ZPS: Estymowanie energii cząstek w kalorymetrii z użyciem sieci neuronowych

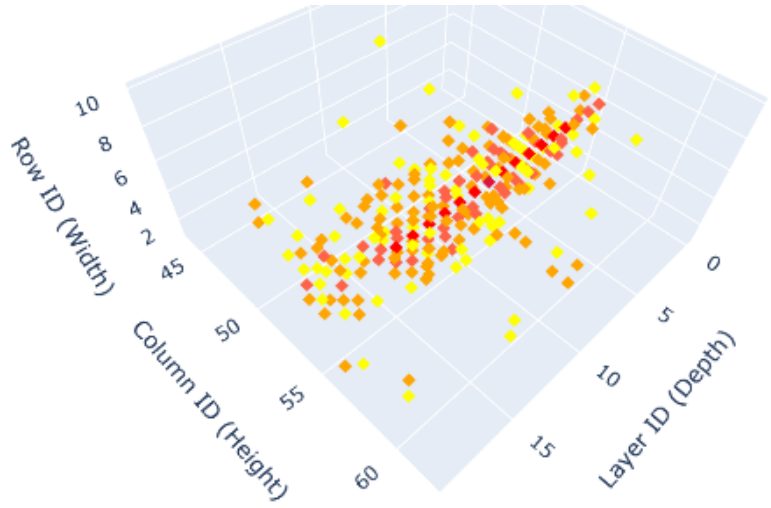
Patrycja Maciejewska, Piotr Marczak,
Stanisław Kiedrzyński, Przemysław Kaleta

Semestr letni 2025

1 Temat projektu

W eksperymentach z dziedziny fizyki cząstek elementarnych, takich jak te prowadzone w LHC, niezbędne jest precyzyjne określenie parametrów cząstek — ich energii, pozycji oraz kierunku ruchu. Jednym z kluczowych elementów aparatury detekcyjnej jest kalorymetr — urządzenie zaprojektowane do pochłaniania energii cząstek i mierzenia jej z wysoką dokładnością. Dzięki temu możliwa jest szczegółowa analiza zjawisk zachodzących w zderzeniach cząstek, identyfikacja typów cząstek oraz pomiar ich energii — co jest absolutnie fundamentalne dla testowania modeli teoretycznych i poszukiwania nowych zjawisk fizycznych.

W ramach eksperymentu LUXE wykonywany ma być pomiar energii pozytonów. Kalorymetr ma strukturę warstwową i składa się z absorbujących płyt wolframowych oraz cienkich warstw detektorów krzemowych. Warstwy absorbera powodują, że cząstki ulegają serii oddziaływań, tworząc kaskadę wtórnych cząstek, której intensywność i rozkład geometryczny można następnie zmierzyć za pomocą detektorów. Przykład takiej kaskady przedstawiono na Rys. 1.



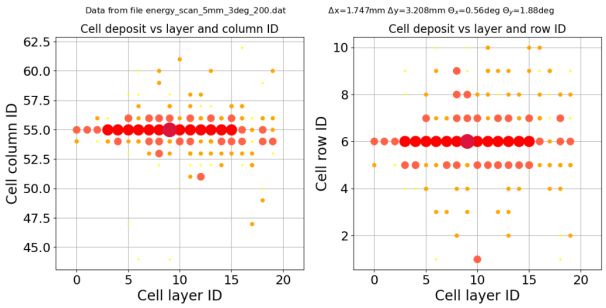
Rysunek 1: Przykład kaskady z rozważanych danych.

Rozważany przez nas kalorymetr składa się z 20 warstw, z których każda zawiera detektor segmentowany na komórki pomiarowe o wymiarach 5 mm × 5 mm. W jednej warstwie znajduje się siatka 11 × 110 takich komórek. Dane używane w projekcie zostały wygenerowane dla opisywanego kalorymetru w zaawansowanych symulacjach z użyciem programu *GEANT4*. Przykład rozkładu depozytu dla arbitralnej kaskady z używanych przedstawiono na Rys.2. Zbiór danych zawierał przypadki rozpadów cząstek o określonych energiach, podzielone na dwa zbiory.

Pierwszy zbiór obejmował przypadki o energiach między 3, a 20 GeV, z krokiem co 1 GeV. Drugi zbiór zawierał przypadki o energiach między 2.5, a 15 GeV, z krokiem co 2.5 GeV.

Ponieważ energia w kalorymetrii jest wielkością ciągłą, ograniczenie się do dyskretnych wartości może prowadzić do przeuczenia modelu na konkretne wartości energii. Aby zminimalizować to ryzyko, do treningu wykorzystano jedynie podzbiór danych z pierwszego zbioru. Walidację modelu przeprowadzono na przykładach z drugiego zbioru, natomiast testowanie odbywało się na przypadkach pochodzących z obu zbiorów.

Ewentualne różnice w wynikach modelu między danymi z dwóch zbiorów testowych mogą służyć jako wskaźnik przeuczenia modelu do specyficznych wartości energii w zbiorze treningowym.

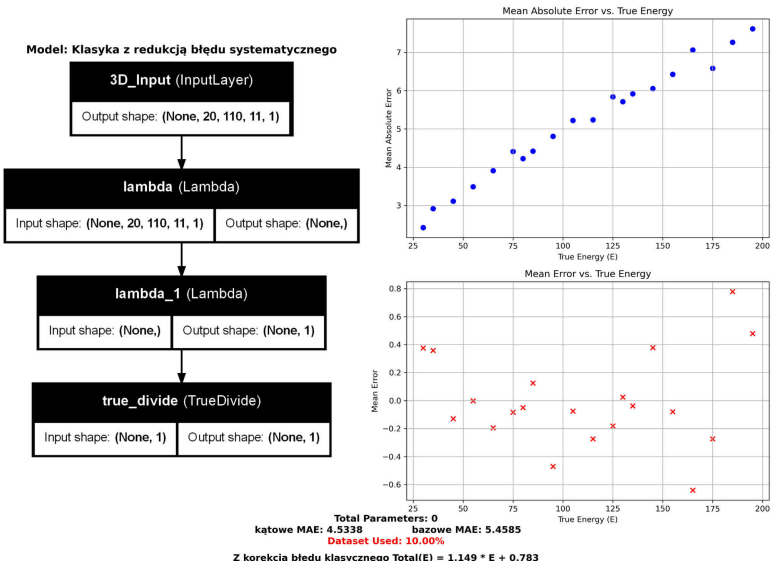


Rysunek 2: Przykład rozkładów depozytów (przecięcia w płaszczyznach XZ i YZ)

Celem projektu jest opracowanie modelu opartego na sieci neuronowej, który umożliwi precyzyjniejszą rekonstrukcję kluczowych parametrów wiązki cząstek — jej energii, pozycji oraz kierunku — niż w przypadku klasycznych metod analizy. Dzięki możliwościom uczenia reprezentacji złożonych wzorców w danych, sieć neuronowa ma szansę lepiej uchwycić nieliniowe zależności pomiędzy sygnałami rejestrowanymi w detektorze a rzeczywistymi właściwościami wiązki. Oczekuje się, że takie podejście pozwoli zredukować wpływ szumów i fluktuacji charakterystycznych dla 'klasycznych' metod, prowadząc do dokładniejszych estymacji.

2 Metoda Klasyczna

Istnieją "klasyczne" metody rekonstrukcji energii, pozycji i kierunku cząstek na podstawie rozkładu depozytów energii w kalorymetrze. W przypadku pomiaru energii najprostszym podejściem jest suma wszystkich zarejestrowanych depozytów energii. Metoda ta jest jednak wrażliwa na fluktuacje związane z procesem jonizacji w ośrodku, zwłaszcza na charakterystyczny, wydłużony ogon rozkładu Landaua. Z tego powodu bardziej zaawansowane podejścia, takie jak nieliniowe kombinacje depozytów, mogą dostarczać dokładniejszych estymacji energii, lepiej uwzględniając właściwości statystyczne sygnału. Dysponując symulacjami interakcji cząstek z kalorymetrem, możemy testować różne metody rekonstrukcji i oceniać ich dokładność. Wykorzystując zestaw danych symulowanych dla wiązki prostopadłej o energiach z zakresu od 3 do 20 GeV, na rysunku 4 przedstawiono wyniki klasycznego podejścia. Pokazano tam średni błąd (ME) oraz średni błąd bezwzględny (MAE) w funkcji energii wiązki.



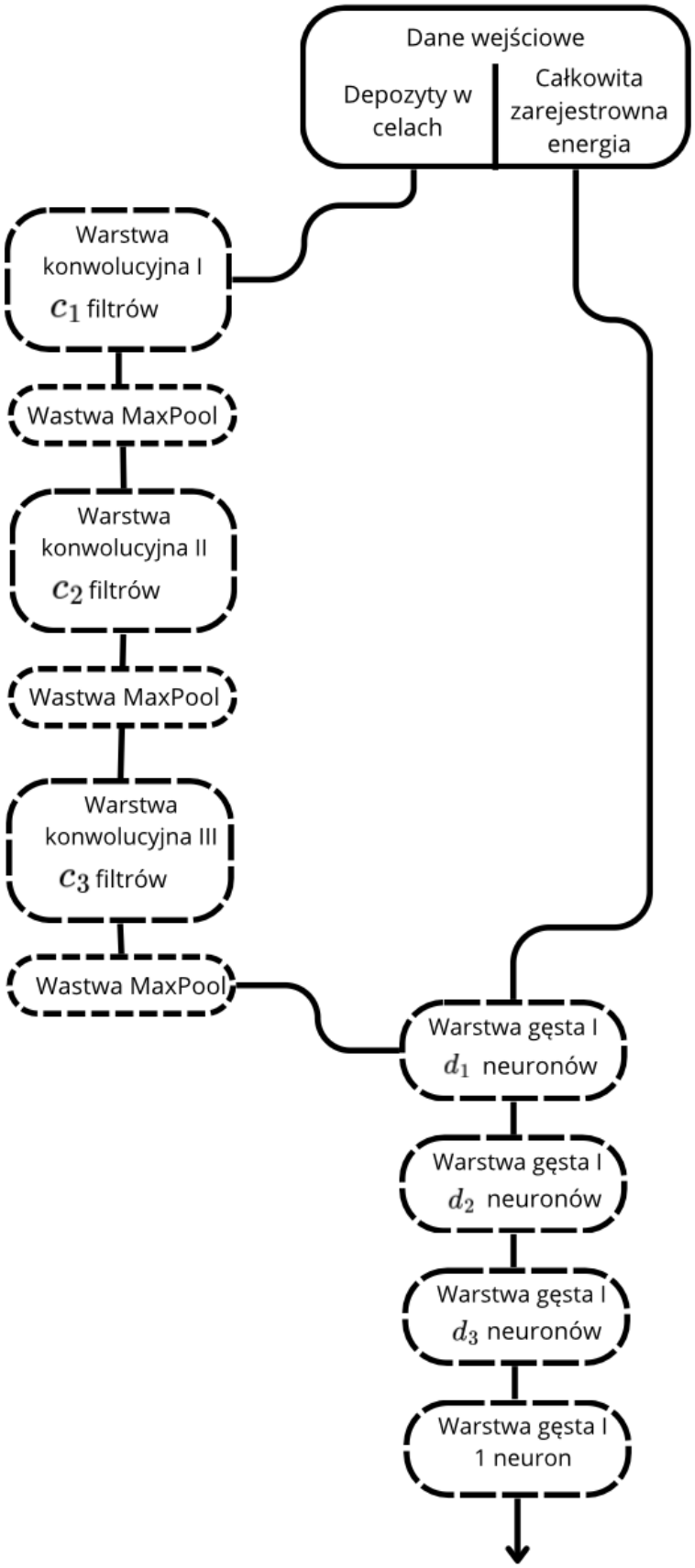
Rysunek 3: MAE i MSE klasycznego zliczania wraz z schematem jego liczenia w środowisku *tensorflow*.

3 Środowisko informatyczne

Do projektowania i testowania modeli używano środowiska *tensorflow*. Wybrano je ze względu na przyjazność dla nowych użytkowników i bogactwo gotowych rozwiązań. Minusem tego środowiska jest trudność tzw. *customizacji* modeli w niestandardowy sposób choć problemy te udało się obejść np. przy implementacji niestandardowych funkcji straty.
Całość napisanego i używanego kodu znaleźć można w repozytorium na platformie github.
Link do github: https://github.com/przemekk125/projekt_studencki.
Sieci szkolono początkowo na własnym sprzęcie, a następnie dzięki uprzejmości prof. Kalinowskiego na maszynach z ICM UW.

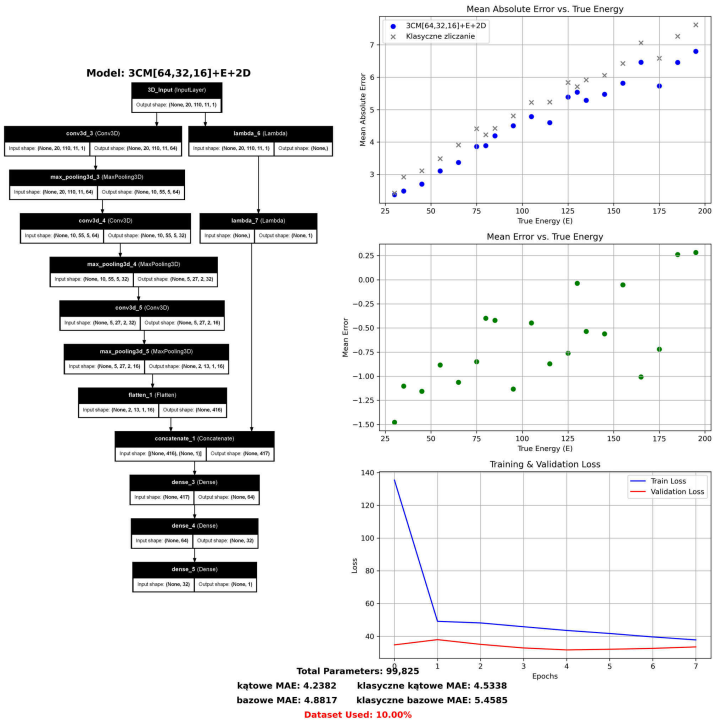
4 Architektura

Problem klasyfikacji cząstek w kalorymetrach stanowi istotne wyzwanie we współczesnej fizyce eksperymentalnej. Z tego względu w literaturze naukowej można znaleźć liczne prace wykorzystujące sieci neuronowe do tego typu zadań. Inspiracją dla naszego podejścia w tym projekcie był artykuł On the Use of Neural Networks for Energy Reconstruction in High-granularity Calorimeters [1]. W przytoczonym artykule analizowano kalorymetr o innej geometrii i przeznaczeniu, dlatego zastosowane tam rozwiązania należało dostosować do naszych realiów.
Idea modelu, którego schemat przedstawiono na Rys.4, jest prosta. Klasyczna metoda zliczania energii sprawdza się całkiem dobrze, jednak nie wykorzystuje informacji o geometrii kaskady. Dlatego chcielibyśmy wzbogacić ją o interpretację geometryczną — do czego świetnie nadają się sieci konwolucyjne (w polskiej literaturze znane również jako splotowe). Standardowym zadaniem sieci neuronowych jest analiza dwuwymiarowych, kolorowych obrazów. Ich działanie można jednak uogólnić na obrazy o wyższych wymiarach. W naszym przypadku geometria kaskady jest zapisana w formie trójwymiarowego „obrazka", w którym rolę „koloru" pełnią lokalne depozyty energii.
Właśnie te idee realizuje przedstawiony poniżej model. Lewa gałąź modelu zawierająca warstwy konwolucyjne odpowiada za interpretacja geometryczną. Prawa gałąź dostarcza nam podawaną jaką część danych wejściowych zsumowaną energię wszystkich depozytów. Zaimplementowane w drugiej części modelu warstwy gęste mają w tym scenariuszu zarówno informacje o całej zarejestrowanej energii, jak i o geometrii wiązki. Założenie jest więc takie, że model będzie zapewniał 'poprawkę geometryczną' do 'klasycznego' zliczania depozytów energetycznych.

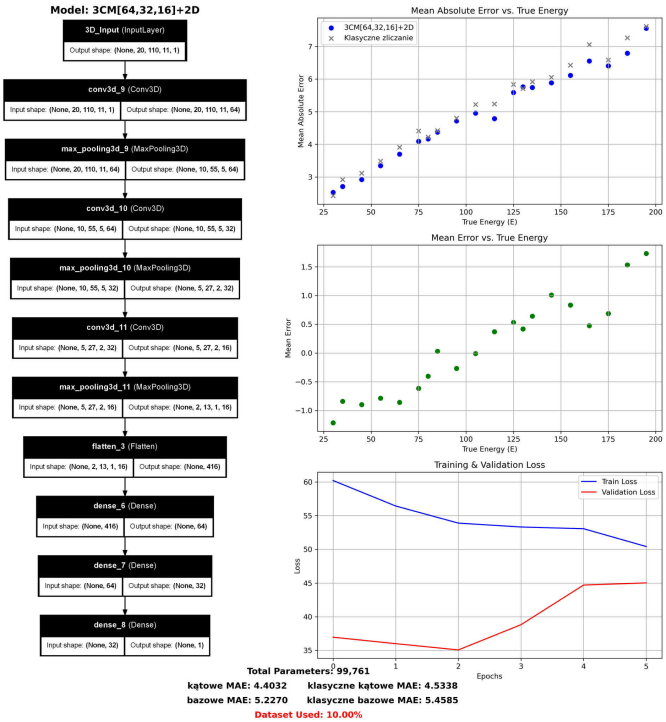


Rysunek 4: Szkic szkieletu używanego modelu. W warstwach konwolucyjnych i trzech pierwszych gęstych używano funkcji sigmoidalnej jako funkcji aktywacji. W ostatniej warstwie gęstej zastosowano aktywacje liniową.

Pierwszym krokiem było przetestowanie przyjętej architektury. W pierwotnych testach przeprowadzono testy różnych architektur mn. sieci w pełni gęstej, sieci opierającej się tylko na geometrii (a więc bez informacji o całkowitej zliczonej energii), czy sieci z różnymi konfiguracjami warstw kownolucyjnych i gęstych. Przykładowe wyniki takich testów przedstawiono na Rys.5,6. Wśród przetestowanych modeli najlepiej sprawował się ten zaprezentowany na Rys. 4.



Rysunek 5: Infografika o jednym z pierwszych testowanych modeli



Rysunek 6: Infografika o jednym z pierwszych testowanych modeli, model bez informacji o całkowitej zliczonej energii.

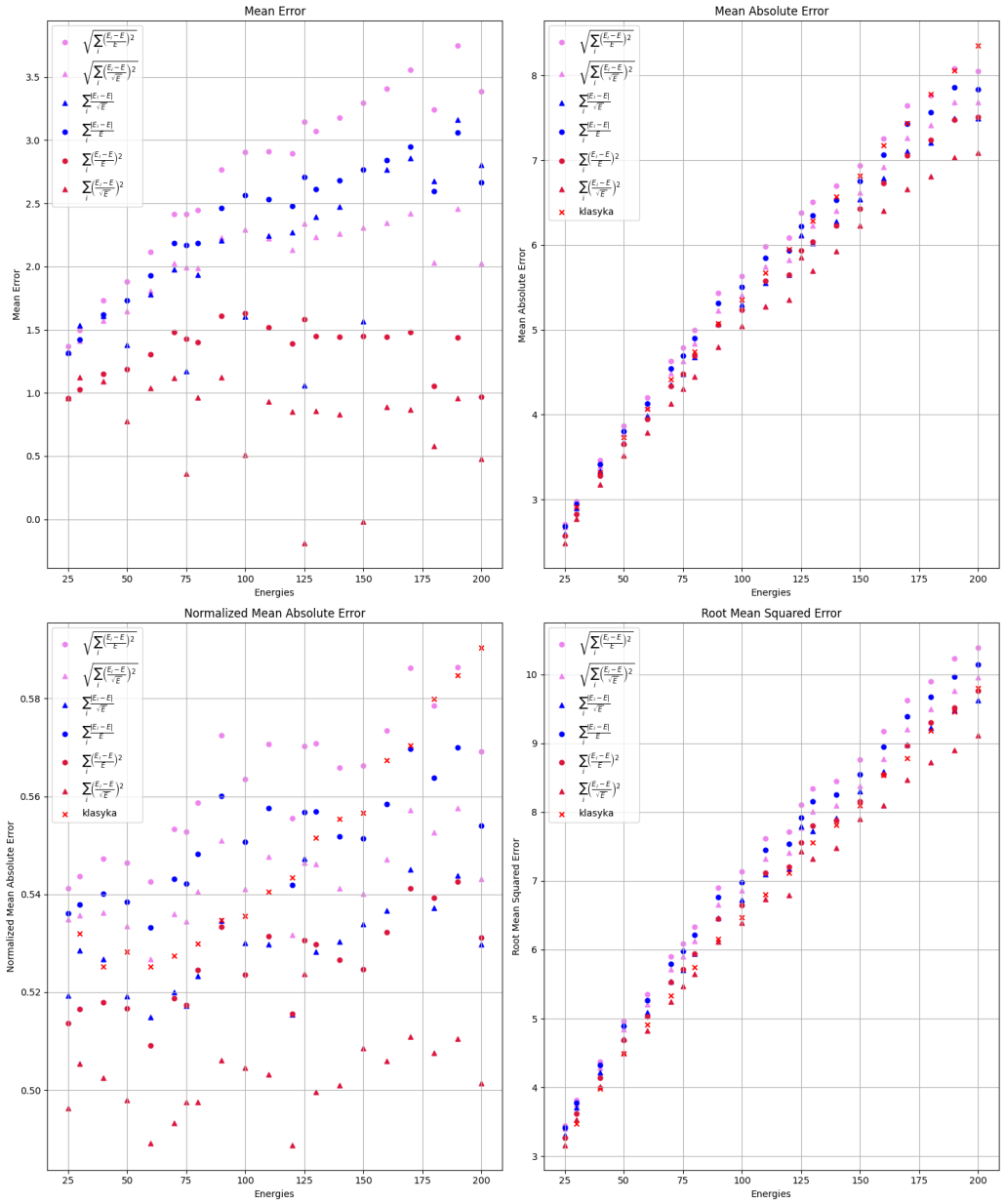
Po ustaleniu surowej architektury modelu zajęto się dopasowaniem optymalnej liczb filtrów c_i i neuronów d_j. W tym celu przebadano dwa modele (na githubie zapisane jako *Chonkier_model.py* i *Smol_model.py*). Okazało się, że model 'większy' tj. o większej ilości filtrów i neuronów bardzo szybko się przeuczał. W ten sposób zdecydowano się na model z następującymi wartościami:

$$c_1 = 32, c_2 = 32, c_3 = 16, d_1 = 64, d_2 = 32, d_3 = 16 \tag{1}$$

Modele tego typu wciąż miały problemy z przeuczaniem, ale mniejsze niż sieci bardziej skomplikowane. Równocześnie działały efektywniej niż modele o mniejszej ilości parametrów. Stąd (1) stosowano we wszystkich szkolonych następnie modelach.

5 Funkcje straty

Kolejną kwestią było dobranie odpowiedniej funkcji straty. Początkowo używano standardowych w uczeniu maszynowym MAE i MSE (tj. średniego błędu bezwględnego i średniego błędu kwadratowego). Funkcje takie nie oddają jednak dobrze fizycznych realiów problemu. Dodatkowo duży rozrzut w wartościach energii (naturalnie też w niepewnościach) powodował przeuczanie się modelu dookoła wyższych energii. W tym celu przetestowano szereg różnych funkcji straty. Oprócz funkcji przedstawionych w legendzie Rys.7 przestowano też tzw. stratę Hubera lecz bez dobrych efektów. Finalne testy rozważanych funkcji przedstawiono na 7. Wykresy poszczególnych modeli można znaleźć na githubie w folderze *losses* (który znajduje się w folderze *results*). Większość funkcji napisano *customowo*, kod źródłowy można znaleźć w pliku *WayTooDeep.py* w folderze *Old*.



Rysunek 7: Porównanie różnych rozpatrywanych funkcji straty

Zdecydowanie najlepiej radził sobie model z funkcją straty:

$$Loss = \sum_i \left(\frac{E^i_{true} - E^i_{pred}}{\sqrt{E^i_{true}}} \right)^2 \tag{2}$$

co odpowiada fizycznym intuicjom. Najlepszą funkcją straty okazała się więc taka o realistycznym, fizycznym sensie co pokazuje, że połączenie metod uczenia maszynowego z fizyczną intuicją na temat problemu jest w stanie dawać bardzo dobre efekty.
Warto zauważyć jak zachowuje się błąd średni (na wykresie: *mean error*). Obserwujemy tutaj względnie stałe oscylacje tego błędu dookoła mniej więcej 1. Model ma więc w miarę stały błąd systematyczny. Model szkolony z funkcją straty (2) ma ten błąd systematyczny najmniejszy, lecz jest on wciąż znaczny zwłaszcza dla niskich energii. Wyniki modelu można by więc poprawiać o błąd systematyczny obliczony na podstawie statystyki wyników na zbiorze testowym co powinno jeszcze poprawić wyniki zwłaszcza w zakresie małych energii.

6 Mechanizmy zapobiegające przeuczaniu i poprawa modelu

W ramach projektu sprawdzono dwa mechanizmy, których zadaniem jest zapobieganie przeuczaniu modelu: regularyzację [2] i dropout [3]]. W ramach sprawdzenia regularyzacji, czyli techniki penalizującej zbyt duże co do wartości wagi modelu, wytrenowano pięć modeli.

	warstwy konwolucyjne	warstwy gęste
model 1	$L_2(10^{-4})$	$L_1(10^{-5})L_2(10^{-3})$
model 2	$L_2(10^{-4})$	$L_1(10^{-3})L_2(10^{-2})$
model 3	$L_2(10^{-3})$	$L_1(10^{-5})L_2(10^{-3})$
model 4	$L_2(10^{-4})$	$L_2(10^{-3})$
model 5	$L_2(10^{-4})$	$L_1(10^{-4})$

Tabela 1: Zestawienie modelów mających na celu sprawdzenie działania regularyzacji. W tabeli wpisany jest typ regularyzacji (L_1 lub L_2), wraz z wpisaną siłą regularyzacji

Najbardziej obiecującą kombinacją zastosowaną w ostatecznym modelu była kombinacja dla modelu o numerze 2. Zastosowanie regularyzacji wydłużyło czas treningu o 1-4 epoki, oraz poprawiło ogólny wynik modelu.

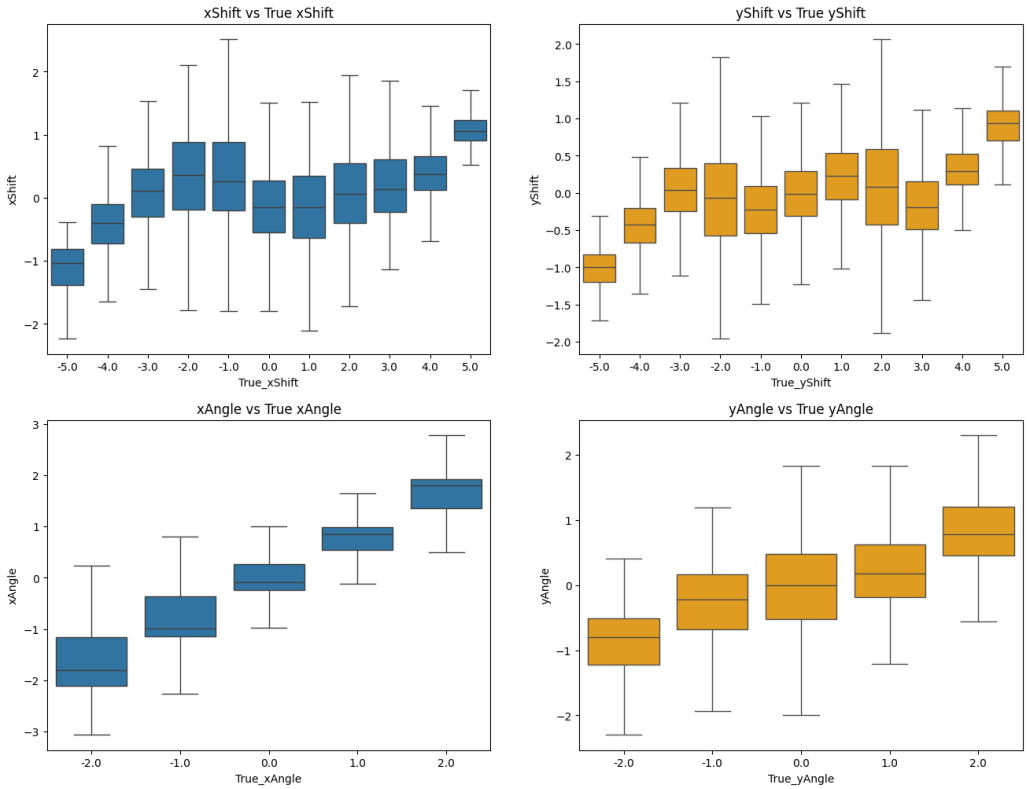
Technika *Dropout* polegająca na losowym wyłączaniu neuronów podczas treningu została przetestowana na każdej wartstwie modelu po kolei, w prawdopodobieństwem wyłączenia neuronu równym 20%. W ramach tej procedury zadecydowano, że zastosowanie tej techniki dla jedynie trzeciej warstwy konwolucyjnej w znaczący sposób poprawia zachowanie modelu, natomiast zmiany prawdopodobieństwa dezaktywacji neuronu bardzo zaburzały działanie tego modelu. Z tych powodów ta technika nie została użyta w ostatecznym modelu.

W celu poprawienia stabilności uczenia i przyspieszenia treningu sprawdzono działanie warstw służących do normalizacji podczas treningu aktualnego *batcha*, nazywanej *BatchNorm* [3]. Warstwa ta używana była po każdej warstwie konwolucyjnej, przed jej funkcją aktywacji. Wydaje się, że ten mechanizm poprawia wyniki, aczkolwiek ze względu na brak porównania modeli identycznych bez oraz z warstwami *BatchNorm* nie jest to potwierdzone wytrenowanymi modelami.

7 Model kątowo-położeniowy

Kolejnym zadaniem z jakim mogłyby się zmierzyć sieci neuronowe jest wyznaczenie miejsca i kąta padania cząstki do kalorymetru. Rozwiązanie tego problemu w sposób klasyczny napotyka jeden główny problem: Największa część energii cząstki, a co za tym idzie, powstałej w jej wyniku kaskady jest zdeponowana kilka warstw w głąb kalorymetru. Interpolacja środków ciężkości depozytów w warstwach generuje duże niepewności położenia w pierwszej warstwie, rzędu wielkości wartości rzeczywistych.

W tym celu zmodyfikowano najlepiej działający model tak, aby na jego wyjściu otrzymać cztery liczby: współrzędne padania (x,y) oraz rzuty kąta padania na płaszczyznę XY (xAngle, yAngle). Prawa gałąź modelu dostarczała kąty wyznaczone klasycznymi metodami regresji. Model dla predykcji kątów nie był badany tak intensywnie jak ten dla wyznaczania energii, gdyż stanowił dodatek do całej pracy. Przeprowadzono uczenie na datasecie zawierającym $\sim 10^6$ kaskad, z czego $\sim 10^5$ przeznaczono na zbiór testowy. Wyniki predykcji przedstawiono na rys. 8.



Rysunek 8: Wykres wartości przewidzianej od wartości nominalnej dla kątów i miejsca padania cząstki

Mimo że wyniki nie dały zadowalających efektów, dały pewien pogląd na to, jak można poprawić model, aby wyniki były dokładniejsze. W idealnej sytuacji, wykresy powinny mieć postać rosnącej funkcji liniowej. Zostało to osiągnięte dla kątów, lecz dla położeń występują pewne punkty przegięcia. Najlepiej wypadło wyznaczenie rzutu kąta padania na oś X.

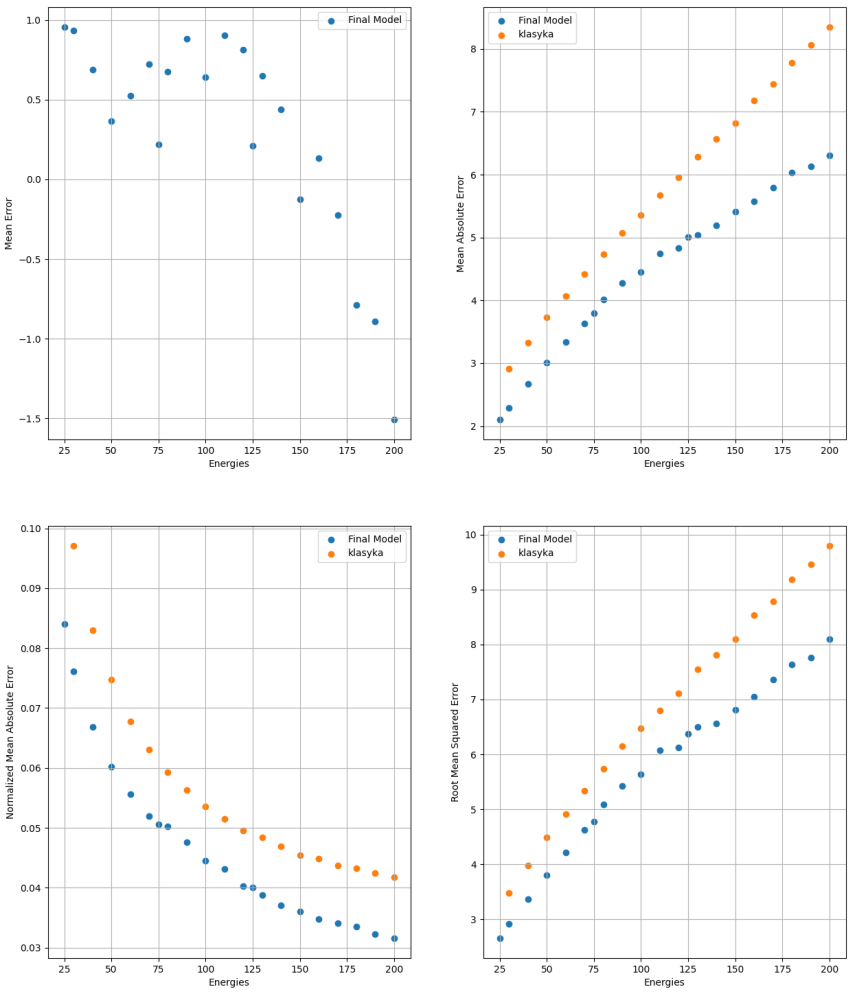
Nie został zbadany powód takiego działania modelu. Hipoteza jest taka, że w czterowymiarowej przestrzeni funkcji strat, model wpada w lokalne minimum, które jest optymalne dla wartości xAngle i dalej nie jest w stanie go opuścić. Rozwiązaniem tego mogłoby być np. zastosowanie warstwy Dropout, która wyłączałaby najsilniejsze neurony odpowiedzialne za przeuczanie względem wartości xAngle, lub regularyzacja.

8 Podsumowanie

W trakcie trwania projektu udało się zrealizować jeden z celów, jakim było za pomocą modeli sieci neuronowych zaprezentowanie metody lepszej rekonstrukcji energii w kalorymetrze elektromagnetycznym. Dla zaproponowanego modelu sprawdzono również jak jego rozmiar oraz różne techniki poprawy treningu lub zapobiegające przeuczaniu wpływają na wyniki. Finalny model służący do rekonstrukcji energii znajduje się na githubie (*scripts/ready_to_ICM/final_energy.py*), natomiast jego porównanie do metod "klasycznych" znajduje się na wykresie (rys. 9). Dalsze propozycje poprawy funkcjonowania modelu to:

- Dalsze eksperymenty z liczbą warstw i liczbą filtrów/neuronów
- Rygorystyczne sprawdzenie działania *BatchNorm*
- Dalsze dopasowanie wartości regularyzacji
- Przetestowanie innych metod zwalczających przeuczanie
- Sprawdzenie metod służących do niwelowania oscylacji funkcji straty podczas treningu (np. większe *batche*)

Wszystkie te techniki i kierunki rozwoju powinny również być sprawdzone dla modelu służącego do wyznaczania kąta oraz punktu początkowego kaskady. Z racji tego, iż ten model był sprawdzany podczas



Rysunek 9: Wykres przedstawiający zależności pomiędzy poszczególnymi metrykami, a wartością energii dla finalnego modelu.

projektu mniej ekstensywnie, należałoby również rozważyć dobranie architektury modelu do tego zadania. Możliwymi kierunkami jest rozdzielenie modelu na 2 mniejsze modele, które oddzielnie określałyby położenie oraz kąt. Taki podział wykluczyłby sytuacje, w których model optymalizuje swoje wagi zmniejszając błąd przy rekonstrukcji kąta, jednocześnie ignorując zadanie estymacji położenia początku kaskady. Naturalnym rozszerzeniem obu modeli również byłoby zadanie estymacji energii kaskad w przypadku, gdy do kalorymetru wpada więcej niż jedna cząstka.

Opisane w powyższym raporcie wyniki udało się uzyskać dzięki współpracy zespołowej i pracy indywidualnej. Poniżej zaprezentowano wkłady poszczególnych uczestników:

- Patrycja Maciejewska odpowiadała za opracowanie zliczania klasycznego energii i klasycznych metod estymacji pozycji wpadania wiązki i kąta. Oprócz tego pracowała też nad dobraniem odpowiednich parametrów regularyzacji i modelem kątowo-położeniowym
- Piotr Marczak wniósł do projektu wiedzę fizyczną z dziedziny cząstek elementarnych której brakowało innym uczestnikom. Odpowiadał za konstrukcje modelu położeniowo-kątowego, przygotowanie do niego danych i analizę wyników. Pracował też nad finalną formą modelu energetycznego.
- Stanisław Kiedrzyński przeprowadził początkowe testy różnych architektur i podejść do zagadnienia estymowania energii. W późniejszej fazie projektu odpowiadał za porównanie modeli trenowanych z różnymi funkcjami straty i modelami z różnymi wartościami. Zajmował się też wizualizacją danych, wyborem metryk i różnymi kwestiami organizacyjnymi. *dropoutu*.
- Przemysław Kaleta wniósł do projektu zdecydowanie najwięcej wiedzy informatycznej. Odpowiadał za przygotowanie pierwotnych danych, obsługę githuba i obsługę zasobów obliczeniowych na ICM. Dzięki jego pracy kilkukrotnie wzrosła szybkość szkolenia modeli. Współtworzył oryginalną architekturę, przyczynił się do testowania regularyzacji i opracował finalny model. Niniejszy raport współtworzony był wspólnie przez uczestników projektu.

Literatura

[1] Xingyu Zhang, Yiping Lu, and Dacheng Tao, *A Survey on the Effects of Batch Normalization in Deep Neural Networks*, arXiv preprint arXiv:2107.10207v3, 2021.

[2] Amit Yadav, *Understanding L1 and L2 Regularization in Machine Learning*, Medium, 2020.

[3] Nisha McNealis, *A Simple Introduction to Dropout Regularization (With Code!)*, Medium, 2021.

[4] Sergey Ioffe and Christian Szegedy, *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, Proceedings of the 32nd International Conference on Machine Learning (ICML), 2015.