) liczbę wag. Połączenie z
392 dużym wejściem realizowane jest przez powielanie tych samych wag w połączeniach z kolejnymi
393 fragmentami wektora wejściowego (Rys. 4). Rezultatem działania filtra na macierz jest wynik
394 operacji splotu. Liczba parametrów przypadająca na każdy filtr jest równa jego rozmiarowi i
395 nie zależy od wielkości wektora wejściowego.

396 W przypadku gdy zamiast wejścia dwuwymiarowego (jak np. obraz czarno-biały), mamy do
397 czynienia z wejściem trójwymiarowym (np. trzeci wymiar to kolejne kolory w kodowaniu RGB),
398 filtry również muszą mieć trzy wymiary, przy czym rozmiar w ostatnim wymiarze musi być
399 równy rozmiarowi w tym kierunku wektora wejściowego. Wynik operacji splotu jest ponownie
400 dwuwymiarowy, gdyż filtr przesuwany jest tylko w dwóch pierwszych wymiarach. Trzeci wymiar
401 powstaje przez składanie kolejnych filtrów. Widać zatem, że również w przypadku gdy na
402 wejścia podawany jest obraz czarno-biały, filtry w kolejnych warstwach konwolucyjnych (oprócz
403 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].

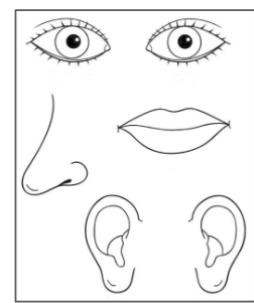
Rysunek 6: Działanie warstwy typu *max-pool*. Źródło: [36].

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

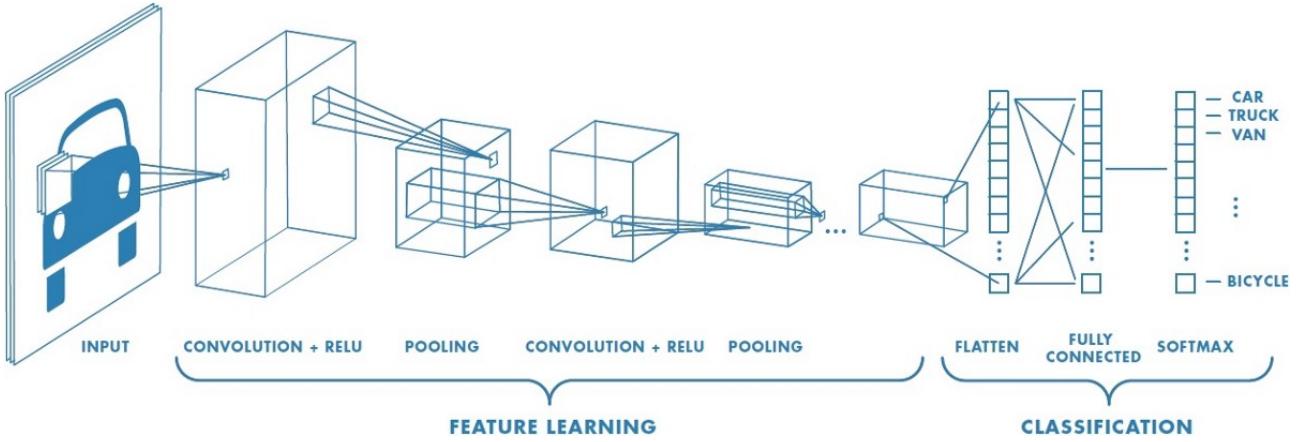
408 Typowa architektura stosowana w przypadku sieci konwolucyjnych jest następująca: naj-
409 pierw warstwy konwolucyjne (pomiędzy nimi czasem warstwy typu *max-pool*), następnie wszyst-
410 kie filtry są rozwijane i składane w długi jednowymiarowy wektor, który przekazywany jest do
411 warstw typu *FC*. W przypadku problemu klasyfikacji, na końcu znajduje się jeszcze warstwa
412 typu *softmax* normalizująca wyjście z sieci do jedynki (Rys. 8).

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

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



Rysunek 7: Źródło: [38].

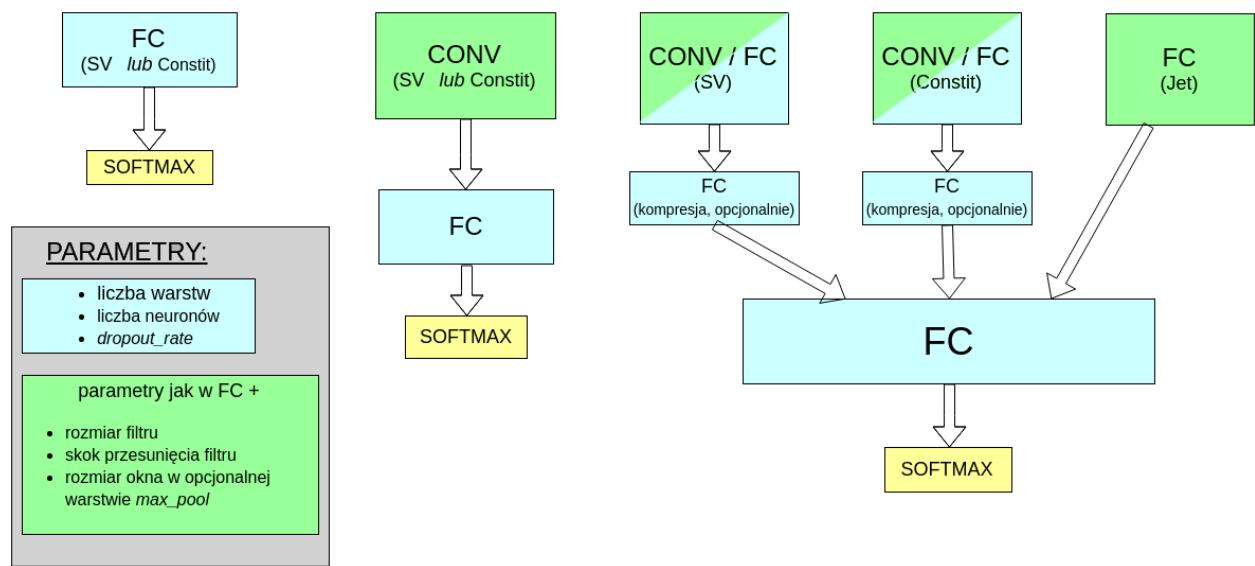


Rysunek 8: Typowa struktura stosowana w sieciach konwolucyjnych. Warstwy konwolucyjne mają za zadanie wydobywać cechy, na podstawie których późniejsze warstwy dokonują klasyfikacji. Widoczna jest charakterystyczna stopniowa zmiana rozmiaru przekazywanej macierzy: rozmiar poprzeczny maleje kosztem głębokości, co odpowiada rosnącej liczbie filtrów i malejącym rozmiarowi obszaru po jakim są one przesuwane. Źródło: [39].

424 Hiperparametry i trenowanie sieci neuronowych na analizowanych danych

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

427 Testowane były trzy architektury: sieci w pełni połączone, sieci konwolucyjne oraz sieć
428 złożona z dwóch gałęzi, osobnych dla wtórnych wierzchołków i częstek tworzących dżet (Rozdz.
429 3) przedstawione schematycznie na Rys. 9. Do trenowania sieci wykorzystano wysokopoziomową
430 bibliotekę **Keras** [40] korzystającą z silnika obliczeniowego zaimplementowanego w **TensorFlow**
431 [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.

432 Zestaw hiperparametrów definiujący działanie sieci w pełni połączonej:

- 433 • liczba warstw ukrytych

- 434 • liczba neuronów w każdej warstwie

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

- 438 • algorytm optymalizacyjny – spadek gradientowy lub jego wariacje

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

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

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

444 • opcjonalnie: warunki stopu (ang. *early stopping*), czyli przerwanie procesu uczenia mające
445 na celu uniknięcie przetrenowania sieci, w momencie gdy błąd popełniany na zbiorze
446 walidacyjnym przestaje maleć

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

448 Ponadto dla sieci konwolucyjnych:

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

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

- 451 • rozmiar filtrów i długość skoku przy ich przesuwaniu

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

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

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

- 460 • funkcję aktywacji: *ReLU*

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

- 462 • stałe w trakcie treningu wartości parametru szybkości uczenia

463 • spośród technik regularyzacyjnych testowano wyłącznie *dropout* [44] z prawdopodobień-
464 stwem odrzucenia równym 0.1, 0.2 lub 0.5

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

- 467 • liczby neuronów/filtrów w warstwach będące zawsze potęgami dwójki oraz stałą liczbę w
468 kolejnych warstwach lub zmieniającą się o stały czynnik, np. 256-128-64, 128-128-128 lub
469 16-32-64
- 470 • liczba warstw *FC*: 2-8, konwolucyjnych 2-6

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

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

480 2.3 Dyskusja użycia dwóch algorytmów

481 Użycie więcej niż jednego algorytmu ma wiele zalet. Po pierwsze daje możliwość porównania
482 wyników. Pozwala to na oszacowanie błędu *bayesowskiego* (najniższego możliwego do osiągnięcia
483 przez jakikolwiek algorytm błędu). Jest to bardzo ważne w sytuacji, gdy nie dysponuje się innym
484 oszacowaniem tego błędu (w wielu problemach naturalnych dla człowieka jak rozpoznawanie
485 obiektów na obrazkach jest nim błąd ludzki lub też błąd popełniany przez zespół ekspertów w
486 bardziej zaawansowanych zastosowaniach).

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

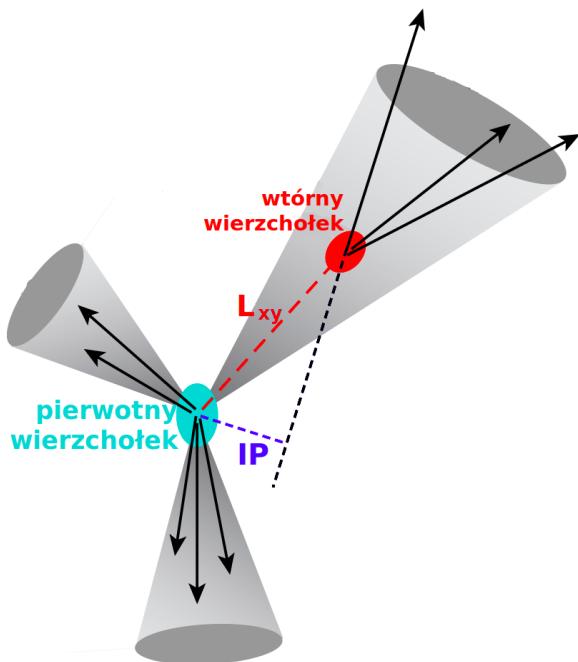
494 Z kolei drzewa decyzyjne posiadają stosunkowo niewielką liczbę parametrów, a ich trenowa-
495 nie jest bardzo szybkie co pozwala na ich ekstensywne przeszukiwanie i otrzymanie wyników,
496 które można uznać za optymalne dla tego algorytmu. Ponadto, w przypadku drzew istnieją
497 niewymagające dodatkowych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

499 3 Dane

500 Dane użyte w analizie pochodzą z symulacji Monte Carlo zderzeń proton-proton przy energii w
 501 układzie środka masy równej $\sqrt{s} = 13$ TeV dostępnych na serwerach eksperymentu ALICE. Są
 502 to pełne symulacje detektora ALICE, wykorzystujące generator zderzeń Pythia8 [45] (wersja:
 503 Monash2013 [46]) oraz pakiet Geant3 [47] do transportu cząstek przez materiał detektora.
 504 Korzystano ze specjalnych symulacji w tzw. *binach p_T*, które zapewniają lepszą statystykę dla
 505 wysokich pędów.

506 Do rekonstrukcji dżetów wykorzystany został algorytm *anti-kt* z parametrem $R = 0.4$ zaim-
 507 plementowany w pakiecie FASTJET [48]. Dżetów poszukiwano wyłącznie wśród cząstek nałado-
 508 wanych (ang. *charged jets*) ze względu na słabe pokrycie przestrzeni fazowej przez kalorymetry
 509 w eksperymencie ALICE.

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



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

513 Dla każdego dżetu obliczony został szereg wielkości, które można podzielić na zmienne cha-
 514 rakteryzujące dżet oraz opisujące wtórne wierzchołki lub cząstki tworzące dżet. Za potencjalne
 515 wtórne wierzchołki uznaje się wszystkie kombinacje trzech cząstek spełniających pewne dosyć
 516 luźne kryteria jak $p_T > 0.15$ GeV (rozważane są wyłącznie trzy-cząstkowe wtórne wierzchołki),
 517 stąd ich liczba może być dużo większa od liczby cząstek tworzących dżet.

518 Lista używanych zmiennych:

- 519 • Zmienne charakteryzujące dżet:
 - 520 – η_{jet}, ϕ_{jet} – pseudopospieszność i kąt azymutalny osi dżetu
 - 521 – $p_{T,jet}$ – pęd poprzeczny dżetu

- 522 – $M_{jet} = \sqrt{(\sum_i E_i)^2 - (\sum_i \vec{p}_i)^2}$ – masa dżetu, gdzie E_i i \vec{p}_i to energia i pęd kolejnych
523 częstek tworzących dżet
- 524 – A_{jet} – powierzchnia dżetu wyliczana przez algorytm *anti-kt* w płaszczyźnie (η, ϕ) .
525 Do powierzchni dżetu zaliczany jest każdy element powierzchni, w którym dodanie
526 częstki o nieskończenie małym pędzie poprzecznym sprawi, że zostanie ona zaliczona
527 do tego dżetu [50]
- 528 – ρ_{bckg} – gęstość tła w danym zdarzeniu

529 • Zmienne opisujące cząstki tworzące dżet (składniki dżetu):

- 530 – η, ϕ – pseudopospieszność i kąt azymutalny cząstki względem osi dżetu
- 531 – p_T – pęd poprzeczny cząstki
- 532 – IP_D – rzut na kierunek poprzeczny wektora parametru zderzenia
- 533 – IP_Z – rzut na oś z wektora parametru zderzenia
- 534 – $N_{Constit}$ – liczba cząstek tworzących dżet

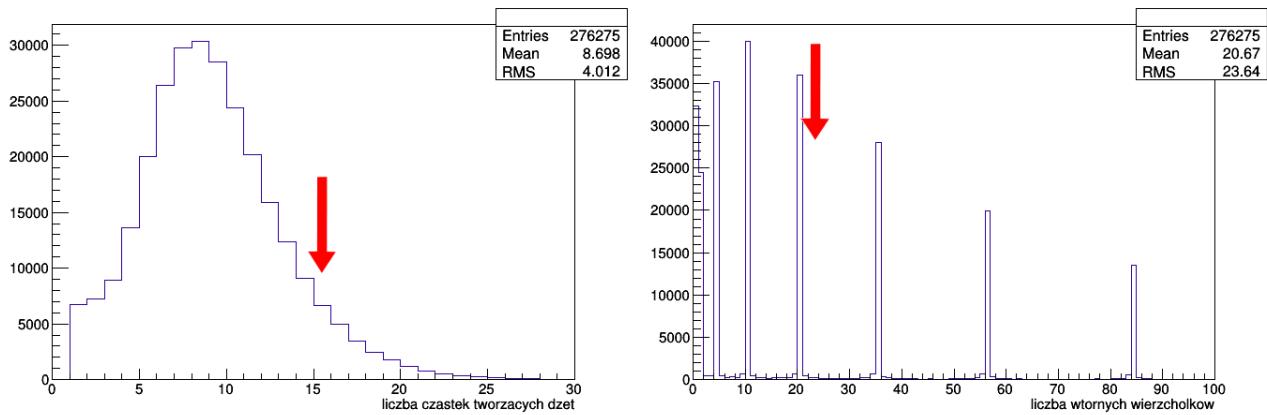
535 • Zmienne opisujące wtórne wierzchołki:

- 536 – L_{xy} – odległość między pierwotnym a wtórnym wierzchołkiem (ang. *decay length*)
- 537 – $\sigma_{L_{xy}}$ – niepewność wyznaczenia L_{xy}
- 538 – $\sigma_{vertex} = \sqrt{d_1^2 + d_2^2 + d_3^2}$ – rozrzut śladów (ang. *tracks*) wokół wtórnego wierzchołka,
539 gdzie d_i to to najmniejsza odległość pomiędzy śladem i -tej cząstki a wtórnym wierz-
540 chołkiem
- 541 – $M_{inv} = \sqrt{(E_1 + E_2 + E_3)^2 - (\vec{p}_1 + \vec{p}_2 + \vec{p}_3)^2}$ – masa niezmiennicza wierzchołka, gdzie
542 E_i, \vec{p}_i to energia i pęd i -tej cząstki tworzącej wierzchołek
- 543 – χ^2/Ndf – jakość dopasowania wtórnego wierzchołka
- 544 – N_{SV} – liczba wtórych wierzchołków

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

554 Istotnym zagadnieniem jest także kolejność w jakiej ułożone będą zmienne. Dla sieci konwo-
555 lucyjnych szukających lokalnych zależności dobrze jest pogrupować zmienne, tak aby obok siebie
556 znajdowały się te same wielkości fizyczne, np. $L_{xy,1}, L_{xy,2}, L_{xy,3} \dots \sigma_{vertex,1}, \sigma_{vertex,2}, \sigma_{vertex,3} \dots$
557 Dla sieci w pełni połączonych oraz drzew decyzyjnych kolejność zmiennych nie ma znaczenia,
558 ale ważne jest aby położenie zmiennych było ustalone dla kolejnych wierzchołków lub cząstek
559 tzn. żeby L_{xy} i $\sigma_{L_{xy}}$ danego wtórnego wierzchołka były w tych samych miejscach, tak aby
560 możliwe było szukanie zależności między nimi.

561 Nastecną kwestią jest wybór wielkości decydującej o kolejności ułożenia elementów, tj. która
562 częstka będzie cząstką nr 1 a która nr 5. Losowe ułożenie elementów sprawioby, że bezpośrednie



Rysunek 11: Rozkłady liczby cząstek tworzących dżet i liczby wtórnych wierzchołków (pokazana jest tylko część, rozkład sięga $N_{SV} = 200$) wraz wartościami cięć.

564 porównywanie wielkości w danych kolumnach (co ma miejsce bezpośrednio w drzewach decy-
 565 zyjnych a pośrednio w sieciach neuronowych) straciłoby sens. Z kolei dobry dobór tej kolejności
 566 pozwala na łatwe odtworzenie przez algorytm uczenia maszynowego motywów fizycznie
 567 algorytmów omówionych w sekcji 1.4.2. Przykładowo cięcie na wartość IP drugiej lub trzeciej
 568 cząstki (gdy są one posortowane wg malejących wartości IP) jest istotą algorytmu nazywanego
 569 *Track Counting – TC* stosowanego w CMS i ALICE. Kolejność w jakiej ułożone będą elementy,
 570 wpływa także na to, które z nich będą odrzucone w przypadku gdy dżet zawiera więcej niż 15
 571 cząstek i 20 wierzchołków. Ponownie posiliłowano się testami z użyciem drzew decyzyjnych.
 572 Ostatecznie wtórne wierzchołki ułożono według malejącego L_{xy} a cząstki – malejącego p_T .

573 4 Analiza

574 4.1 Dobór metryki

575 Bardzo ważnym elementem w trenowaniu algorytmów uczenia maszynowego jest dobór odpowiedniej metryki. Kilka najczęściej używanych metryk wymieniono w Tab. A1. Klasycznym 576 złym przykładem jest używanie dokładności (ang. *accuracy*) do oceniania klasyfikacji binarnej 577 w przypadku dużego niezrównoważenia klas – algorytm przewidujący zawsze klasę większą 578 składową może osiągnąć dużą wartość dokładności będąc jednocześnie bardzo słabym modelem.

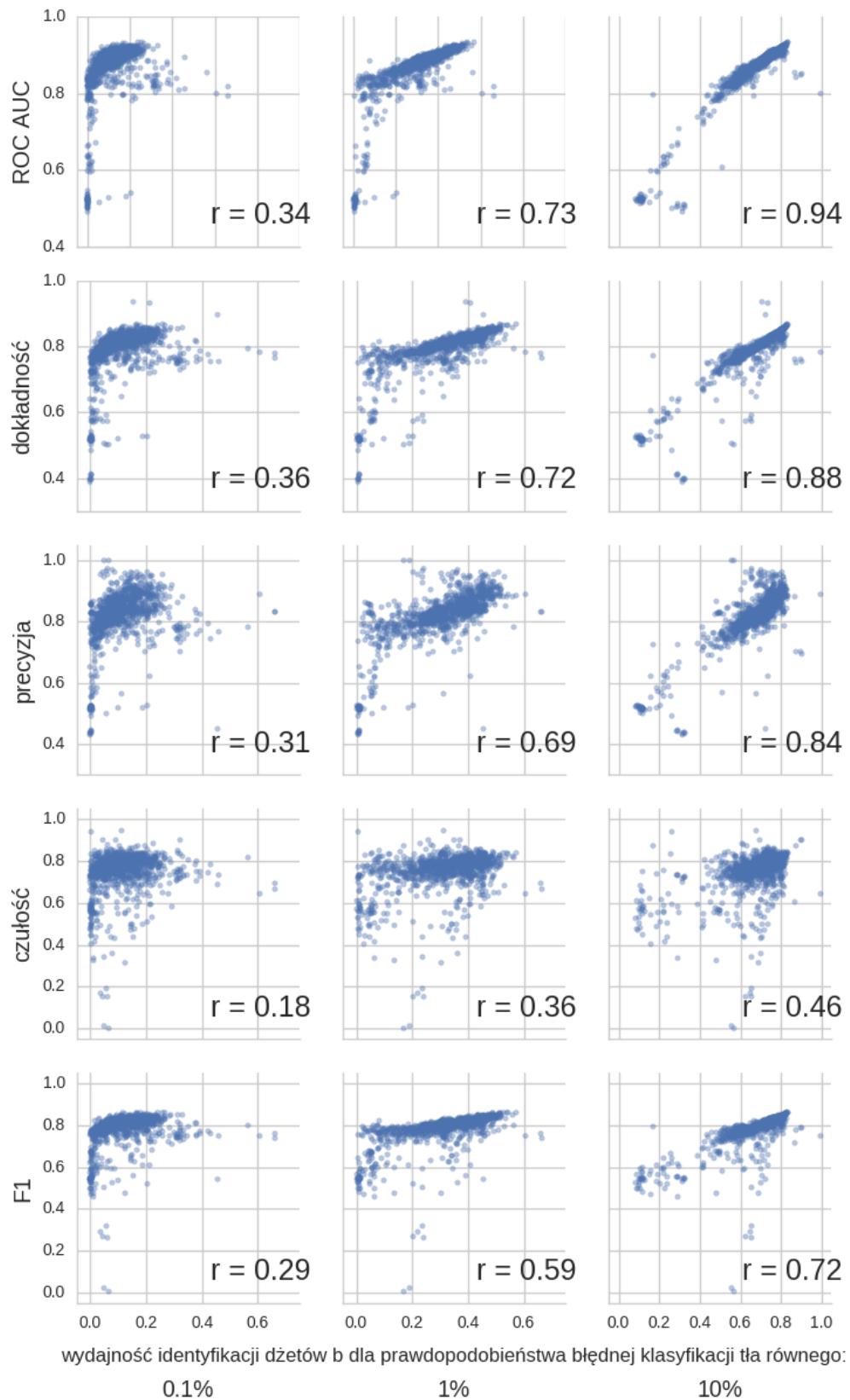
580 Używanie i porównywanie kilku miar efektywności jest często niepraktyczne dlatego dobrze 581 jest wybrać jedną metrykę. Przy jej wyborze należy kierować się potencjalnymi zastosowaniami 582 modelu. W tym przypadku są to analizy fizyczne, które mogą mieć różne wymagania dotyczące 583 czystości i liczebności otrzymywanych próbek, a co za tym idzie, preferować inne punkty 584 pracy zdefiniowane jako pary liczb: wydajność poprawnej klasyfikacji dżetów b (ang. *tagging 585 efficiency = true positive rate = recall*), i ułamek niepoprawnie zaklasyfikowanych przypadków 586 tła (ang. *mistagging rate = false positive rate*).

587 Naturalnym wyborem wydaje się pole pod powierzchnią krzywej *ROC* (ang. *ROC Area Under Curve – ROC AUC*) [51]. Potencjalną przeszkodą może być zakres rozsądnego wartości 588 prawdopodobieństwa błędnej klasyfikacji przypadków tła: dżety b stanowią tylko kilka procent 589 liczby wszystkich dżetów, zatem z punktu widzenia analizy dopuszczalne będą punkty pracy 590 zapewniające wydajność identyfikacji dżetów b ok. 10 – 100 razy większą niż częstość niepoprawnego 591 zaklasyfikowania przypadków tła. Oznacza to, że zdecydowana większość punktów 592 pracy znajdujących się na krzywej *ROC* jest nie do zaakceptowania – interesujące są tylko te 593 o najniższych częstościach błędnej klasyfikacji.

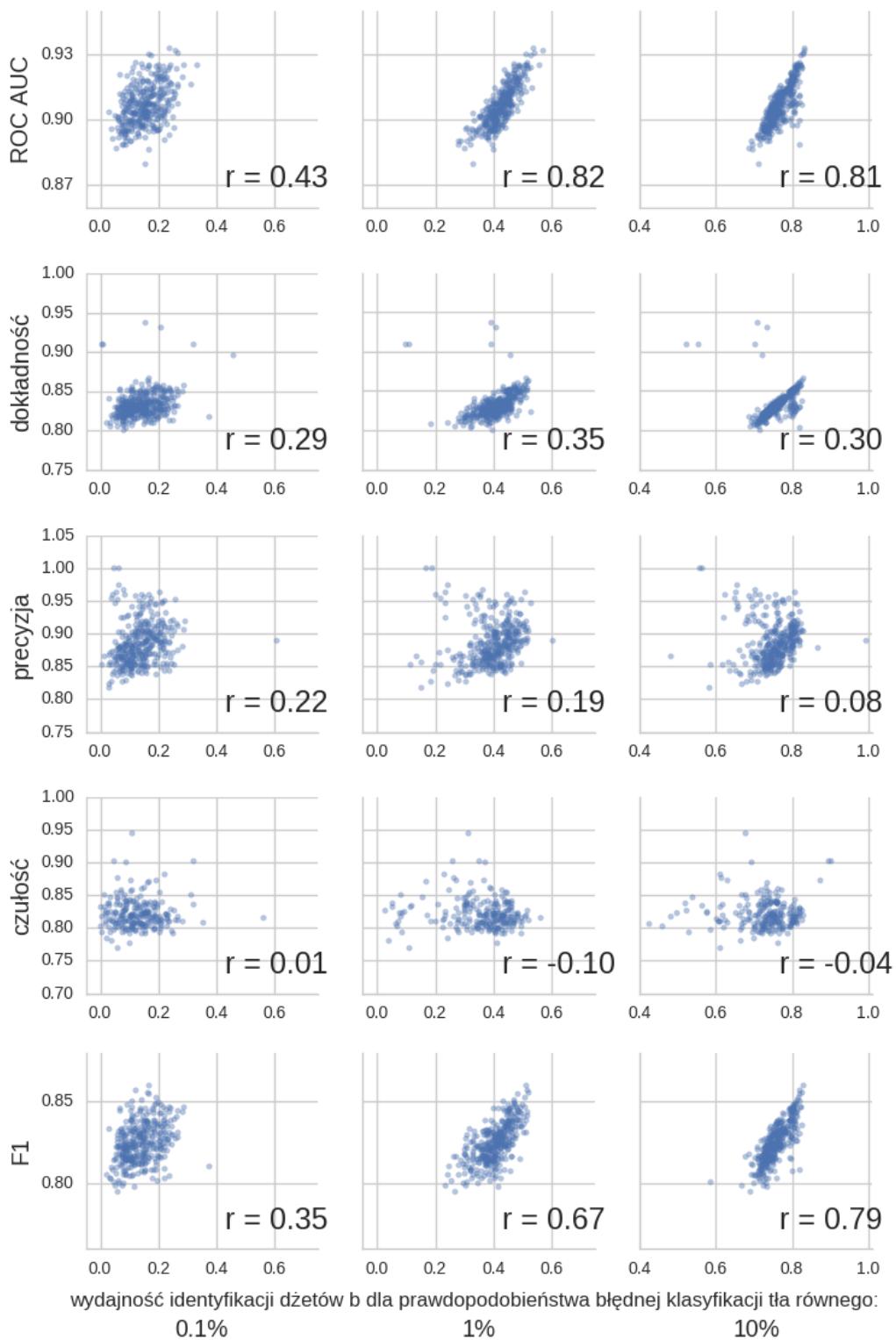
594 Aby ilościowo porównywać różne algorytmy wprowadzone zostaną trzy punkty pracy: o 595 prawdopodobieństwie błędnej klasyfikacji tła równej 0.1%, 1% oraz 10%. Na Rys. 12 przed- 596 stawione zostały zależności poszczególnych metryk od wydajności identyfikacji dżetów b w 597 tych trzech punktach pracy. Każdy punkt odpowiada jednemu eksperymentowi (dla dowolnego 598 algorytmu) przeprowadzonemu w trakcie przygotowywania analizy. Daje to pogląd, na to war- 599 tościami której metryki należy się kierować przy wyborze algorytmu i hiperparametrów, aby 600 zapewnić sobie jednocześnie wysokie wartości wydajności na identyfikację dżetów b w wybra- 601 nych punktach pracy. Najwyższe koreacje występują dla punktu pracy o najwyższym prawdo- 602 podobieństwie błędnej klasyfikacji tła – jak będzie to pokazane później wyniki dla tego punktu 603 pracy są najbardziej stabilne. Spośród analizowanych metryk najwyższe wartości współczyn- 604nika Pearsona otrzymano dla pola pod powierzchnią krzywej *ROC*, dosyć wysokie również dla 605 dokładności i precyzji. Widać, że wartości czułości są najsłabiej skorelowane z wydajnością iden- 606 tyfikacji dżetów b . Jest rzeczą zrozumiałą, że korelacja jest słabsza niż dla precyzji, ponieważ 607 wartości czułości maleją za każdym razem gdy błędnie klasyfikowane są dżety b stanowiące 608 sygnał, podczas gdy wartości precyzji maleją, gdy błędnie klasyfikowane są przypadki tła. Z 609 tych dwóch błędów, drugi jest bardziej kosztowny, gdyż błędna klasyfikacja nawet niewielkiej 610 części tła znacznie pogarsza czystość otrzymywanej próbki.

611 Na Rys. 13 przedstawiono te same wielkości co na Rys. 12, ale tym razem wybrano tylko 612 25% najwyższych (najlepszych) wartości dla każdej metryki – te punkty są bardziej znaczące, 613 gdyż ostatecznie modele z eksperymentów dających najlepsze wyniki będą używane. Dla takiej 614 selekcji otrzymano zdecydowaną dominację *ROC AUC* – wybór modeli dających najwyższe 615 pole pod powierzchnią krzywej *ROC* zapewnia jednocześnie otrzymanie wysokich wartości wy- 616 dajności identyfikacji dżetów b dla wybranych punktów pracy.

617 Pole pod powierzchnią krzywej *ROC* zostało wybrane jako główna metryka używana w 618 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.

620 4.2 Wyniki dla poszczególnych modeli

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

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

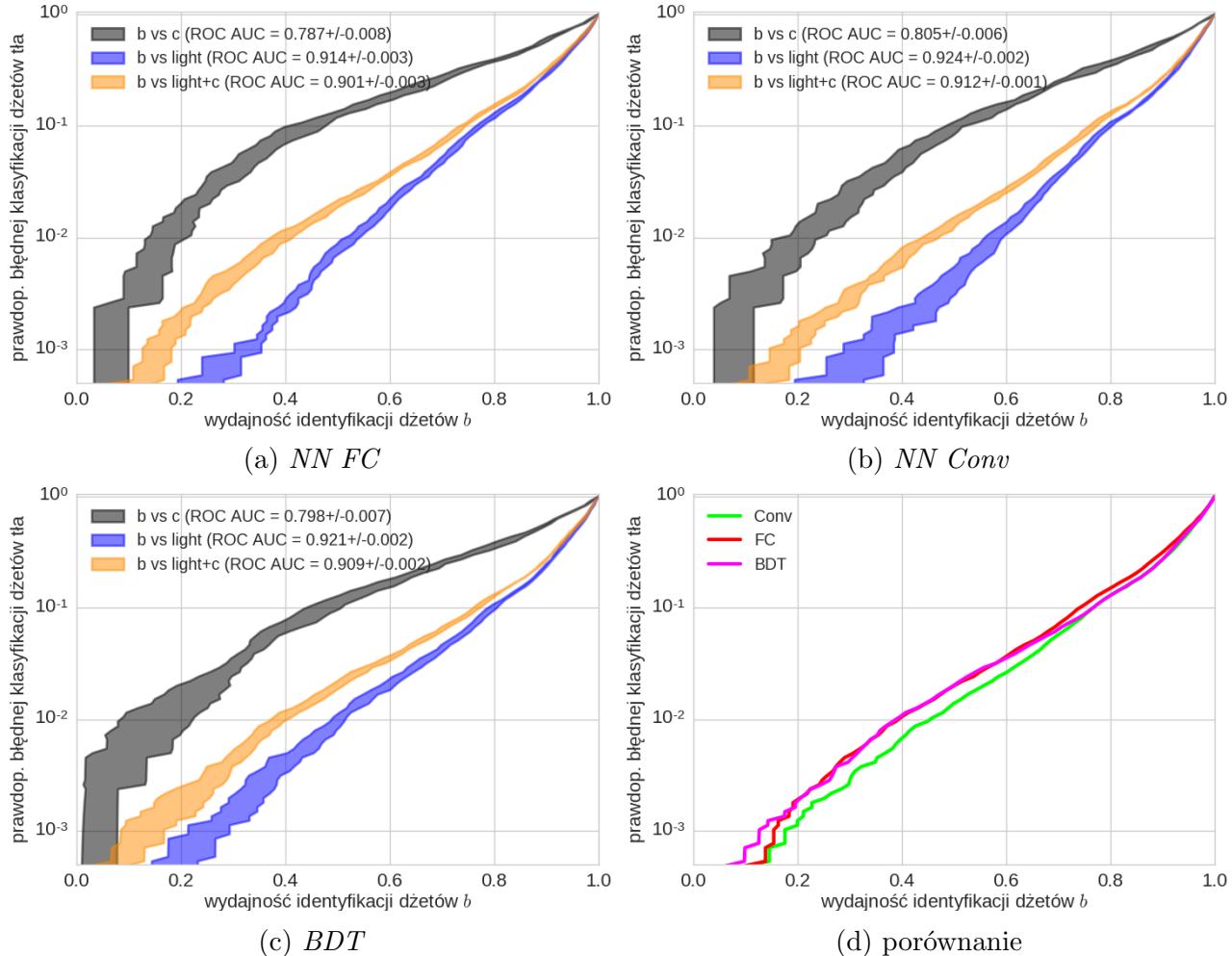
629 Na Rys. 14, 15, 16 przedstawiono odmianę krzywej *ROC* – wykresy przedstawiające zależ-
630 ności wydajności identyfikacji dżetów b z niepewnościami (na osi poziomej) dla danego praw-
631 dopodobieństwa błędnej klasyfikacji dżetów tła (na osi pionowej).

632 Trzy krzywe na każdym wykresie odpowiadają separacji dżetów b od dżetów: lekkich, c
633 oraz mieszanej próbki złożonej w 90% z dżetów lekkich i w 10% z dżetów powabnych. Cztery
634 wykresy na każdym rysunku odpowiadają trzem użytym algorytmom: sieciom neuronowym
635 typu *FC* (na górze po lewej), konwolucyjnym sieciom neuronowych (na górze po prawej) i
636 wzmacnianym drzewom decyzyjnym (na dole po lewej) oraz porównaniu wszystkich trzech
637 (na dole po prawej). Porównane zostały tylko krzywe odpowiadające separacji dżetów b od
638 mieszanego tła (bez niepewności).

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

641 4.2.1 Wyniki dla zmiennych związanych z wtórnymi wierzchołkami

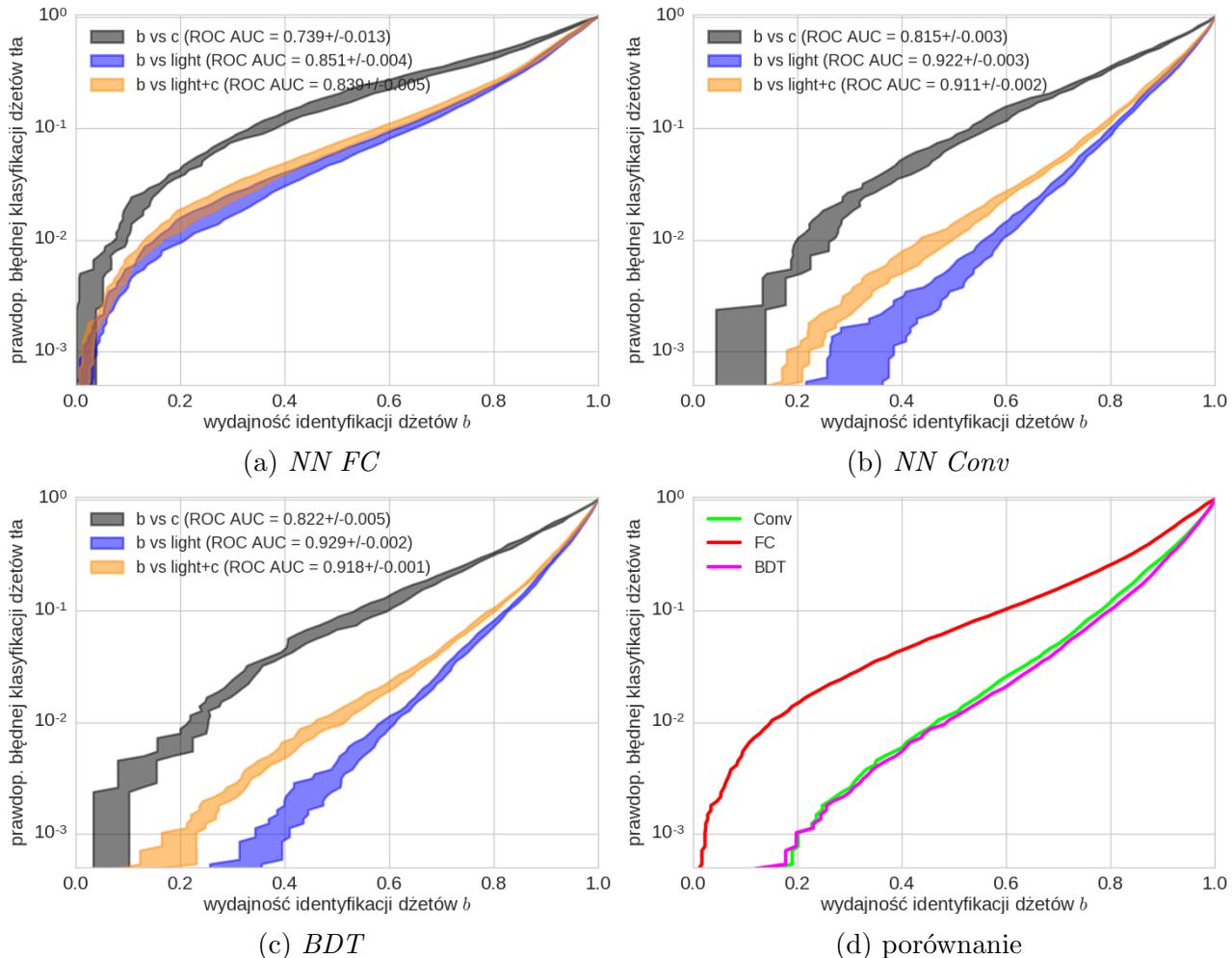
642 Na Rys. 14 przedstawiono rezultaty uzyskane przy trenowaniu na zbiorze danych *SV*. Uzyskano
 643 bardzo zbliżone wyniki dla wszystkich trzech algorytmów, sieci konwolucyjne były nieznacznie
 644 lepsze od pozostałych przy wydajnościach na identyfikację dżetów b poniżej 70%. Dla tych
 645 samych prawdopodobieństw błędnej klasyfikacji tła uzyskiwały wydajności identyfikacji lepsze
 646 o ok. 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*.

647 4.2.2 Wyniki dla zmiennych związanych z częstками tworzącymi dżet

648 Na Rys. 15 przedstawiono rezultaty uzyskane przy trenowaniu na zbiorze danych *constit*. Wyniki
 649 dla *BDT* oraz *Conv* są niemal identyczne, dla *BDT* trochę lepsze niż w przypadku zestawu da-
 650nych *SV*, natomiast te uzyskane dla sieci neuronowych typu *FC* są znacznie gorsze. Stosunkowo
 651 najmniejsze różnice pomiędzy zestawami danych *SV* i *constit* dla algorytmu *FC* otrzymano w
 652 przypadku separacji dżetów *b* od *c*.

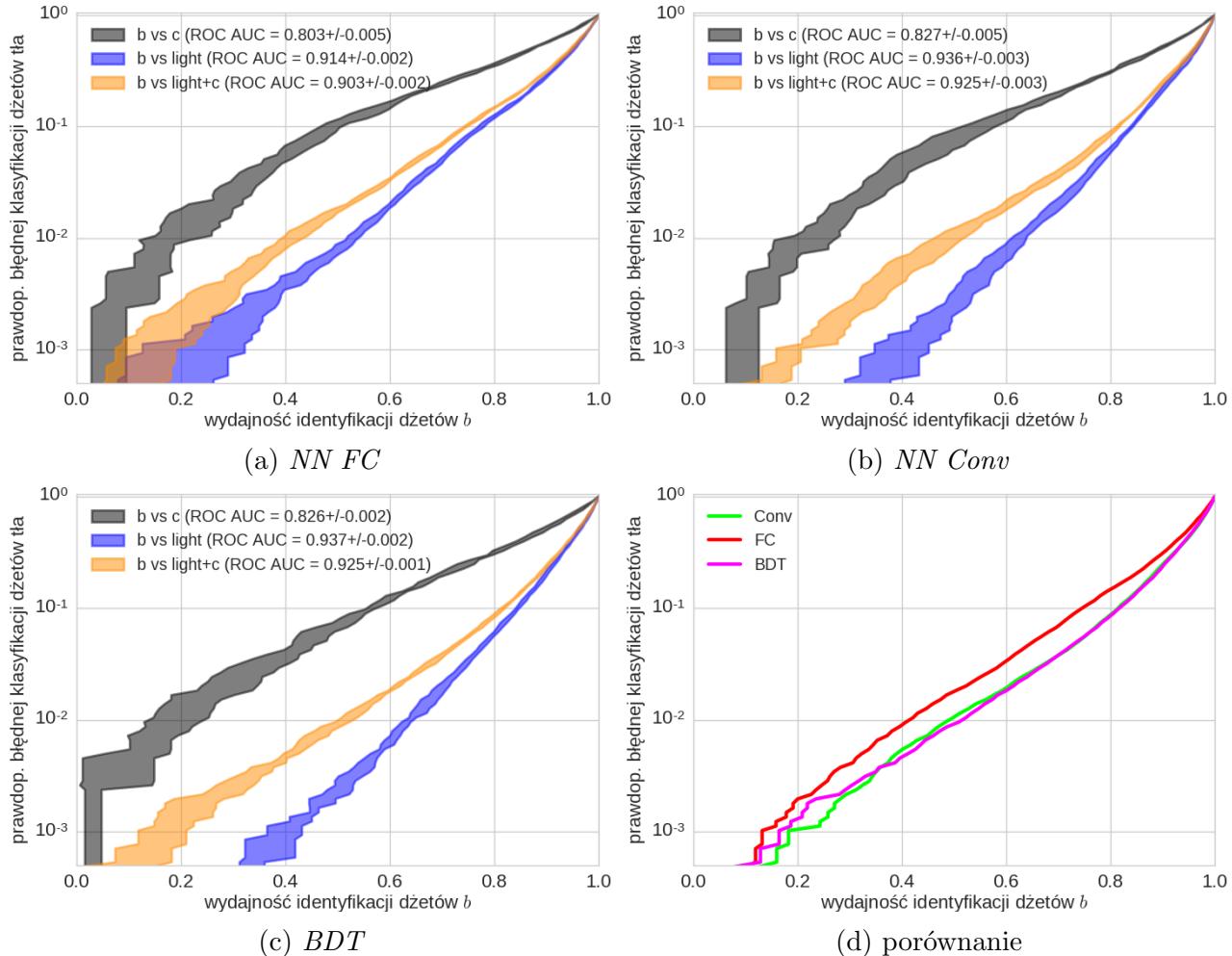


Rysunek 15: Zależności wydajności identyfikacji dżetów *b* od prawdopodobieństwa błędnej klasyfikacji dżetów tła dla poszczególnych algorytmów oraz ich porównanie. Algorytmy wytreno-
wane na zmiennych *constit*.

653 4.2.3 Wyniki dla wszystkich zmiennych

654 Na Rys. 16 przedstawiono rezultaty uzyskane przy trenowaniu na zbiorze danych *merged*. Po-
 655 nownie krzywe *ROC* wzmacnianych drzew decyzyjnych i sieci konwolucyjnych prawie się nie
 656 różnią, natomiast dla sieci w pełni połączonych krzywa jest przesunięta o ok. 10% w stronę
 657 niższych wydajności.

658 W porównaniu do poprzednich zbiorów danych wszystkie algorytmy poprawiły swoje pre-
 659 dykce, najmniej *FC*, którego zdolności separacyjne są prawie takie same jak w przypadku *SV*
 660 (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*.

661 4.2.4 Podsumowanie - wyniki poszczególnych modeli

662 Uzyskane wartości $ROC AUC$ oraz wydajności identyfikacji dżetów b dla trzech punktów pracy
 663 zebrane w Tab. 1.

664 Tak jak było to widać na Rys. 13, wartości $ROC AUC$ są najbardziej czułe na wartości
 665 wydajności osiągane dla punktu pracy o najwyższych wydajnościach. Wynika to z faktu, że
 666 pierwsze dwa punkty pracy leżą na samym początku krzywej ROC (dają niewielki wkład do
 667 pola pod krzywą).

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

674 W każdym przypadku najgorsze wyniki uzyskano dla sieci w pełni połączonych. Z tabeli
 675 wynika, że duże znaczenie mają zarówno użyty algorytm jak i zestaw danych. Podobne wyniki
 676 uzyskiwane dla BDT oraz $Conv$ pozwalają przypuszczać, że uzyskiwane przez nie poziom błędu
 677 jest zbliżony do minimum osiągalnego przy tych zestawach danych (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.

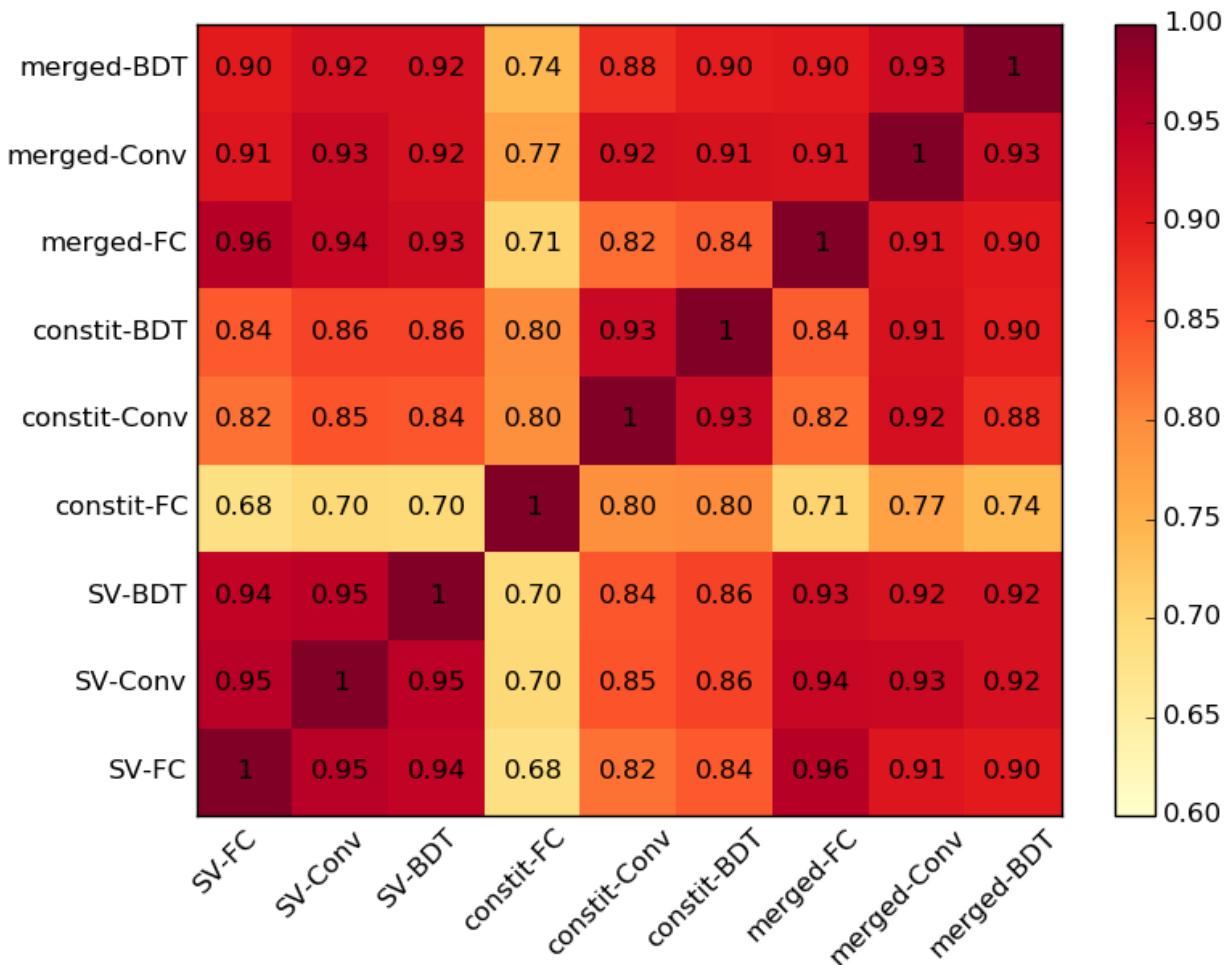
678 4.3 Korelacje predykcji modeli

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

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

685 Najmniejszymi korelacjami z innymi wyróżnia się model *constit-FC*, który dawał zdecydowanie najsłabsze wyniki.
 686

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



Rysunek 17: Średnie współczynniki korelacji Pearsona pomiędzy predykcjami poszczególnych modeli. Odchylenia standardowe nie zostały pokazane na wykresie, ich wartości wynoszą 0.001 – 0.009.

692 4.4 Analiza istotności zmiennych w *BDT*

693 Jedną z zalet drzew decyzyjnych jest możliwość dokładnego prześledzenia procesu decyzyjnego
694 poprzedzającego klasyfikację danego przypadku, co pozwala na lepsze zrozumienie działania
695 modelu. Nie jest to możliwe dla klasyfikatora złożonego z dużej liczby drzew jak *BDT*, jednak
696 nadal istnieją dosyć proste sposoby na zrozumienie, które cechy miały największy wpływ na
697 predykcję.

698 Wykorzystano trzy miary istotności zmiennych zaimplementowane w bibliotece **XGBoost**:⁴

- 699 • *weight* informuje ile razy dana zmienna została wykorzystana w podziałach drzew.
- 700 • *total_gain* jest sumą po spadkach funkcji straty uzyskiwanych dzięki podziałom wykorzy-
701 stującym daną zmienną
- 702 • *total_cover* to suma liczb przykładów trenujących, na które miały wpływ podziały wyko-
703 rzystujące daną zmienną.

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

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

718 Zdecydowanie najważniejszymi zmiennymi okazały się: L_{xy} w przypadku wtórnych wierz-
719 chołków oraz IP_D w przypadku częstek składowych dżetu, czyli zmienne, które zależą od czasu
720 życia hadronów. Wyraźna jest także tendencja, że im wyższa pozycja elementu na liście, tym
721 istotniejsze są jego właściwości (najistotniejsze są elementy nr 0).

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

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

729 Wyniki uzyskane dla modelu *merged-BDT* (prawa strona Tab. 3) pokazują, że przy treно-
730 waniu na pełnym zestawie danych zmienne ze zbioru *SV* są bardziej znaczące niż te ze zbioru
731 *constit*, co jest zgodne z odnotowaną silniejszą korelacją modeli *merged-X* z modelami *SV-X*
732 niż 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>	osobno <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*.

733 4.5 Analiza wpływu zmiennych na predykcje modeli

734 Dalszą analizę wpływu poszczególnych zmiennych przeprowadzono przy użyciu wykresów zależ-
735 ności cząstkowych (ang. *partial dependence plots*). Ze względu na wysoki koszt obliczeniowy tej
736 metody, przedstawione wyniki wyjątkowo nie są uśrednieniem kilka powtórzeń całej procedury.
737

738 Metoda wykresów zależności cząstkowych to metoda polegająca na sprawdzeniu zachowania
739 się predykcji już wytrenowanego modelu na skutek zmiany wartości wybranej zmiennej. W tym
740 celu przeprowadzane są predykcje na normalnym zbiorze danych, ze zmienioną jedną kolumną,
741 zawierającą wartości analizowanej zmiennej. Wszystkie wartości w tej kolumnie ustawiane są na
742 tę samą wartość, która jest stopniowo zmieniana od minimum do maksimum wartości przyjmowa-
743nych przez daną zmienną, za każdym razem przeprowadzana jest predykcja. Współrzędnymi
744 punktów na wykresie zależności cząstkowej są: kolejne wartości ustawiane w analizowanej ko-
745 lumnie (oś pozioma) i średnia wartość predykcji modelu (oś pionowa). Im wyższe wartości tej
746 drugiej tym większa część przykładów zostały zaliczona przez algorytm do klasy pozytywnej
(uznana za dżet b).

747 Taka analiza służy kilku celom. Po pierwsze, pozwala na weryfikację, czy modele działają
748 zgodnie z naszą wiedzą, w przypadkach gdy wiadomo jak powinien zachować się algorytm. Po
749 drugie, w przypadku braku wiedzy na temat pożdanego zachowania pozwala na wyciągnięcie
750 wniosków na podstawie sposobu działania algorytmu co wzbogaca intuicję i zrozumienie anali-
751 zowanych danych. Po trzecie, jest to jeden ze sposobów na zrozumienie działania algorytmów,
752 innych niż drzewa decyzyjne czy regresja liniowa, w których nie istnieją proste metryki pozwala-
753 jące oszacować wpływ poszczególnych zmiennych na predykcję. W przypadku sieci neuronowych
754 często jest to dużym wyzwaniem.

755 Na Rys. 18 i 19 przedstawiono wykresy zależności cząstkowych dla wybranych zmiennych.
756 Przy wyborze zmiennych posiłkowano się wartościami miar istotności cech z sekcji 4.4, kiero-
757 wano się także wartością ilustracyjną dla omawianych zagadnień. Skala osi pionowej jest inna
758 dla każdej zmiennej, im większy zakres, tym większy bezpośredni wpływ tej zmiennej na predyk-
759 cje, tym istotniejsza jest ta zmienna. Poniżej każdego wykresu przedstawiono znormalizowane
760 rozkłady prawdopodobieństwa wartości analizowanej zmiennej dla dżetów b oraz dżetów tła.
761 Warto zaznaczyć na wstępie, że najbardziej istotne są obszary wykresów w okolicach maksimów
762 rozkładów prawdopodobieństwa zmiennych.

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

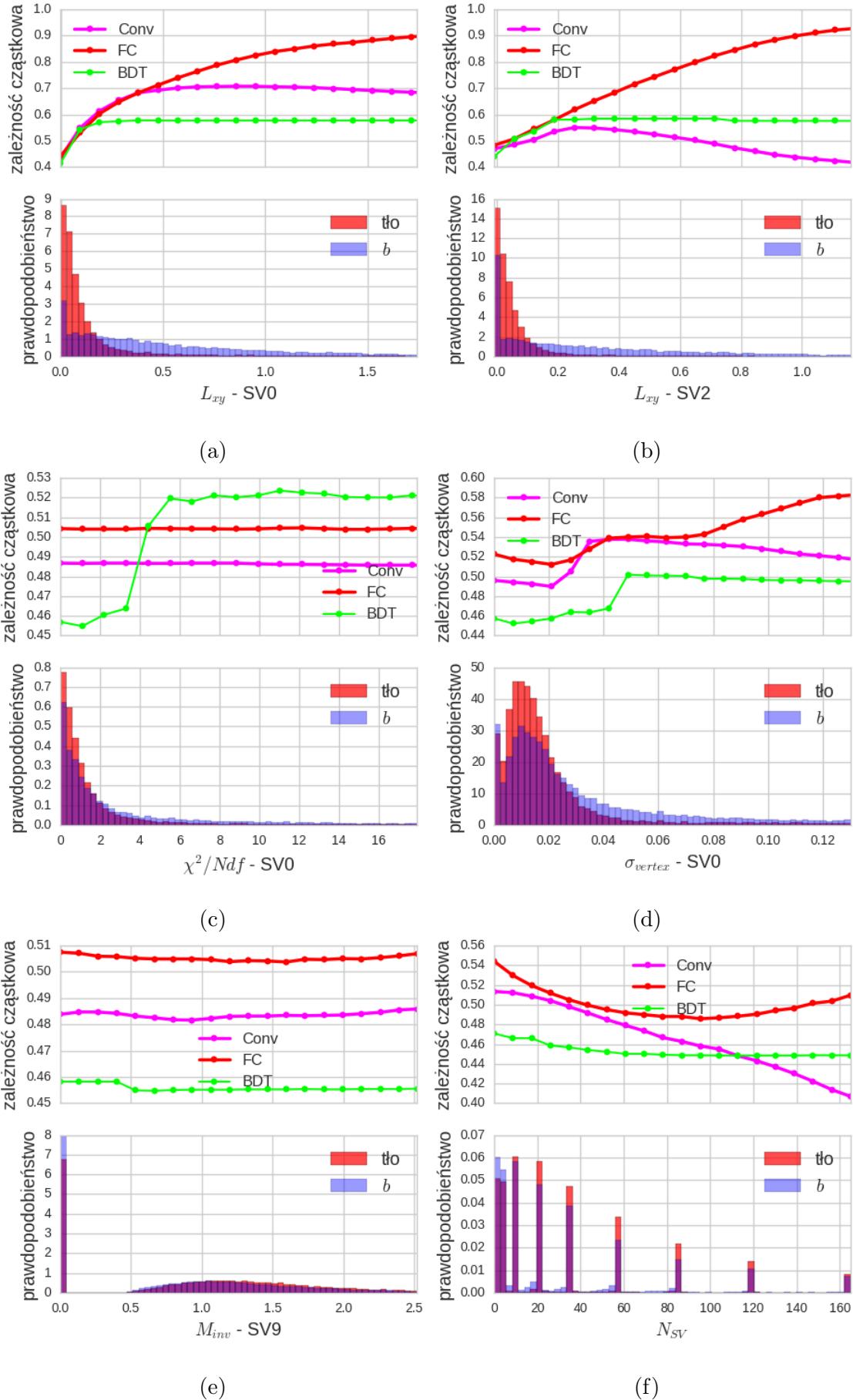
768 Analiza tych pięciu wykresów pozwala na zaobserwowanie ciekawych różnic między algoryt-
769 mami. Pierwsza z nich: zależności cząstkowe wydają się mieć mniejszą amplitudę w przypadku
770 BDT niż w przypadku sieci neuronowych. Może wynikać to z kształtu funkcji – w przypadku
771 BDT jest to złożenie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

779 Druga obserwacja to to asymetria w predykcji BDT na wykresie $IP_Z - C1$. Pomimo wyjąt-
780 kowo symetrycznego rozłożenia wartości tej zmiennej wokół zera, predykcje BDT są wyraźnie

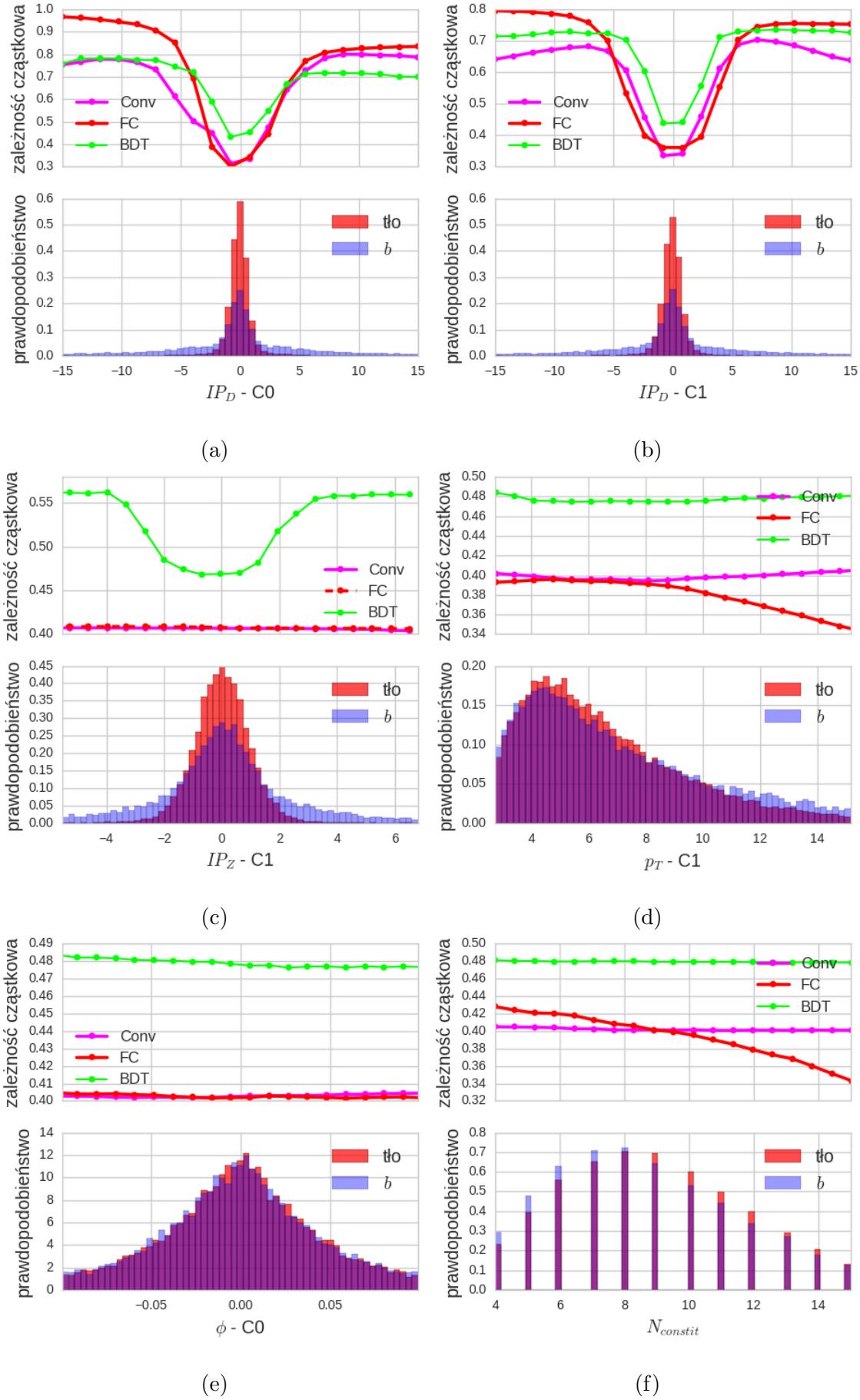
781 przesunięte, z minimum zlokalizowanym wokół wartości -1. Przyczyną takiego zachowania może
782 być specyficzny sposób trenowania drzew decyzyjnych, które w przypadku takiego rozkładu nie
783 mogą podzielić przestrzeń fazowej na 3 części w jednym kroku, co byłoby najbardziej korzystne,
784 ale zmuszone są do dwóch kolejnych podziałów, np. wzduż wartości -2 i 2, z czego pierwszy
785 podział musi niejako złamać symetrię rozkładu. Konieczność wykonania dwóch podziałów, z
786 których pierwszy nie zapewnia dobrej separacji klas może być przyczyną, dla której L_{xy} było
787 chętniej wykorzystywane przez model *merged-BDT* (Tab. 3) – w przypadku tej zmiennej dobrą
788 separację klas daje już pierwszy podział.

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

792 Wreszcie wykresy przedstawione na Rys. 18 e) oraz Rys. 19 e) pokazują, że nawet w przy-
793 padku zmiennych intensywnie wykorzystywanych przy podziałach drzewa (o czym świadczy
794 Tab. 2) zmiana wyłącznie ich wartości ma często znikomy wpływ na predykcje całego modelu.
795 Są to zmienne, których rozkłady dla sygnału i tła prawie się nie różnią. Ich wpływ na predykcje
796 algorytmu ujawnia się dopiero w kombinacji z innymi zmiennymi. Wykorzystanie tych zmien-
797 nych jest dużo trudniejsze dla ludzi i między innymi na tym polega przewaga podejścia analizy
798 wielowymiarowej i uczenia maszynowego. Wykresy te sygnalizują jednocześnie ograniczenie me-
799 tody zależności cząstkowych, która jak każda metoda graficzna nie radzi sobie z przestrzeniami
800 zmiennych o liczbie wymiarów większej niż 3-4.



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*.

5 Podsumowanie i wnioski

W pracy przeanalizowano możliwości użycia algorytmów uczenia maszynowego w problemie identyfikacji dżetów b . Bazowano na danych z symulacji Monte Carlo przeprowadzonych dla detektora ALICE. Sprawdzono działanie trzech rodzajów algorytmów: wzmacnianych drzew decyzyjnych, sieci neuronowych w pełni połączonych oraz konwolucyjnych. Procedurę uczenia przeprowadzono osobno na trzech zbiorach zmiennych, uzyskując tym samym $3 \times 3 = 9$ modeli.

Zaprezentowano wyniki poszczególnych modeli (w postaci krzywych ROC oraz wydajności na identyfikację dżetów b w trzech punktach pracy) oraz obliczono korelacje między ich predykcjami. Sieci konwolucyjne oraz wzmacniane drzewa decyzyjne okazały się dawać lepsze wyniki od sieci w pełni połączonych. Dla najlepszych modeli uzyskano wartości $ROCAUC = 0.925(1)$. Dla prawdopodobieństwa błędnej klasyfikacji tła równego 1% i 10% uzyskano wydajności na identyfikację dżetów b na poziomie kolejno 50% oraz 81%. Predykcje wszystkich modeli okazały się być dosyć silnie skorelowane, przy czym czynnikiem wzmacniającym korelacje okazało się użycie do treningu tych samych zmiennych, natomiast nie zaobserwowano wzmacnienia korelacji dla algorytmów tego samego rodzaju.

W celu lepszego zrozumienia działania modeli przeprowadzono analizę wpływu poszczególnych zmiennych na ich predykce. W tym celu wykorzystano miary istotności zmiennych dostępne dla drzew decyzyjnych oraz wykresy zależności cząstkowych dla wszystkich trzech algorytmów. Analiza pokazuje (tylko dla drzew decyzyjnych), że wszystkie używane wielkości okazały się mieć niepomijalny wpływ na predykce. Ponadto wszystkie modele działają zgodnie z podstawową wiedzą nt. właściwości dżetów b , tzn. zmienne wykorzystywane w klasycznych metodach cięć (takie jak odsunięcie wtórnego wierzchołka L_{xy} czy odległości śladów od pierwotnego wierzchołka $IP_{D,Z}$) mają największy wpływ na predykce algorytmów.

Istnieje kilka sposobów na potencjalną poprawę uzyskanych wyników. Pierwszym z nich mogłyby być wykorzystanie dodatkowych wielkości, związanych z półleptonowym kanałem rozpadu ciężkich hadronów jak np. pęd poprzeczny leptonu względem osi dżetu lub z wewnętrzna strukturą dżetów, jak np. tzw. *pull* [52]. Kolejnym mogłyby być użycie innych algorytmów lub architektur sieci neuronowych, co jednak mojej ocenie ma niewielki niewielki potencjał bez zmiany w danych treningowych. Istotniejsze mogłyby okazać się użycie większej ilości danych, co niemal na pewno poprawiłyby wyniki głębokich sieci neuronowych, które znane są ze świetnego skalowania się w obszarze ogromnych zbiorów danych.

Kolejnym etapem pracy powinna być próba użycia modeli na danych eksperymentalnych. Wiąże się to z szeregiem nowych trudności, głównie związanych z brakiem możliwości bezpośredniej oceny wyników uzyskiwanych przez algorytm, jest jednak niezbędne do wyjścia poza obszar czysto akademickich rozważań. Innym kierunkiem rozwoju byłoby przeprowadzenie podobnej analizy dla dużo bardziej trudnych danych ze zderzeń ciężkich jonów.

837 Bibliografia

- 838 [1] Donald H. Perkins. "Oddziaływanie międzykarkowe i chromodynamika kwantowa". W:
839 *Wstęp do Fizyki Wysokich Energii*. 2 wydr. PWN, 2005, 171–193.
- 840 [2] David J. Gross i Frank Wilczek. "Ultraviolet Behavior of Nonabelian Gauge Theories".
841 W: *Phys. Rev. Lett.* 30 (1973). [,271(1973)], s. 1343–1346. DOI: 10.1103/PhysRevLett.
842 30.1343.
- 843 [3] H. David Politzer. "Reliable Perturbative Results for Strong Interactions?" W: *Phys. Rev.*
844 *Lett.* 30 (1973). [,274(1973)], s. 1346–1349. DOI: 10.1103/PhysRevLett.30.1346.
- 845 [4] C. Patrignani i in. "Review of Particle Physics". W: *Chin. Phys.* C40.10 (2016), s. 100001.
846 DOI: 10.1088/1674-1137/40/10/100001.
- 847 [5] John C. Collins i M. J. Perry. "Superdense Matter: Neutrons Or Asymptotically Free
848 Quarks?" W: *Phys. Rev. Lett.* 34 (1975), s. 1353. DOI: 10.1103/PhysRevLett.34.1353.
- 849 [6] N. Cabibbo i G. Parisi. "Exponential Hadronic Spectrum and Quark Liberation". W:
850 *Phys. Lett.* 59B (1975), s. 67–69. DOI: 10.1016/0370-2693(75)90158-6.
- 851 [7] D. Boyanovsky, H. J. de Vega i D. J. Schwarz. "Phase transitions in the early and the
852 present universe". W: *Ann. Rev. Nucl. Part. Sci.* 56 (2006), s. 441–500. DOI: 10.1146/
853 annurev.nucl.56.080805.140539. arXiv: hep-ph/0602002 [hep-ph].
- 854 [8] Mark G. Alford i Kai Schwenzer. "What the Timing of Millisecond Pulsars Can Teach
855 us about Their Interior". W: *Phys. Rev. Lett.* 113.25 (2014), s. 251102. DOI: 10.1103/
856 PhysRevLett.113.251102. arXiv: 1310.3524 [astro-ph.HE].
- 857 [9] Vardan Khachatryan i in. "Evidence for collectivity in pp collisions at the LHC". W:
858 *Phys. Lett.* B765 (2017), s. 193–220. DOI: 10.1016/j.physletb.2016.12.009. arXiv:
859 1606.06198 [nucl-ex].
- 860 [10] Jaroslav Adam i in. "Enhanced production of multi-strange hadrons in high-multiplicity
861 proton-proton collisions". W: *Nature Phys.* 13 (2017), s. 535–539. DOI: 10.1038/nphys4111.
862 arXiv: 1606.07424 [nucl-ex].
- 863 [11] Matteo Cacciari, Gavin P. Salam i Gregory Soyez. "The Anti-k(t) jet clustering algori-
864 thm". W: *JHEP* 04 (2008), s. 063. DOI: 10.1088/1126-6708/2008/04/063. arXiv:
865 0802.1189 [hep-ph].
- 866 [12] Vardan Khachatryan i in. "Charged-particle nuclear modification factors in PbPb and
867 pPb collisions at $\sqrt{s_{NN}} = 5.02$ TeV". W: *JHEP* 04 (2017), s. 039. DOI: 10.1007/
868 JHEP04(2017)039. arXiv: 1611.01664 [nucl-ex].
- 869 [13] Betty Abelev i in. "Centrality Dependence of Charged Particle Production at Large Trans-
870 verse Momentum in Pb–Pb Collisions at $\sqrt{s_{NN}} = 2.76$ TeV". W: *Phys. Lett.* B720 (2013),
871 s. 52–62. DOI: 10.1016/j.physletb.2013.01.051. arXiv: 1208.2711 [hep-ex].
- 872 [14] Carlos A. Salgado i Urs Achim Wiedemann. "Calculating quenching weights". W: *Phys.*
873 *Rev.* D68 (2003), s. 014008. DOI: 10.1103/PhysRevD.68.014008. arXiv: hep-ph/0302184
874 [hep-ph].
- 875 [15] Yuri L. Dokshitzer i D. E. Kharzeev. "Heavy quark colorimetry of QCD matter". W:
876 *Phys. Lett.* B519 (2001), s. 199–206. DOI: 10.1016/S0370-2693(01)01130-3. arXiv:
877 hep-ph/0106202 [hep-ph].
- 878 [16] Georges Aad i in. "Performance of b-Jet Identification in the ATLAS Experiment". W:
879 *JINST* 11.04 (2016), P04008. DOI: 10.1088/1748-0221/11/04/P04008. arXiv: 1512.
880 01094 [hep-ex].

- 881 [17] M. Aaboud i in. “Measurements of b-jet tagging efficiency with the ATLAS detector using
 882 $t\bar{t}$ events at $\sqrt{s} = 13$ TeV”. W: *JHEP* 08 (2018), s. 089. DOI: 10.1007/JHEP08(2018)089.
 883 arXiv: 1805.01845 [hep-ex].
- 884 [18] Serguei Chatrchyan i in. “Identification of b-quark jets with the CMS experiment”. W:
 885 *JINST* 8 (2013), P04013. DOI: 10.1088/1748-0221/8/04/P04013. arXiv: 1211.4462
 886 [hep-ex].
- 887 [19] A. M. Sirunyan i in. “Identification of heavy-flavour jets with the CMS detector in pp
 888 collisions at 13 TeV”. W: *JINST* 13.05 (2018), P05011. DOI: 10.1088/1748-0221/13/
 889 05/P05011. arXiv: 1712.07158 [physics.ins-det].
- 890 [20] Linus Feldkamp. “Study of b-jet tagging performance in ALICE”. W: *J. Phys. Conf. Ser.*
 891 509 (2014), s. 012061. DOI: 10.1088/1742-6596/509/1/012061. arXiv: 1310.2817
 892 [hep-ex].
- 893 [21] Rüdiger Haake. “Machine and deep learning techniques in heavy-ion collisions with ALICE”.
 894 W: *Proceedings, 2017 European Physical Society Conference on High Energy Physics*
 895 (*EPS-HEP 2017*): *Venice, Italy, July 5-12, 2017*. T. EPS-HEP2017. 2017. DOI: 10.22323/
 896 1.314.0498. arXiv: 1709.08497 [physics.data-an]. URL: <https://pos.sissa.it/314/498/pdf>.
- 897 [22] Roel Aaij i in. “Identification of beauty and charm quark jets at LHCb”. W: *JINST* 10.06
 898 (2015), P06013. DOI: 10.1088/1748-0221/10/06/P06013. arXiv: 1504.07670 [hep-ex].
- 900 [23] K. Aamodt i in. “The ALICE experiment at the CERN LHC”. W: *JINST* 3 (2008),
 901 S08002. DOI: 10.1088/1748-0221/3/08/S08002.
- 902 [24] Betty Bezverkhny Abelev i in. “Performance of the ALICE Experiment at the CERN
 903 LHC”. W: *Int. J. Mod. Phys.* A29 (2014), s. 1430044. DOI: 10.1142/S0217751X14300440.
 904 arXiv: 1402.4476 [nucl-ex].
- 905 [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).
- 906 [26] Sotiris B Kotsiantis, I Zaharakis i P Pintelas. “Supervised machine learning: A review
 907 of classification techniques”. W: *Emerging artificial intelligence applications in computer*
 908 *engineering* 160 (2007), s. 3–24.
- 909 [27] Marcin Wolter. “Metody analizy wielu zmiennych w fizyce wysokich energii”. Prac. dokt.
 910 IFJ PAN, 2012.
- 911 [28] Leo Breiman. “Bagging predictors”. W: *Machine learning* 24.2 (1996), s. 123–140.
- 912 [29] Yoav Freund i Robert E Schapire. “A decision-theoretic generalization of on-line learning
 913 and an application to boosting”. W: *Journal of computer and system sciences* 55.1 (1997),
 914 s. 119–139.
- 915 [30] Tianqi Chen i Carlos Guestrin. “XGBoost: A Scalable Tree Boosting System”. W: *Proce-
 916 dings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and*
 917 *Data Mining*. KDD ’16. San Francisco, California, USA: ACM, 2016, s. 785–794. ISBN:
 918 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: <http://doi.acm.org/10.1145/2939672.2939785>.
- 919 [31] James Bergstra i Yoshua Bengio. “Random Search for Hyper-parameter Optimization”.
 920 W: *J. Mach. Learn. Res.* 13 (lut. 2012), s. 281–305. ISSN: 1532-4435. URL: <http://dl.acm.org/citation.cfm?id=2188385.2188395>.

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

- 970 [50] Matteo Cacciari i Gavin P. Salam. “Pileup subtraction using jet areas”. W: *Phys. Lett.*
971 B659 (2008), s. 119–126. DOI: 10.1016/j.physletb.2007.09.077. arXiv: 0707.1378
972 [[hep-ph](#)].
- 973 [51] Andrew P Bradley. “The use of the area under the ROC curve in the evaluation of
974 machine learning algorithms”. W: *Pattern recognition* 30.7 (1997), s. 1145–1159.
- 975 [52] Jason Gallicchio i Matthew D. Schwartz. “Seeing in Color: Jet Superstructure”. W: *Phys.*
976 *Rev. Lett.* 105 (2010), s. 022001. DOI: 10.1103/PhysRevLett.105.022001. arXiv: 1001.
977 5027 [[hep-ph](#)].

978 Dodatek A Metryki

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

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

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

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

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

985

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.

986 Podanie tylko jednej metryki jest zwykle niewystarczające i stosuje się pary metryk, np.
 987 precyzja - czułość. Jeśli predykcja klasyfikatora ma charakter ciągły, to poprzez zmienianie
 988 wartości progowej otrzymuje się różne punkty pracy, scharakteryzowane przez wartości obu
 989 metryk. Kolejne punkty pracy wykreślone np. na wykresie $TPR(FPR)$ dają krzywą nazywaną
 990 krzywą ROC. Wykres ten w przypadku klasyfikatora losowego jest prostą łączącą punkty o
 991 współrzędnych $(0,0)$ i $(1,1)$. Im lepszy klasyfikator tym bardziej wykres wygięty jest w stronę
 992 punktu $(0,1)$. Pole pod tą krzywą (ang. *ROC Area Under Curve – ROC AUC*) jest kolejną
 993 metryką, bardzo często wykorzystywaną w praktyce, gdyż łączy w sobie wszystkie możliwe
 994 punkty pracy, jest także odporne na niezrównoważenie klas. Pole pod krzywą *ROC* przyjmuje
 995 wartość 0.5 dla klasyfikatora losowego oraz 1 dla idealnego.

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

1000 Dodatek B Skróty i oznaczenia

1001 W pracy używane są następujące skróty i oznaczenia:

- 1002 • algorytmy:
1003 FC – sieci neuronowe w pełni połączone,
1004 $Conv$ – sieci neuronowe konwolucyjne,
1005 BDT – wzmacniane drzewa decyzyjne
- 1006 • zbiory danych:
1007 SV – zestaw zawierający tylko zmienne związane z wtórnymi wierzchołkami,
1008 $constit$ – zestaw zawierający tylko zmienne związane z częstotliwościami tworzącymi dżet,
1009 $merged$ – zestaw zawierający wszystkie zmienne (tj. $SV + Constit +$ zmienne charaktery-
1010 zujące dżet jako całość)
- 1011 • modele (zestaw danych + algorytm): nazywane są wg wzoru:
1012 (oznaczenie_zbioru_danych)-(oznaczenie_algorytmu),
1013 np. $SV\text{-}Conv$ oznacza konwolucyjną sieć neuronową wytrenowaną na zmiennych zwi-
1014 ązanych z wtórnymi wierzchołkami a $merged\text{-}BDT$ - wzmacniane drzewa decyzyjne, do
1015 treningu których użyte zostały wszystkie zmienne. Zapis $merged\text{-}X$ stanowi zbiorcze ozna-
1016 czenie modeli $merged\text{-}SV$, $merged\text{-}Conv$ i $merged\text{-}BDT$.
- 1017 • poszczególne zmienne (kolumny w zbiorze danych) oznaczane są według wzoru:
1018 (nazwa_zmiennej) – (numer_obiektu),
1019 np. $\sigma_{L_{xy}}$ – $SV2$ oznacza niepewność wyznaczenia L_{xy} dla wtórnego wierzchołka nr 2, a
1020 $IP_Z\text{-}C5$ – parametr zderzenia wzdłuż osi wiązki cząstki nr 5 – numery na posortowanych
1021 listach (Rozdz. 3, gdzie znajdują się także opisy wielkości fizycznych).