

# **Spis treści**

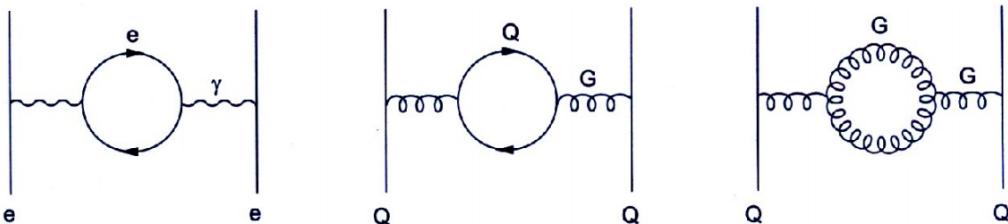
<b>1 Fizyka dżetów cząstek</b>	<b>2</b>
1.1 Chromodynamika kwantowa . . . . .	2
1.2 Plazma kwarkowo-gluonowa . . . . .	3
1.3 Dżety . . . . .	4
1.4 Dżety $b$ . . . . .	4
1.5 Eksperyment ALICE . . . . .	5
<b>2 Uczenie maszynowe</b>	<b>9</b>
2.1 Wzmacniane drzewa decyzyjne . . . . .	9
2.2 Sieci neuronowe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

# 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  $\alpha$ , którą nazywamy efektywną  
43 lub biegącą stałą sprzężenia (ang. *running coupling constant*).



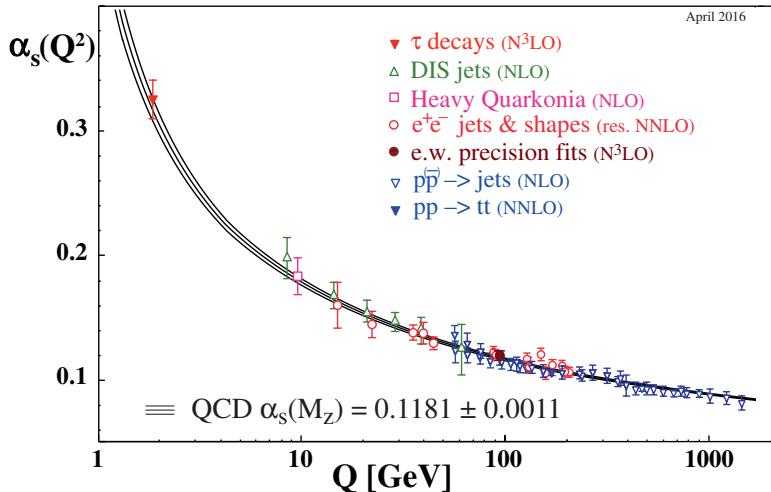
Rysunek 1: Diagramy Feynmana opisujące polaryzację próżni w *QED* (lewy) i *QCD* (środkowy i prawy). Rysunki lewy i środkowy są swoimi odpowiednikami w tych dwóch teoriach, natomiast prawy, w którym oddziałują jedynie bozony pośredniczące nie ma swojego odpowiednika w *QED*. Źródło: [1]

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  $\alpha_{em}$  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  $\alpha_S$   
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. Obecnie najpowszechniej stosowanym algorytmem jest algorytm *anti-kt*  
93 [11].

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

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

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

## 121 1.4 Dżety *b*

### 122 1.4.1 Właściwości

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

128 Identyfikacja dżetów *b* jest sporym wyzwaniem ze względu na zdecydowanie częściej wy-  
129 stępujące dżety lekkie, tj. powstałe z hadronizacji kwarków *u,d,s* lub gluonów. Rozpoznawanie  
130 dżetów *b* bazuje na charakterystycznych właściwościach hadronów zawierających kwark piękny:

131 relatywnie długim czasie życia oraz (w mniejszym stopniu) na ich półleptonowych rozpadach o  
132 względnej częstości rozpadu w tym kanale (ang. *branching ratio*) na poziomie 10% [4].

#### 133 1.4.2 Przegląd algorytmów używanych do identyfikacji dżetów *b* na LHC

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

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

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

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

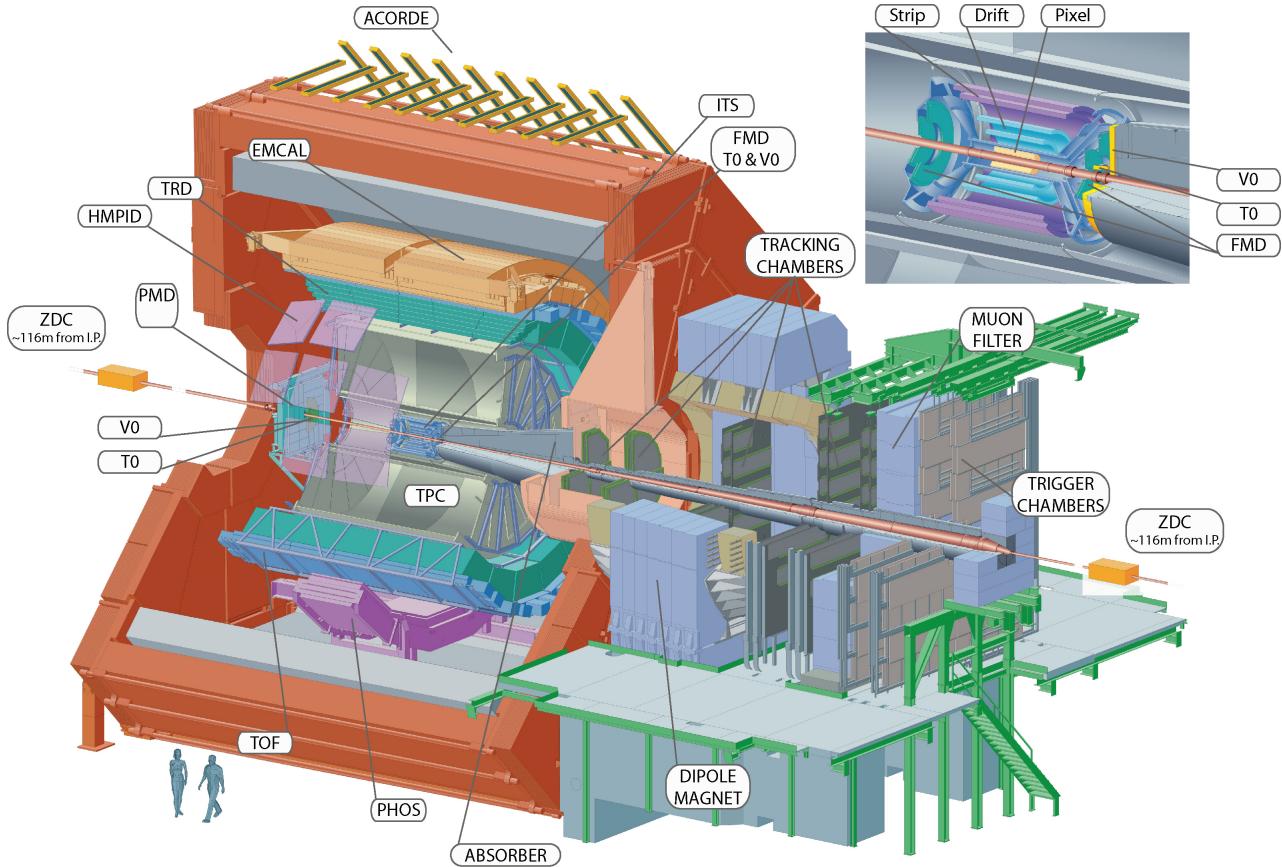
159 Znaczącą poprawę zdolności predykcyjnej można uzyskać łącząc kilka różnych modeli. Al-  
160 gorytm łączący może pobierać na wejściu albo tylko predykcje klasyfikatorów niższego poziomu  
161 (CMS: cMVAv2) lub dodatkowo także ich zmienne wejściowe (ATLAS: MV2, DL1). Do scalania  
162 używane są zwykle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

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

### 168 1.5 Eksperiment ALICE

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

176  $\eta$  (ang. *pseudorapidity*)<sup>1</sup>. Detektor ALICE został zoptymalizowany do mierzeniach takich  
 177 przypadków, jak również pod kątem rekonstrukcji i identyfikacji cząstek o szerokim zakresie  
 178 pędów ( $100 \text{ MeV}/c - 100 \text{ GeV}/c$ ).



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

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

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

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

wiązki i kącie azymutalnym. Trzecia składowa trajektorii jest otrzymywana na podstawie czasu dotarcia elektronów do elektrod. TPC jest najwolniejszym detektorem ALICE (ze względu na ograniczający czas dryfu elektronów wynoszący  $\sim 90 \mu\text{s}$ ), użycie detektora tego typu podyktowane jest jego zdolnością do mierzenia śladów tysięcy cząstek spodziewanych w centralnych zderzeniach PbPb.

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

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

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

• kalorymetry

- Photon Spectrometer (PHOS) – elektromagnetyczny kalorymetr o wysokiej rozdzielczości energetycznej i przestrzennej (podzielony na kryształy o rozmiarze poprzecznym  $2.2 \times 2.2 \text{ cm}$ , co odpowiada przestrzeni fazowej  $\Delta\eta \times \Delta\phi = 0.004 \times 0.004$ ). PHOS pokrywa zakres pseudopospieszności  $|\eta| < 0.12$  i kąta azymutalnego równy  $100^\circ$ . PHOS ma za zadanie identyfikację i pomiar czteropędów fotonów, w szczególności tych niepochodzących z rozpadu innych cząstek (ang. *direct photons*) oraz lekkich mezonów neutralnych (np.  $\pi^0$ ) przez dwufotonowy kanał rozpadu.
- Electromagnetic Calorimeter (EMCal) – drugi elektromagnetyczny kalorymetr ALICE o mniejszej ziarnistości ( $\Delta\eta \times \Delta\phi = 0.014 \times 0.014$ ), ale dużo większej akceptancji ( $|\eta| < 0.7$ ,  $\Delta\phi = 107^\circ$ ). EMCal poprawia możliwości ALICE w zakresie pomiarów tłumienia dżetów, pozwalając na wyznaczanie neutralnej składowej energii dżetów (energię niesioną przez neutralne cząstki). Dzięki innej charakterystyce dla elektronów i hadronów (elektrony typowo deponują niemal całą energię a hadrony tylko niewielką część) pozwala je odróżnić na podstawie stosunku zmierzonych 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.

- 241 • Muon spectrometer – spektrometr mionowy, złożony z dwóch pasywnych absorberów,  
242 znajdujących się między nimi 10 warstw detektora śladowego oraz komór systemu wy-  
243 zwalającego na końcu. Przedni absorber, gruby na 4 metry ( $\sim 60X_0$ ) wykonany z betonu  
244 i grafitu, zatrzymuje hadrony oraz miony o niższych energiach (np. z rozpadów pionów  
245 i kaonów). Jest on zoptymalizowany aby minimalizować rozpraszanie mionów i zapewnić  
246 ochronę pozostałych detektorów ALICE przed wtórnymi cząstkkami powstałymi w jego  
247 materiale. Komory śladowe mają zdolność rozdzielczą ok.  $100 \mu\text{m}$ , co pozwala osiągnąć  
248 wysoką rozdzielcość przy wyznaczaniu masy niezmienniczej rzędu  $100 \text{ MeV}/c^2$ . Spek-  
249 trometr mionowy służy głównie do mierzenia mezonów wektorowych ( $\omega$ ,  $\phi$ ,  $J/\Psi$ ,  $\Upsilon$ )  
250 rozpadających się w kanale  $\mu^+ \mu^-$ .
- 251 • Detektory przednie, wyznaczające min. centralność zderzeń oraz płaszczyznę reakcji.
- 252     – ZDC – zespół czterech kalorymetrów (po dwa do pomiaru protonów i neutronów)  
253         pokrywających inną przestrzeń fazową ponieważ tory protonów są odchylane przez  
254         pole magnetyczne, mierzących energię nukleonów nieuczestniczących w zderzeniu  
255         tzw. obserwatorów, co pozwala na określenie liczby nukleonów oddziałujących, tzw.  
256         uczestników. Znajdują się one 116 m od miejsca interakcji.
- 257     – PMD – detektor gazowy mierzący krotkości oraz rozkład przestrzenny fotonów
- 258     – FMD – krzemowy detektor paskowy mierzący precyzyjnie liczbę naładowanych cza-  
259         stek w zakresie pseudopospieszności wykraczającym poza akceptancję detektora ITS.
- 260     – V0 – liczniki scyntylacyjne położone po obu stronach detektora, używane w sys-  
261         temie wyzwalania o minimalnym obciążeniu (ang. *minimum bias trigger*) – wymóg  
262         obecności sygnału w obu detektorach pozwala na odrzucenie przypadków tła z od-  
263         ziaływania wiązki protonów z reszkami gazu obecnymi w rurach próżniowych.
- 264     – T0 – zespół dwóch liczników Cherenkova, który dostarcza dokładny czas interak-  
265         cji potrzebny dla detektora TOF, pozwala także na śledzenie świetlności w czasie  
266         rzeczywistym.

## 2 Uczenie maszynowe

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

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

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

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

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

### 2.1 Wzmacniane drzewa decyzyjne

Wzmacniane drzewa decyzyjne są jednym z rozwinięć klasycznego algorytmu drzewa decyzyjnego. Pojedyncze drzewo decyzyjne dzieli przestrzeń cech uczących przy pomocy prostopadłych cięć, na mniejsze/większe niż zadana wartość w przypadku zmiennej ciągłej lub na należące/nie należące do danej klasy w przypadku zmiennej kategorycznej. Każdy podział, nazywany węzłem, daje dwie gałęzie, które można dalej niezależnie dzielić aż do ostatniego poziomu (liści). Kolejne podziały wybierane są tak, aby zbiory przykładów wpadające do poszczególnych gałęzi były jak najbardziej jednorodne. Stosuje się różne miary nieporządku takie tak: indeks Gini  $G_L$  lub entropia  $S_L$ <sup>3</sup>.

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

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

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

<sup>3</sup>  $G_L = 1 - \sum_k p_k^2$  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  $L$

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

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

### 317 Parametry i sposób trenowania drzew decyzyjnych na analizowanych danych

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

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

334 Parametry optymalizowane w opisanym procesie to:

- 335 • *max\_depth* – maksymalna głębokość każdego drzewa (niekoniecznie osiągana)
- 336 • *n\_estimators* – liczba drzew
- 337 • *learning\_rate* – parametr szybkości uczenia, komplementarny do *n\_estimators*, w praktyce  
338 można ustalić liczbę drzew i szukać optymalnej szybkości uczenia
- 339 • *subsample*, *colsample\_bytree*, *colsample\_bylevel* – parametry regularyzacyjne określające  
340 ułamek kolejno: wierszy użytych do trenowania każdego drzewa, kolumn użytych w ka-  
341 dym drzewie (cechy losowane raz dla danego drzewa), kolumn użytych przy każdym po-  
342 dziale (cechy losowane przy każdym podziale)
- 343 •  $\gamma$  – minimalny zysk w postaci zmniejszenia wartości funkcji straty konieczny do wykonania  
344 podziału

## 345 2.2 Sieci neuronowe

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

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

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

### 357 Sieci w pełni połączone

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

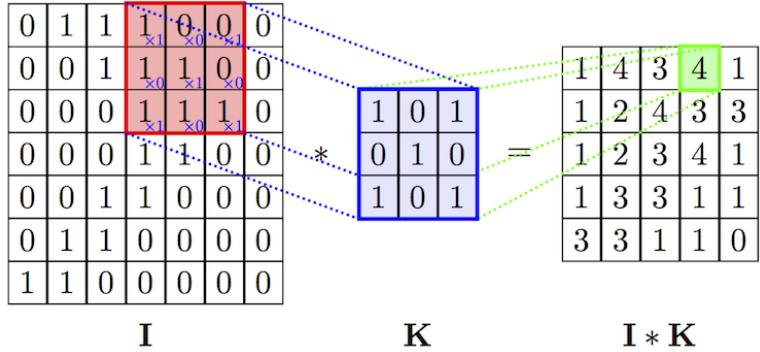
367 Istnieje twierdzenie o sieciach neuronowych jako uniwersalnych aproksymatorach funkcji  
368 (ang. *universal approximation theorem*) [34], mówiące, że już sieć neuronowa o jednej warstwie  
369 ukrytej jest zdolna do przybliżenia dowolnej funkcji z dowolną dokładnością. Twierdzenie to  
370 nie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

### 375 Sieci konwolucyjne

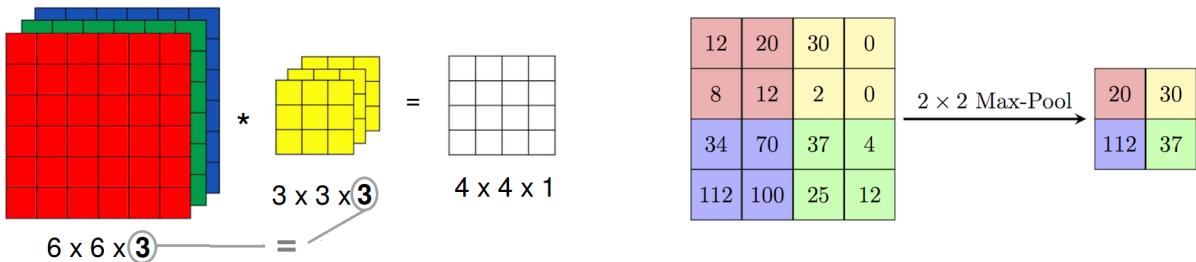
376 Jednym ze sposobów na ograniczenie liczby trenowanych parametrów jest użycie konwolucyj-  
377 nych sieci neuronowych [35] (bardziej poprawną choć rzadko używaną nazwą w języku polskim  
378 jest sieć splotowa). Są one inspirowane połączeniami w korze wzrokowej zwierząt i wywodzą  
379 się z badań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

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

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



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



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

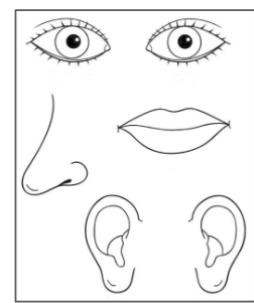
Rysunek 6: Działanie warstwy typu *max-pool*. Źródło: [36].

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

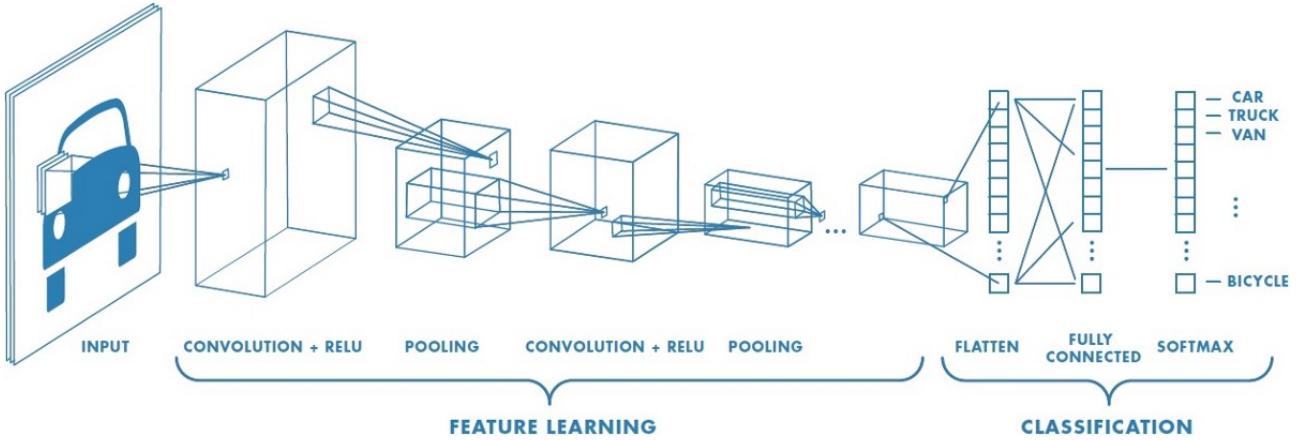
Typowa architektura stosowana w przypadku sieci konwolucyjnych jest następująca: najpierw warstwy konwolucyjne (pomiędzy nimi czasem warstwy typu *max-pool*), następnie wszystkie filtry są rozwijane i składane w długi jednowymiarowy wektor, który przekazywany jest do warstw typu *FC*. W przypadku problemu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

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

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



Rysunek 7: Źródło: [38].

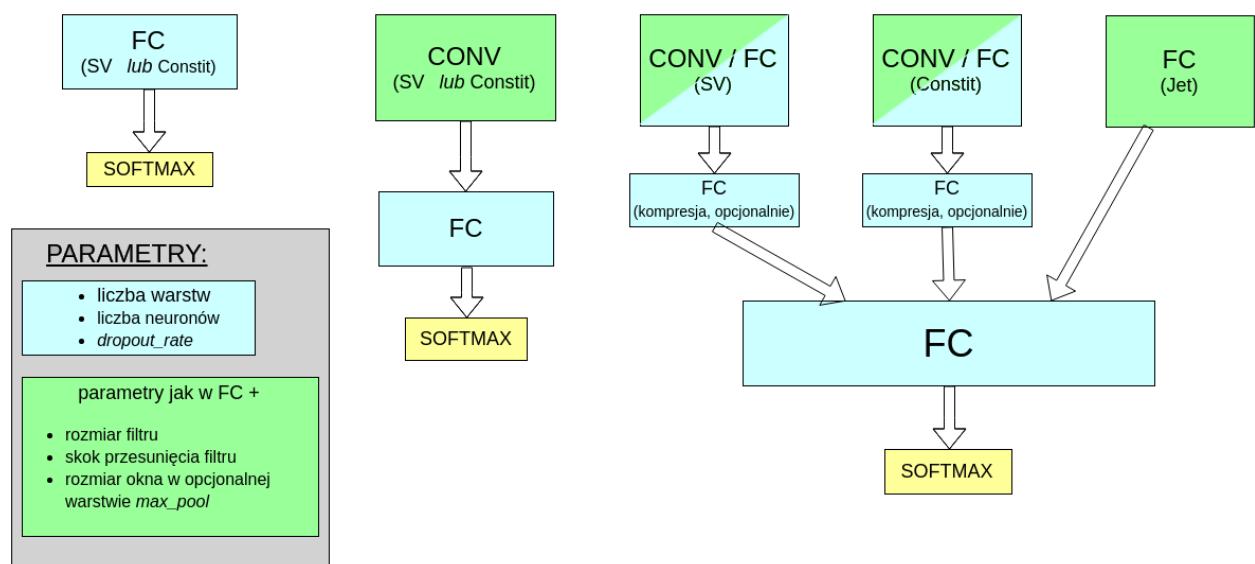


Rysunek 8: Typowa struktura stosowana w sieciach konwolucyjnych. Warstwy konwolucyjne mają za zadanie wydobywać cechy, na podstawie których późniejsze warstwy dokonują klasyfikacji. Widoczna jest charakterystyczna stopniowa zmiana rozmiaru przekazywanej macierzy: rozmiar poprzeczny maleje kosztem głębokości,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

#### 417 Hiperparametry i trenowanie sieci neuronowych na analizowanych danych

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

420 Testowane były trzy architektury: sieci w pełni połączone, sieci konwolucyjne oraz sieć  
421 złożona z dwóch gałęzi, osobnych dla wtórnych wierzchołków i częstek tworzących dżet (Rozdz.  
422 3) przedstawione schematycznie na Rys. 9. Do trenowania sieci wykorzystano wysokopoziomową  
423 bibliotekę **Keras** [40] korzystającą z silnika obliczeniowego zaimplementowanego w **TensorFlow**  
424 [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.

425 Zestaw hiperparametrów definiujący działanie sieci w pełni połączonej:

- 426 • liczba warstw ukrytych

- 427 • liczba neuronów w każdej warstwie

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

- 431 • algorytm optymalizacyjny – spadek gradientowy lub jego wariacje

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

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

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

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

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

441 Ponadto dla sieci konwolucyjnych:

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

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

- 444 • rozmiar filtrów i długość skoku przy ich przesuwaniu

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

- 449 • *batch\_size* zawsze równy 64 (inne testowane wartości: 16, 32, 128)

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

- 453 • funkcję aktywacji: *ReLU*

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

- 455 • stałe w trakcie treningu wartości parametru szybkości uczenia

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

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

- 460 • liczby neuronów/filtrów w warstwach będące zawsze potęgami dwójki oraz stałą liczbę w  
461 kolejnych warstwach lub zmieniającą się o stały czynnik, np. 256-128-64, 128-128-128 lub  
462 16-32-64
- 463 • liczba warstw *FC*: 2-8, konwolucyjnych 2-6

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

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

### 473 2.3 Dyskusja użycia dwóch algorytmów

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

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

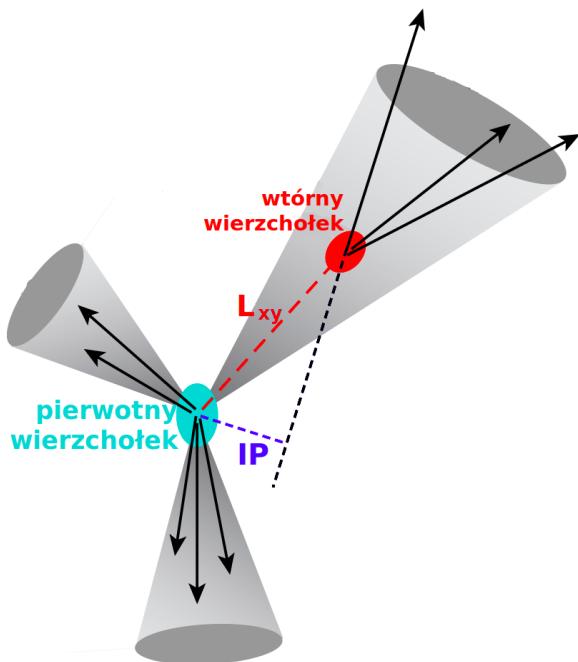
487 Z kolei drzewa decyzyjne posiadają stosunkowo niewielką liczbę parametrów, a ich trenowa-  
488 nie jest bardzo szybkie co pozwala na ich ekstensywne przeszukiwanie i otrzymanie wyników,  
489 które można uznać za optymalne dla tego algorytmu. Ponadto, w przypadku drzew istnieją  
490 niewymagające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

### 492 3 Dane

493 Dane użyte w analizie pochodzą z symulacji Monte Carlo zderzeń proton-proton przy energii w  
 494 układzie środka masy równej  $\sqrt{s} = 13$  TeV dostępnych na serwerach eksperymentu ALICE. Są  
 495 to pełne symulacje detektora ALICE, wykorzystujące generator zderzeń Pythia8 [45] (wersja:  
 496 Monash2013 [46]) oraz pakiet Geant3 [47] do transportu cząstek przez materiał detektora.  
 497 Korzystano ze specjalnych symulacji w tzw. *binach p<sub>T</sub>*, które zapewniają lepszą statystykę dla  
 498 wysokich pędów.

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

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



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

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

513 Lista używanych zmiennych:

- 514 • Zmienne charakteryzujące dżet:
  - $\eta_{jet}, \phi_{jet}$  – pseudopospiesznosć i kąt azymutalny osi dżetu
  - $p_{T,jet}$  – pęd poprzeczny dżetu

- 515           –  $M_{jet} = \sqrt{(\sum_i E_i)^2 - (\sum_i \vec{p}_i)^2}$  – masa dżetu, gdzie  $E_i$  i  $\vec{p}_i$  to energia i pęd kolejnych  
 516           częstek tworzących dżet
- 517           –  $A_{jet}$  – powierzchnia dżetu wyliczana przez algorytm *anti-kt* w płaszczyźnie  $(\eta, \phi)$ .  
 518           Do powierzchni dżetu zaliczany jest każdy element powierzchni, w którym dodanie  
 519           częstki o nieskończenie małym pędzie poprzecznym sprawi, że zostanie ona zaliczona  
 520           do tego dżetu [50]
- 521           –  $\rho_{bckg}$  – gęstość tła w danym zdarzeniu

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

- 523           –  $\eta, \phi$  – pseudopospieszność i kąt azymutalny cząstki względem osi dżetu
- 524           –  $p_T$  – pęd poprzeczny cząstki
- 525           –  $IP_D$  – rzut na kierunek poprzeczny wektora parametru zderzenia
- 526           –  $IP_Z$  – rzut na oś  $z$  wektora parametru zderzenia
- 527           –  $N_{Constit}$  – liczba cząstek tworzących dżet

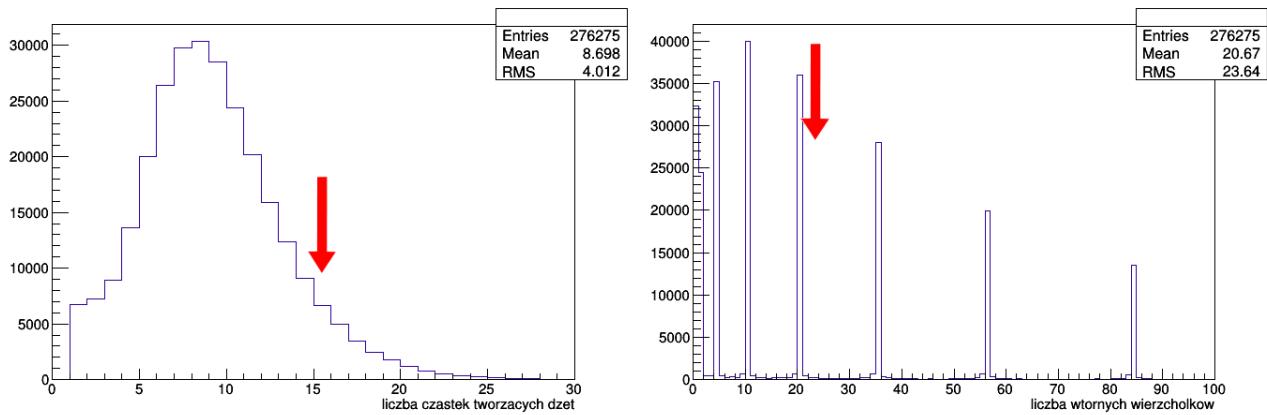
528       • Zmienne opisujące wtórne wierzchołki:

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

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

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

555       Następną kwestią jest wybór wielkości decydującej o kolejności ułożenia elementów, tj. która  
 556       cząstka będzie cząstką nr 1 a która nr 5. Losowe ułożenie elementów sprawiło, że bezpośrednio



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

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

## 566 4 Analiza

### 567 4.1 Dobór metryki

568 Bardzo ważnym elementem w trenowaniu algorytmów uczenia maszynowego jest dobór odpowiednich metryk. Kilka najczęściej używanych metryk wymieniono w Tab. A1. Klasycznym 569 złym przykładem jest używanie dokładności (ang. *accuracy*) do oceniania klasyfikacji binarnej 570 w przypadku dużego niezrównoważenia klas – algorytm przewidujący zawsze klasę większą 571 572 składową może osiągnąć dużą wartość dokładności będąc jednocześnie bardzo słabym modelem.

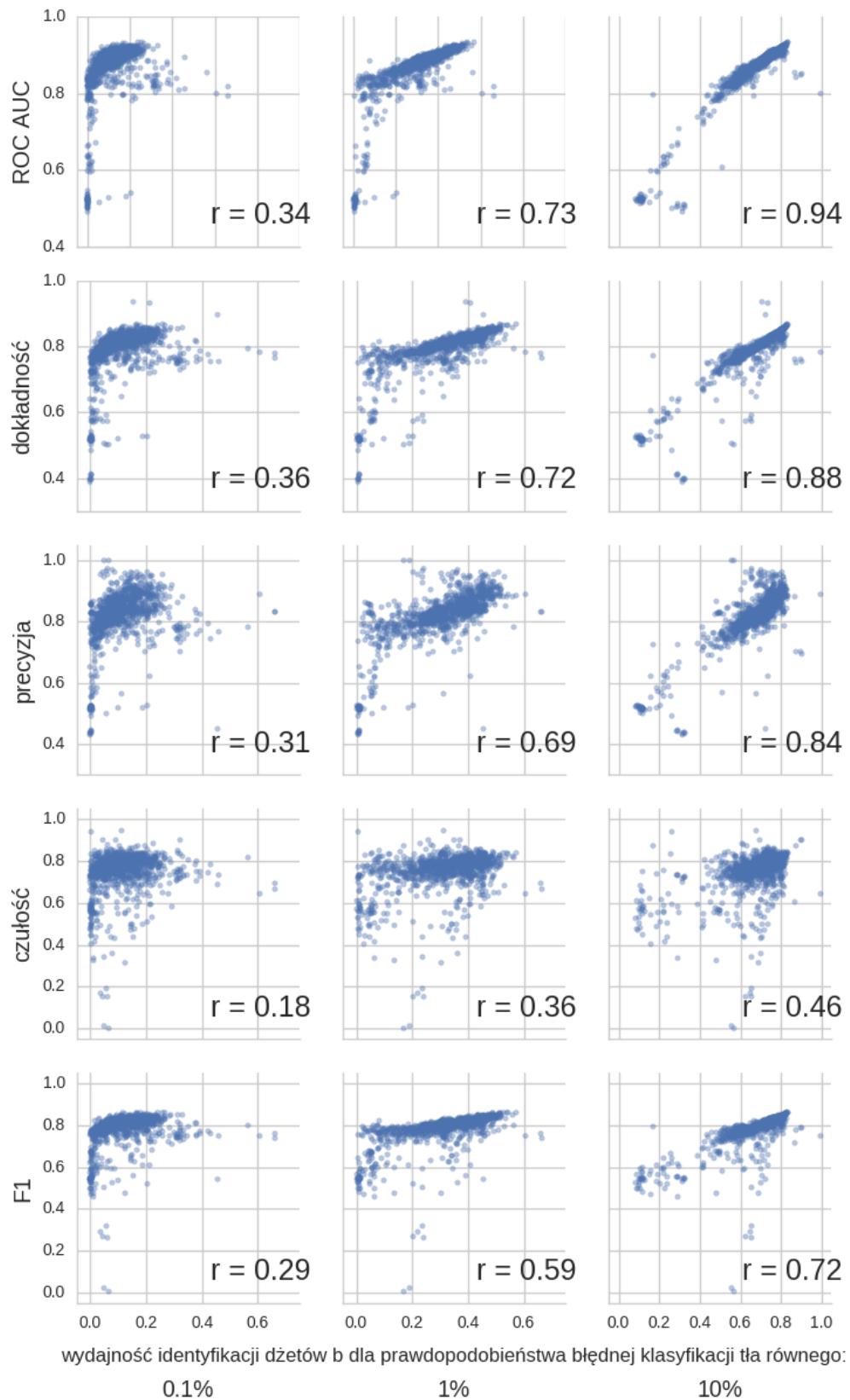
573 Używanie i porównywanie kilku miar efektywności jest często niepraktyczne dlatego dobrze 574 jest wybrać jedną metrykę. Przy jej wyborze należy kierować się potencjalnymi zastosowaniami 575 modelu. W tym przypadku są to analizy fizyczne, które mogą mieć różne wymagania dotyczące 576 czystości i liczebności otrzymywanych próbek, a co za tym idzie, preferować inne punkty 577 pracy zdefiniowane jako pary liczb: wydajność poprawnej klasyfikacji dżetów  $b$  (ang. *tagging* 578 *efficiency = true positive rate = recall*), i ułamek niepoprawnie zaklasyfikowanych przypadków 579 tła (ang. *mistagging rate = false positive rate*).

580 Naturalnym wyborem wydaje się pole pod powierzchnią krzywej *ROC* (ang. *ROC Area Under Curve – ROC AUC*) [51]. Potencjalną przeszkodą może być zakres rozsądnego wartości 581 prawdopodobieństwa błędnej klasyfikacji przypadków tła: dżety  $b$  stanowią tylko kilka procent 582 liczby wszystkich dżetów, zatem z punktu widzenia analizy dopuszczalne będą punkty pracy 583 zapewniające wydajność identyfikacji dżetów  $b$  ok. 10 – 100 razy większą niż częstość niepoprawnego 584 zaklasyfikowania przypadków tła. Oznacza to, że zdecydowana większość punktów 585 pracy znajdujących się na krzywej *ROC* jest nie do zaakceptowania – interesujące są tylko te 586 o najniższych częstościach błędnej klasyfikacji.

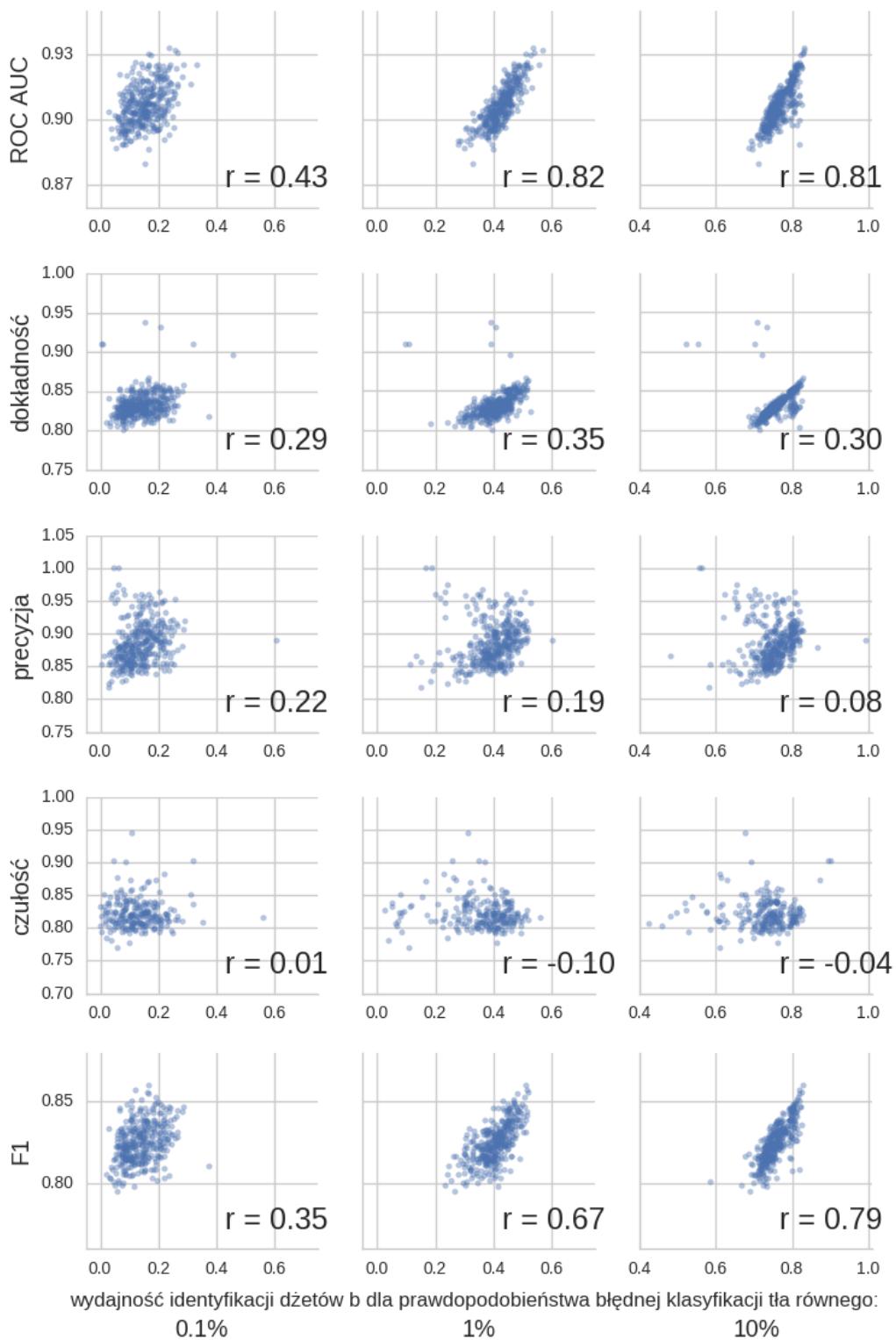
587 Aby ilościowo porównywać różne algorytmy wprowadzone zostaną trzy punkty pracy: o 588 prawdopodobieństwie błędnej klasyfikacji tła równej 0.1%, 1% oraz 10%. Na Rys. 12 przed- 589 stawione zostały zależności poszczególnych metryk od wydajności identyfikacji dżetów  $b$  w 590 tych trzech punktach pracy. Każdy punkt odpowiada jednemu eksperymentowi (dla dowolnego 591 algorytmu) przeprowadzonemu w trakcie przygotowywania analizy. Daje to pogląd, na to war- 592 tościami której metryki należy się kierować przy wyborze algorytmu i hiperparametrów, aby 593 zapewnić sobie jednocześnie wysokie wartości wydajności na identyfikację dżetów  $b$  w wybra- 594 nych punktach pracy. Najwyższe koreacje występują dla punktu pracy o najwyższym prawdo- 595 podobieństwie błędnej klasyfikacji tła – jak będzie to pokazane później wyniki dla tego punktu 596 pracy są najbardziej stabilne. Spośród analizowanych metryk najwyższe wartości współczyn- 597nika Pearsona otrzymano dla pola pod powierzchnią krzywej *ROC*, dosyć wysokie również dla 598 dokładności i precyzji. Widać, że wartości czułości są najsłabiej skorelowane z wydajnością iden- 599 tyfikacji dżetów  $b$ . Jest rzeczą zrozumiałą, że korelacja jest słabsza niż dla precyzji, ponieważ 600 wartości czułości maleją za każdym razem gdy błędnie klasyfikowane są dżety  $b$  stanowiące 601 sygnał, podczas gdy wartości precyzji maleją, gdy błędnie klasyfikowane są przypadki tła. Z 602 tych dwóch błędów, drugi jest bardziej kosztowny, gdyż błędna klasyfikacja nawet niewielkiej 603 części tła znacznie pogarsza czystość otrzymywanej próbki.

604 Na Rys. 13 przedstawiono te same wielkości co na Rys. 12, ale tym razem wybrano tylko 605 25% najwyższych (najlepszych) wartości dla każdej metryki – te punkty są bardziej znaczące, 606 gdyż ostatecznie modele z eksperymentów dających najlepsze wyniki będą używane. Dla takiej 607 selekcji otrzymano zdecydowaną dominację *ROC AUC* – wybór modeli dających najwyższe 608 pole pod powierzchnią krzywej *ROC* zapewnia jednocześnie otrzymanie wysokich wartości wy- 609 dajności identyfikacji dżetów  $b$  dla wybranych punktów pracy.

610 Pole pod powierzchnią krzywej *ROC* zostało wybrane jako główna metryka używana w 611 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.

## 613 4.2 Wyniki dla poszczególnych modeli

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

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

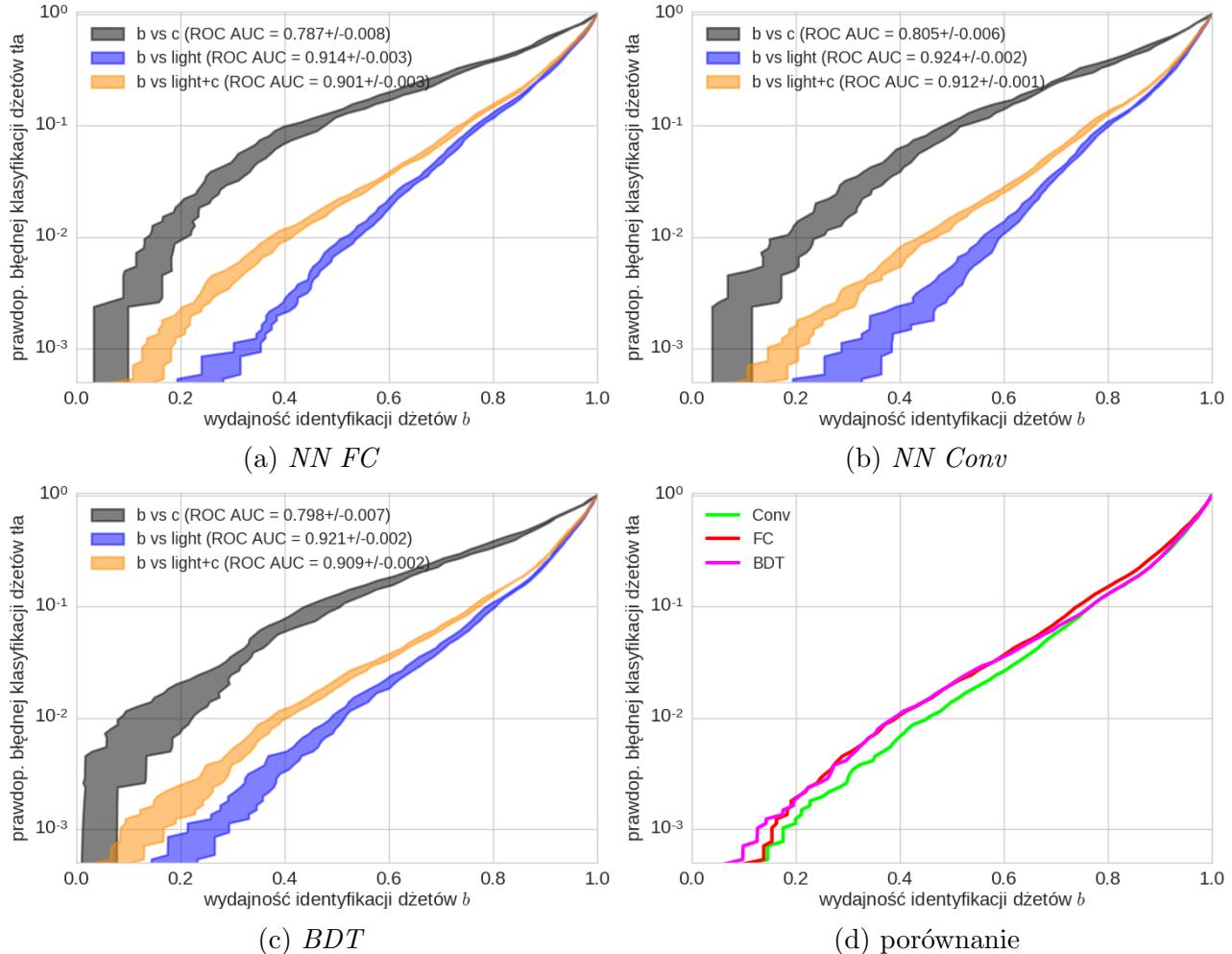
622 Na Rys. 14, 15, 16 przedstawiono odmianę krzywej *ROC* – wykresy przedstawiające zależ-  
623 ności wydajności identyfikacji dżetów  $b$  z niepewnościami (na osi poziomej) dla danego praw-  
624 dopodobieństwa błędnej klasyfikacji dżetów tła (na osi pionowej).

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

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

#### 634 4.2.1 Wyniki dla zmiennych związanych z wtórnymi wierzchołkami

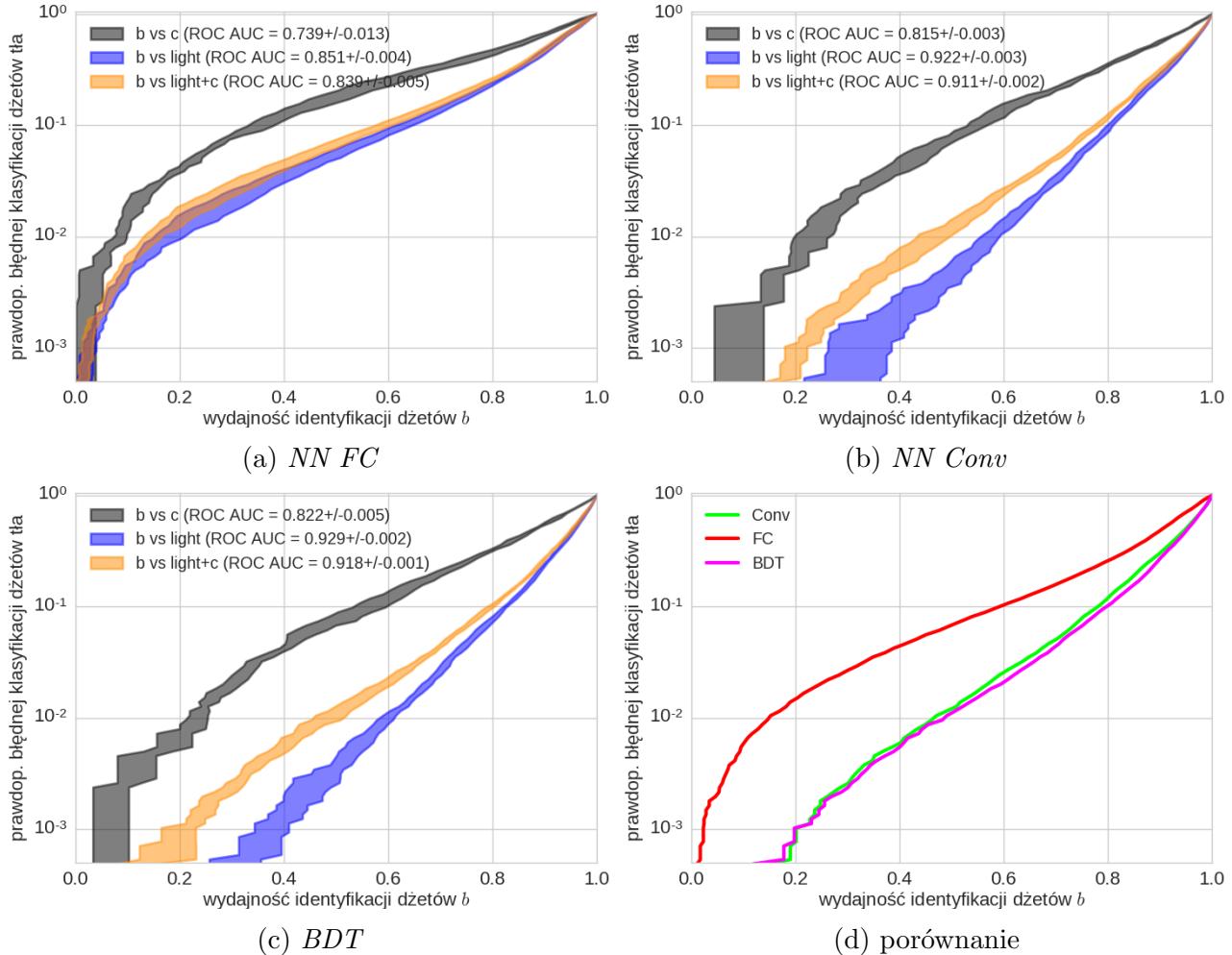
635 Na Rys. 14 przedstawiono rezultaty uzyskane przy trenowaniu na zbiorze danych *SV*. Uzyskano  
 636 bardzo zbliżone wyniki dla wszystkich trzech algorytmów, sieci konwolucyjne były nieznacznie  
 637 lepsze od pozostałych przy wydajnościach na identyfikację dżetów  $b$  poniżej 70%. Dla tych  
 638 samych prawdopodobieństw błędnej klasyfikacji tła uzyskiwały wydajności identyfikacji lepsze  
 639 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*.

#### 640 4.2.2 Wyniki dla zmiennych związanych z częstками tworzącymi dżet

641 Na Rys. 15 przedstawiono rezultaty uzyskane przy trenowaniu na zbiorze danych *constit*. Wyniki  
 642 dla *BDT* oraz *Conv* są niemal identyczne, dla *BDT* trochę lepsze niż w przypadku zestawu da-  
 643nych *SV*, natomiast te uzyskane dla sieci neuronowych typu *FC* są znacznie gorsze. Stosunkowo  
 644 najmniejsze różnice pomiędzy zestawami danych *SV* i *constit* dla algorytmu *FC* otrzymano w  
 645 przypadku separacji dżetów *b* od *c*.

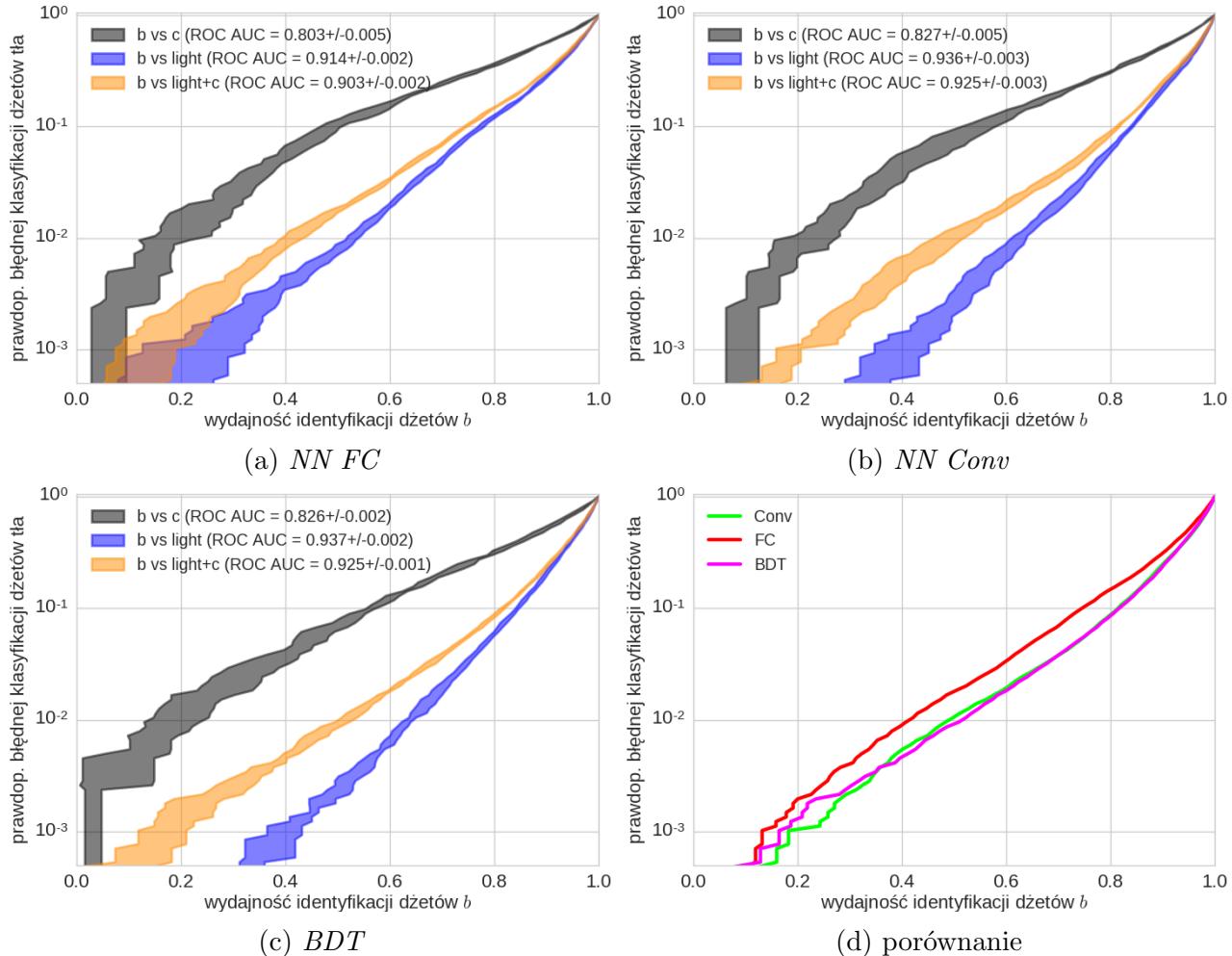


Rysunek 15: Zależności wydajności identyfikacji dżetów *b* od prawdopodobieństwa błędnej klasyfikacji dżetów tła dla poszczególnych algorytmów oraz ich porównanie. Algorytmy wytreno-  
 wane na zmiennych *constit*.

#### 646 4.2.3 Wyniki dla wszystkich zmiennych

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

651 W porównaniu do poprzednich zbiorów danych wszystkie algorytmy poprawiły swoje pre-  
 652 dykce, najmniej *FC*, którego zdolności separacyjne są prawie takie same jak w przypadku *SV*  
 653 (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*.

#### 654 4.2.4 Podsumowanie - wyniki poszczególnych modeli

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

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

661 Algorytm wytrenowany na połączonym zbiorze danych daje trochę lepsze wyniki niż trenowa-  
 662 wany na zbiorach  $SV$  i  $constit$ , co jest oczywiście oczekiwane. Można było natomiast spodziewać  
 663 się większej poprawy, co pokazuje, że predykcje algorytmów trenowanych na  $SV$  oraz  $constit$   
 664 muszą być skorelowane (przy założeniu, że model trenowany na dwóch połączonych zbiorach  
 665 danych daje wyniki niegorsze niż trywialne połączenie dwóch modeli trenowanych na osobnych  
 666 zbiorach danych).

667 W każdym przypadku najgorsze wyniki uzyskano dla sieci w pełni połączonych. Z tabeli  
 668 wynika, że duże znaczenie mają zarówno użyty algorytm jak i zestaw danych. Podobne wyniki  
 669 uzyskiwane dla  $BDT$  oraz  $Conv$  pozwalają przypuszczać, że uzyskiwane przez nie poziom błędu  
 670 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%
$SV-FC$	$0.901 \pm 0.003$	$15 \pm 3$	$39 \pm 2$	$73.7 \pm 0.6$
$SV-Conv$	$0.912 \pm 0.001$	$18 \pm 3$	$45 \pm 2$	$76.4 \pm 0.8$
$SV-BDT$	$0.909 \pm 0.002$	$13 \pm 4$	$38 \pm 1$	$76.3 \pm 0.6$
$constit-FC$	$0.839 \pm 0.005$	$2 \pm 1$	$15 \pm 2$	$58.6 \pm 1.5$
$constit-Conv$	$0.911 \pm 0.002$	$20 \pm 2$	$46 \pm 3$	$77.8 \pm 0.6$
$constit-BDT$	$0.918 \pm 0.001$	$20 \pm 3$	$48 \pm 3$	$79.4 \pm 0.7$
$merged-FC$	$0.903 \pm 0.002$	$13 \pm 6$	$41 \pm 2$	$74.2 \pm 0.3$
$merged-Conv$	$0.925 \pm 0.003$	$18 \pm 2$	$48 \pm 3$	$81.3 \pm 0.4$
$merged-BDT$	$0.925 \pm 0.001$	$16 \pm 5$	$51 \pm 1$	$81.4 \pm 0.4$

Tablica 1: Tabela podsumowująca wyniki uzyskane przez poszczególne modele.

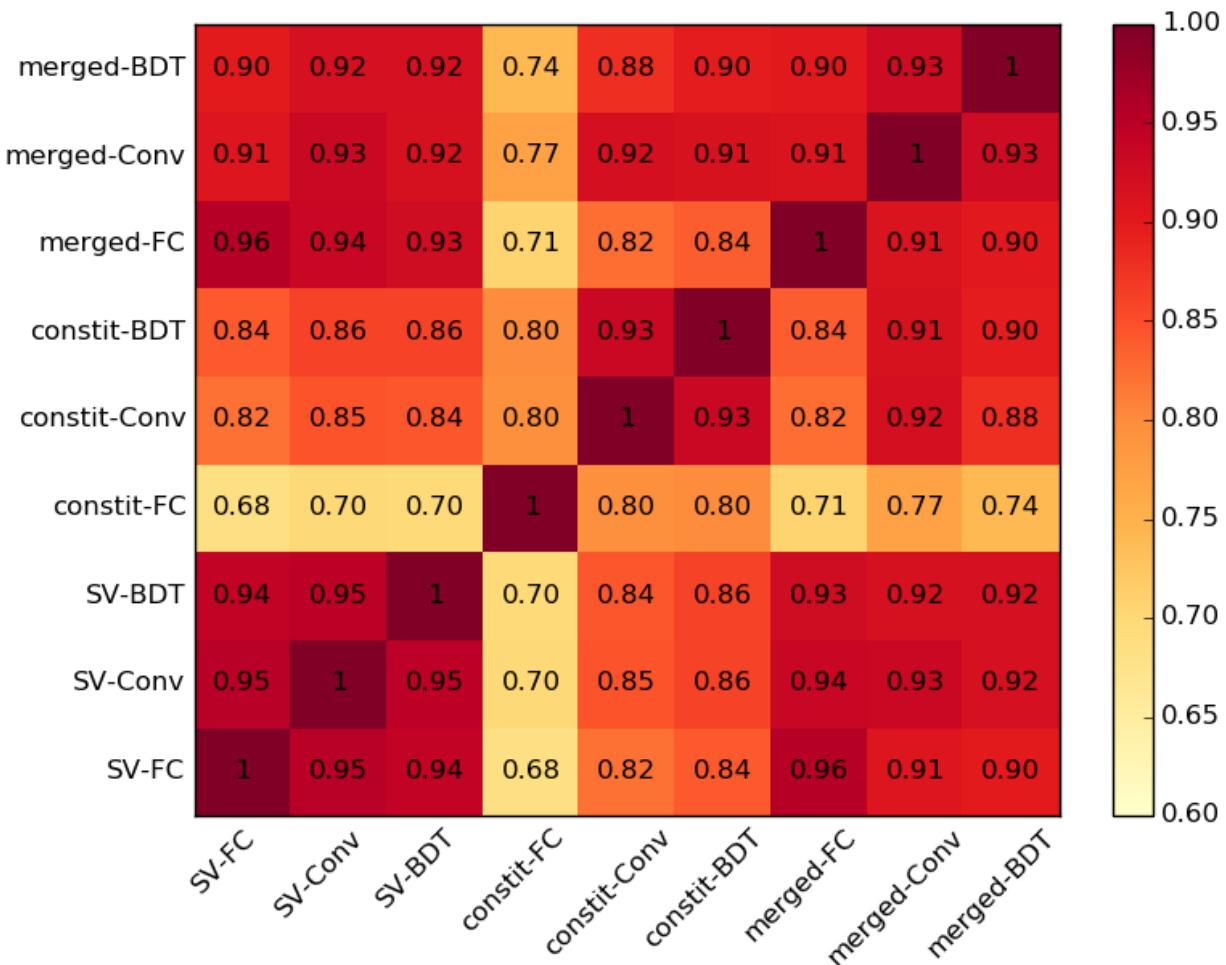
### 671 4.3 Korelacje predykcji modeli

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

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

678 Najmniejszymi korelacjami z innymi wyróżnia się model *constit-FC*, który dawał zdecydowanie najsłabsze wyniki.  
 679

680 Widać wyraźny wzrost korelacji w przypadku modeli trenowanych na tych samych zbiorach  
 681 danych (szczególnie dla modeli trenowanych na zmiennych związanych z wtórnymi wierzchołkami). Modele trenowane na połączonym zbiorze danych są nieco silniej skorelowane z modelami  
 683 *SV-X* niż *constit-X*. Nie obserwuje się natomiast istotnie większych korelacji między modelami  
 684 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.

## 685 4.4 Analiza istotności zmiennych w *BDT*

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

691 Wykorzystano trzy miary istotności zmiennych zaimplementowane w bibliotece **XGBoost**:<sup>4</sup>

- 692 • *weight* informuje ile razy dana zmienna została wykorzystana w podziałach drzew.
- 693 • *total\_gain* jest sumą po spadkach funkcji straty uzyskiwanych dzięki podziałom wykorzy-  
694 stującym daną zmienną
- 695 • *total\_cover* to suma liczb przykładów trenujących, na które miały wpływ podziały wyko-  
696 rzystujące daną zmienną.

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

706 W Tab. 2 przedstawiono wartości miar istotności zmiennych dla 15 najważniejszych cech  
707 (wg *total\_gain*) dla modeli *SV-BDT* oraz *constit-BDT*. Z kolei w Tab. 3 przedstawiono wartości  
708 tych miar pogrupowane wg wielkości fizycznych. Wartości każdej miary zostały unormowane  
709 do 100 (suma dla wszystkich elementów, nie tylko tych przedstawionych w tabelach jest równa  
710 100). System oznaczeń opisany jest w Dodatku B.

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

715 O ile miary *total\_gain* i *total\_cover* są dosyć silnie skorelowane (co widać w obu tabelach)  
716 to *weight*, czyli częstość ich użycia wydaje się być niezależna i przyjmować zbliżone wartości  
717 dla wszystkich zmiennych. Okazuje się, że nie ma zmiennych, które byłyby wyraźnie rzadziej  
718 wykorzystywane od innych.

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

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

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

	<i>weight</i>	trening osobno <i>total_gain</i>	<i>total_cover</i>
$L_{xy}$ - SV0	2.76	13.37	4.68
$L_{xy}$ - SV2	1.88	9.83	3.80
$L_{xy}$ - SV3	1.66	7.68	3.51
$L_{xy}$ - SV1	1.34	6.79	2.84
$L_{xy}$ - SV5	0.77	3.79	1.78
$L_{xy}$ - SV4	1.03	2.79	2.06
$\chi^2/Ndf$ - 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
$\sigma_{vertex}$ - SV0	2.70	1.40	2.08
$\sigma_{Lxy}$ - SV9	0.91	1.36	1.38
$\chi^2/Ndf$ - 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
$\phi$ - 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
$\sigma_{Lxy}$	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
$\sigma_{vertex}$	17.0	8.2	14.2	6.6	6.0	6.9
$\chi^2/Ndf$	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
$\phi$	23.2	8.7	8.2	12.4	8.4	8.2
$\eta$	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*.

## 726 4.5 Analiza wpływu zmiennych na predykcje modeli

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

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

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

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

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

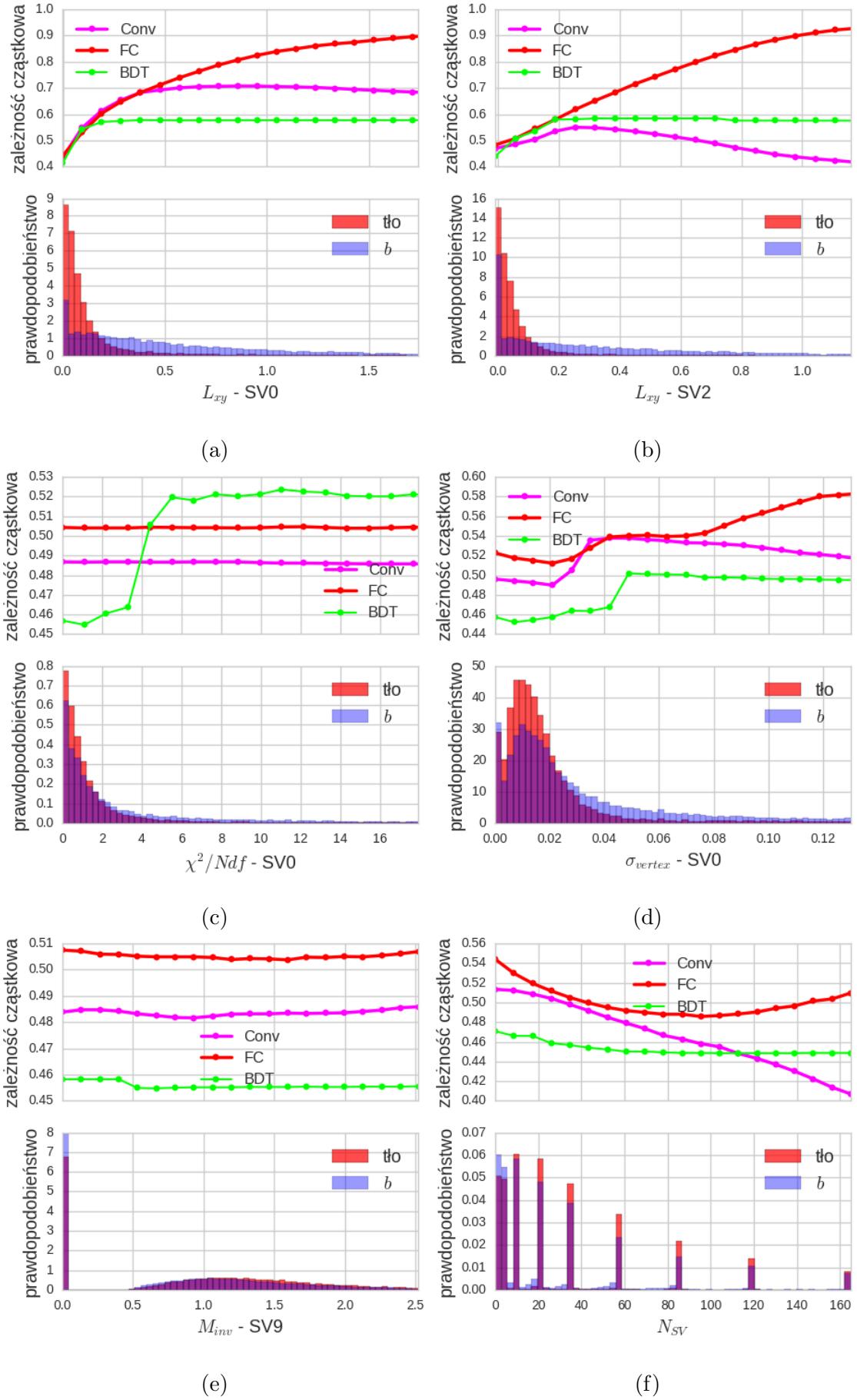
732 Analiza tych pięciu wykresów pozwala na zaobserwowanie ciekawych różnic między algorytmami. Pierwsza z nich: zależności cząstkowe wydają się mieć mniejszą amplitudę w przypadku  $BDT$  niż w przypadku sieci neuronowych. Może wynikać to z kształtu funkcji – w przypadku  $BDT$  jest to złożenie kilku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

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

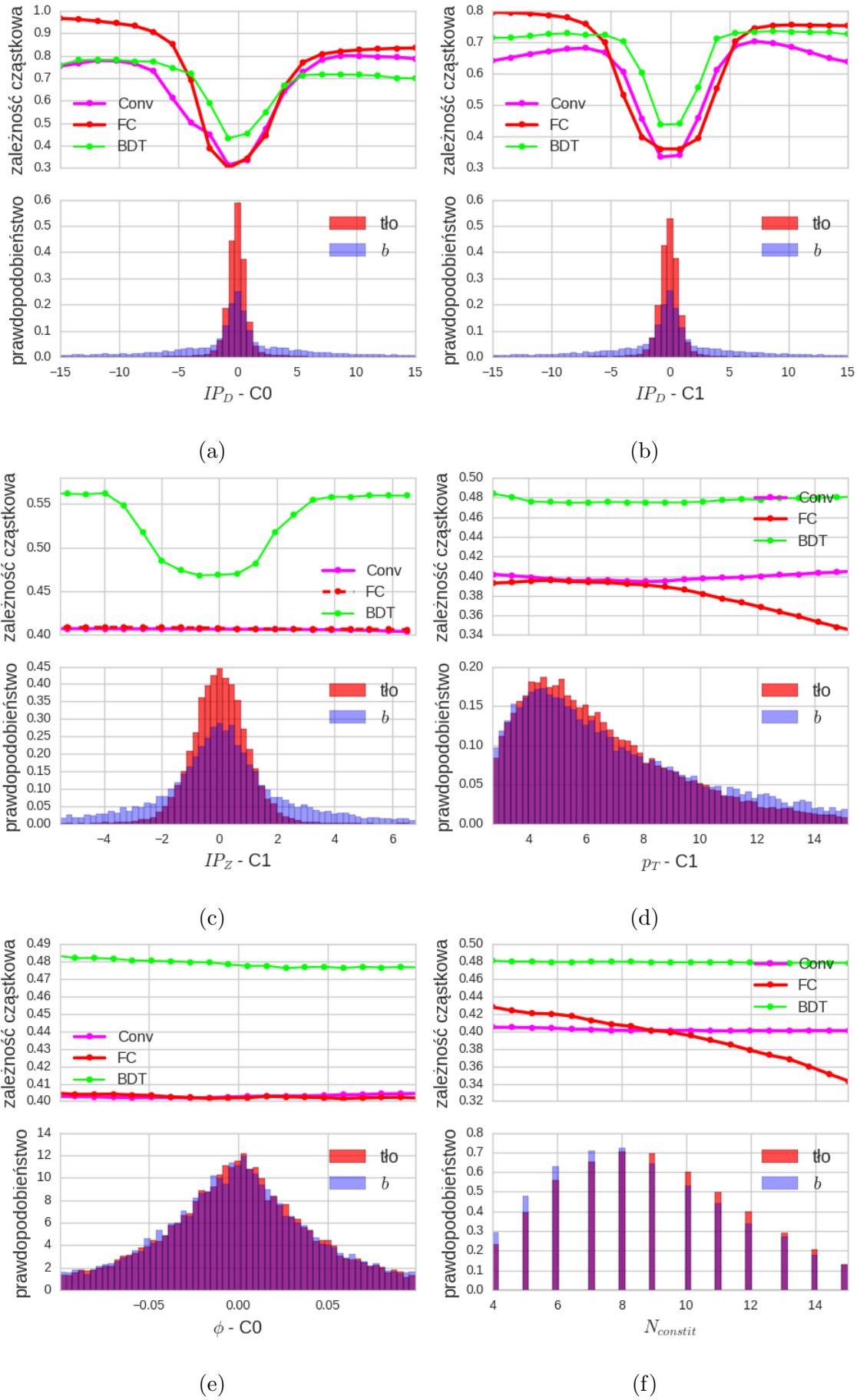
774 przesunięte, z minimum zlokalizowanym wokół wartości -1. Przyczyną takiego zachowania może  
775 być specyficzny sposób trenowania drzew decyzyjnych, które w przypadku takiego rozkładu nie  
776 mogą podzielić przestrzeń fazowej na 3 części w jednym kroku, co byłoby najbardziej korzystne,  
777 ale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

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

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

## 794 5 Podsumowanie i wnioski

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

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

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

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

825 Kolejnym etapem pracy powinna być próba użycia modeli na danych eksperymentalnych,  
826 co wiąże się z szeregiem nowych trudności, głównie związanych z brakiem możliwości bez-  
827 pośredniej oceny wyników uzyskiwanych przez algorytm. Innym kierunkiem rozwoju byłoby  
828 przeprowadzenie podobnej analizy dla dużo trudniejszych danych ze zderzeń ciężkich jonów.

## 829 Bibliografia

- 830 [1] Donald H. Perkins. "Oddziaływanie międzykarkowe i chromodynamika kwantowa". W:  
831 *Wstęp do Fizyki Wysokich Energii*. 2 wydr. PWN, 2005, 171–193.
- 832 [2] David J. Gross i Frank Wilczek. "Ultraviolet Behavior of Nonabelian Gauge Theories".  
833 W: *Phys. Rev. Lett.* 30 (1973). [,271(1973)], s. 1343–1346. DOI: 10.1103/PhysRevLett.  
834 30.1343.
- 835 [3] H. David Politzer. "Reliable Perturbative Results for Strong Interactions?" W: *Phys. Rev.*  
836 *Lett.* 30 (1973). [,274(1973)], s. 1346–1349. DOI: 10.1103/PhysRevLett.30.1346.
- 837 [4] C. Patrignani i in. "Review of Particle Physics". W: *Chin. Phys.* C40.10 (2016), s. 100001.  
838 DOI: 10.1088/1674-1137/40/10/100001.
- 839 [5] John C. Collins i M. J. Perry. "Superdense Matter: Neutrons Or Asymptotically Free  
840 Quarks?" W: *Phys. Rev. Lett.* 34 (1975), s. 1353. DOI: 10.1103/PhysRevLett.34.1353.
- 841 [6] N. Cabibbo i G. Parisi. "Exponential Hadronic Spectrum and Quark Liberation". W:  
842 *Phys. Lett.* 59B (1975), s. 67–69. DOI: 10.1016/0370-2693(75)90158-6.
- 843 [7] D. Boyanovsky, H. J. de Vega i D. J. Schwarz. "Phase transitions in the early and the  
844 present universe". W: *Ann. Rev. Nucl. Part. Sci.* 56 (2006), s. 441–500. DOI: 10.1146/  
845 annurev.nucl.56.080805.140539. arXiv: hep-ph/0602002 [hep-ph].
- 846 [8] Mark G. Alford i Kai Schwenzer. "What the Timing of Millisecond Pulsars Can Teach  
847 us about Their Interior". W: *Phys. Rev. Lett.* 113.25 (2014), s. 251102. DOI: 10.1103/  
848 PhysRevLett.113.251102. arXiv: 1310.3524 [astro-ph.HE].
- 849 [9] Vardan Khachatryan i in. "Evidence for collectivity in pp collisions at the LHC". W:  
850 *Phys. Lett.* B765 (2017), s. 193–220. DOI: 10.1016/j.physletb.2016.12.009. arXiv:  
851 1606.06198 [nucl-ex].
- 852 [10] Jaroslav Adam i in. "Enhanced production of multi-strange hadrons in high-multiplicity  
853 proton-proton collisions". W: *Nature Phys.* 13 (2017), s. 535–539. DOI: 10.1038/nphys4111.  
854 arXiv: 1606.07424 [nucl-ex].
- 855 [11] Matteo Cacciari, Gavin P. Salam i Gregory Soyez. "The Anti-k(t) jet clustering algori-  
856 thm". W: *JHEP* 04 (2008), s. 063. DOI: 10.1088/1126-6708/2008/04/063. arXiv:  
857 0802.1189 [hep-ph].
- 858 [12] Vardan Khachatryan i in. "Charged-particle nuclear modification factors in PbPb and  
859 pPb collisions at  $\sqrt{s_{NN}} = 5.02$  TeV". W: *JHEP* 04 (2017), s. 039. DOI: 10.1007/  
860 JHEP04(2017)039. arXiv: 1611.01664 [nucl-ex].
- 861 [13] Betty Abelev i in. "Centrality Dependence of Charged Particle Production at Large Trans-  
862 verse Momentum in Pb–Pb Collisions at  $\sqrt{s_{NN}} = 2.76$  TeV". W: *Phys. Lett.* B720 (2013),  
863 s. 52–62. DOI: 10.1016/j.physletb.2013.01.051. arXiv: 1208.2711 [hep-ex].
- 864 [14] Carlos A. Salgado i Urs Achim Wiedemann. "Calculating quenching weights". W: *Phys.*  
865 *Rev.* D68 (2003), s. 014008. DOI: 10.1103/PhysRevD.68.014008. arXiv: hep-ph/0302184  
866 [hep-ph].
- 867 [15] Yuri L. Dokshitzer i D. E. Kharzeev. "Heavy quark colorimetry of QCD matter". W:  
868 *Phys. Lett.* B519 (2001), s. 199–206. DOI: 10.1016/S0370-2693(01)01130-3. arXiv:  
869 hep-ph/0106202 [hep-ph].
- 870 [16] Georges Aad i in. "Performance of *b*-Jet Identification in the ATLAS Experiment". W:  
871 *JINST* 11.04 (2016), P04008. DOI: 10.1088/1748-0221/11/04/P04008. arXiv: 1512.  
872 01094 [hep-ex].

- 873 [17] M. Aaboud i in. “Measurements of b-jet tagging efficiency with the ATLAS detector using  
 874  $t\bar{t}$  events at  $\sqrt{s} = 13$  TeV”. W: *JHEP* 08 (2018), s. 089. DOI: 10.1007/JHEP08(2018)089.  
 875 arXiv: 1805.01845 [hep-ex].
- 876 [18] Serguei Chatrchyan i in. “Identification of b-quark jets with the CMS experiment”. W:  
 877 *JINST* 8 (2013), P04013. DOI: 10.1088/1748-0221/8/04/P04013. arXiv: 1211.4462  
 878 [hep-ex].
- 879 [19] A. M. Sirunyan i in. “Identification of heavy-flavour jets with the CMS detector in pp  
 880 collisions at 13 TeV”. W: *JINST* 13.05 (2018), P05011. DOI: 10.1088/1748-0221/13/  
 881 05/P05011. arXiv: 1712.07158 [physics.ins-det].
- 882 [20] Linus Feldkamp. “Study of b-jet tagging performance in ALICE”. W: *J. Phys. Conf. Ser.*  
 883 509 (2014), s. 012061. DOI: 10.1088/1742-6596/509/1/012061. arXiv: 1310.2817  
 884 [hep-ex].
- 885 [21] Rüdiger Haake. “Machine and deep learning techniques in heavy-ion collisions with ALICE”.  
 886 W: *Proceedings, 2017 European Physical Society Conference on High Energy Physics*  
 887 (*EPS-HEP 2017*): *Venice, Italy, July 5-12, 2017*. T. EPS-HEP2017. 2017. DOI: 10.22323/  
 888 1.314.0498. arXiv: 1709.08497 [physics.data-an]. URL: <https://pos.sissa.it/314/498/pdf>.
- 890 [22] Roel Aaij i in. “Identification of beauty and charm quark jets at LHCb”. W: *JINST* 10.06  
 891 (2015), P06013. DOI: 10.1088/1748-0221/10/06/P06013. arXiv: 1504.07670 [hep-ex].
- 892 [23] K. Aamodt i in. “The ALICE experiment at the CERN LHC”. W: *JINST* 3 (2008),  
 893 S08002. DOI: 10.1088/1748-0221/3/08/S08002.
- 894 [24] Betty Bezverkhny Abelev i in. “Performance of the ALICE Experiment at the CERN  
 895 LHC”. W: *Int. J. Mod. Phys.* A29 (2014), s. 1430044. DOI: 10.1142/S0217751X14300440.  
 896 arXiv: 1402.4476 [nucl-ex].
- 897 [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).
- 900 [26] Sotiris B Kotsiantis, I Zaharakis i P Pintelas. “Supervised machine learning: A review  
 901 of classification techniques”. W: *Emerging artificial intelligence applications in computer*  
 902 *engineering* 160 (2007), s. 3–24.
- 903 [27] Marcin Wolter. “Metody analizy wielu zmiennych w fizyce wysokich energii”. Prac. dokt.  
 904 IFJ PAN, 2012.
- 905 [28] Leo Breiman. “Bagging predictors”. W: *Machine learning* 24.2 (1996), s. 123–140.
- 906 [29] Yoav Freund i Robert E Schapire. “A decision-theoretic generalization of on-line learning  
 907 and an application to boosting”. W: *Journal of computer and system sciences* 55.1 (1997),  
 908 s. 119–139.
- 909 [30] Tianqi Chen i Carlos Guestrin. “XGBoost: A Scalable Tree Boosting System”. W: *Proce-*  
 910 *edings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and*  
 911 *Data Mining*. KDD ’16. San Francisco, California, USA: ACM, 2016, s. 785–794. ISBN:  
 912 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: <http://doi.acm.org/10.1145/2939672.2939785>.
- 913 [31] James Bergstra i Yoshua Bengio. “Random Search for Hyper-parameter Optimization”.  
 914 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/](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](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](https://www-d0.fnal.gov/Run2Physics/top/singletop_observation/singletop_observation_updated.html) (term. wiz. 15.07.2018).

- 962 [50] Matteo Cacciari i Gavin P. Salam. “Pileup subtraction using jet areas”. W: *Phys. Lett.*  
963 B659 (2008), s. 119–126. DOI: 10.1016/j.physletb.2007.09.077. arXiv: 0707.1378  
964 [[hep-ph](#)].
- 965 [51] Andrew P Bradley. “The use of the area under the ROC curve in the evaluation of  
966 machine learning algorithms”. W: *Pattern recognition* 30.7 (1997), s. 1145–1159.
- 967 [52] Jason Gallicchio i Matthew D. Schwartz. “Seeing in Color: Jet Superstructure”. W: *Phys.*  
968 *Rev. Lett.* 105 (2010), s. 022001. DOI: 10.1103/PhysRevLett.105.022001. arXiv: 1001.  
969 5027 [[hep-ph](#)].

## 970 Dodatek A Metryki

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

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

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

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

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

977

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.

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

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

## 992 Dodatek B Skróty i oznaczenia

993 W pracy używane są następujące skróty i oznaczenia:

- 994 • algorytmy:

995  $FC$  – sieci neuronowe w pełni połączone,

996  $Conv$  – sieci neuronowe konwolucyjne,

997  $BDT$  - wzmacniane drzewa decyzyjne

- 998 • zbiory danych:

999  $SV$  – zestaw zawierający tylko zmienne związane z wtórnymi wierzchołkami,

1000  $constit$  – zestaw zawierający tylko zmienne związane z częstками tworzącymi dżet,

1001  $merged$  – zestaw zawierający wszystkie zmienne (tj.  $SV + Constit +$  zmienne charaktery-  
1002 zujące dżet jako całość)

- 1003 • modele (zestaw danych + algorytm): nazywane są wg wzoru:

1004 (oznaczenie\_zbioru\_danych)-(oznaczenie\_algorytmu),

1005 np.  $SV\text{-}Conv$  oznacza konwolucyjną sieć neuronową wytrenowaną na zmiennych zwi-  
1006 zanych z wtórnymi wierzchołkami a  $merged\text{-}BDT$  - wzmacniane drzewa decyzyjne, do  
1007 treningu których użyte zostały wszystkie zmienne. Zapis  $merged\text{-}X$  stanowi zbiorcze ozna-  
1008 czenie modeli  $merged\text{-}SV$ ,  $merged\text{-}Conv$  i  $merged\text{-}BDT$ .

- 1009 • poszczególne zmienne (kolumny w zbiorze danych) oznaczane są według wzoru:

1010 (nazwa\_zmiennej) – (numer\_obiektu),

1011 np.  $\sigma_{L_{xy}}$  –  $SV2$  oznacza niepewność wyznaczenia  $L_{xy}$  dla wtórnego wierzchołka nr 2, a  
1012  $IP_Z\text{-}C5$  – parametr zderzenia wzdłuż osi wiązki cząstki nr 5 – numery na posortowanych  
1013 listach (Rozdz. 3, gdzie znajdują się także opisy wielkości fizycznych).