Predykcja właściwości związków chemicznych

Celem zadania było wytrenowanie i zbadanie dwóch modeli opartych na grafowych sieciach neuronowych. Jeden z nich ma za zadanie identyfikować klasyfikować związki chemiczne na te które są lub nie inhibitorem beta-sekretazy, zaś drugi ma za zadanie przewidzieć wartość momentu dipolowego cząsteczek.

Klasyfikator inhibitorów beta-sekretazy:

W celu znalezienia optymalnego modelu wybrano do testów sieci składające się odpowiednio z warstwy typu *TransformerConv* oraz *GCNConv.* Każda z wariantów sieci posiadała 9 kanałów wejściowych, 64 jako rozmiar osadzenia i wyjściową warstwę liniową. Pomiędzy warstwami grafowymi zastosowano funkcję *relu.* Poniżej zebrano wyniki wartości miary accuracy na zbiorze walidacyjnym każdej z wariatów sieci przy danej liczbie warstw.

Liczba warstw	*TransformerConv*	*GCNConv*
1	*0.7237*	*0.6579*
2	*0.7039*	*0.7829*
3	*0.7171*	*0.7368*
4	*0.7237*	*0.7961*
6	*0.6447*	*0.8158*
8	*0.6201*	*0.5395*
26	*0.5395*	*0.4605*

Tab 1. Tabela wartości ACC dla różnej ilości warstw na zbiorze walidacyjnym

Najdokładniejsza okazała się być sieć złożoną z 6 warstw *GCNConv* i tą też architekturę rozwijano dalej. Porównywano też jak na jakoś klasyfikacji wpływa wielkość osadzeń przy wykorzystaniu wcześniej wybranej architektury.

Rozmiar osadzeń	4	16	32	64	128
ACC	0.4605	0.7566	0.7566	0.8158	0.6118

Tab 2. Wartości miary accracy dla różnych wartości rozmiaru osadzeń

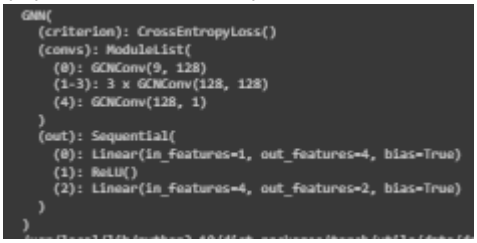
W poniższych eksperymentach użyto rozmiar osadzeń 1,2 i 64. Testowano również *dropout* po każdej z warstwie grafowej z prawdopodobieństwem 0.5, jednakże wyniki dokładności spadły z 82% do 63%. W wyniku czego zrezygnowano z tego kroku.

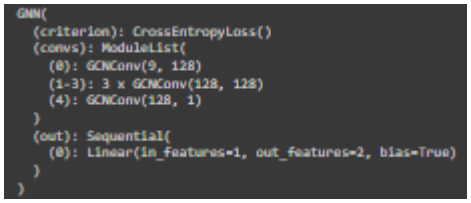
Kolejno przeprowadzono testy na różnych wielkościach warstwy ukrytej predykatora nieliniowego:

l. neuronów ukrytych	2	4	8	32	64	128	256
ACC	0.7434	0.7434	0.4605	0.5395	0.7434	0.5395	0.4605

Tab 3. Wartości miary accracy dla różnych ilości neuronów ukrytych w predykatorze nieliniowym

Optymalne architektury sieci:





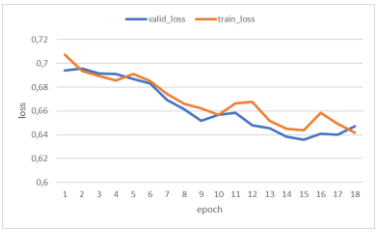
Zbiór BACE został podzielony na treningowy, testowy i walidacyjny w stosunku 0.8, 0.1, 0.1. Jako funkcję straty wykorzystano CrossEntropyLoss z odpowiednio dobranymi wagami, aby przeciwdziałać niezbalansowaniu zbioru. W procesie nauki wykorzystano optymalizator Adam z learning_rate równym 0.005. Aby zapobiec przeuczeniu zastosowano mechanizm *early stopping* w oparciu o zbiór walidacyjny. Nauka odbywała się na 1000 epokach.

Rys. 4 Wykres funkcji celu modelu z MLP dla osadzenia 64

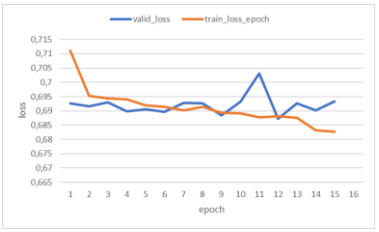
Rys. 4 Wykres funkcji celu modelu z MLP dla osadzenia 2

Rys. 4 Wykres funkcji celu modelu z MLP dla osadzenia 1

Wykres funkcji celu modelu bez MLP dla osadzenia 64



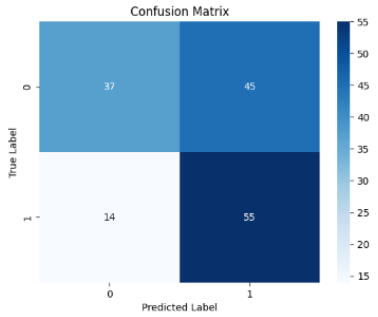
Wykres funkcji celu modelu bez MLP dla osadzenia 2



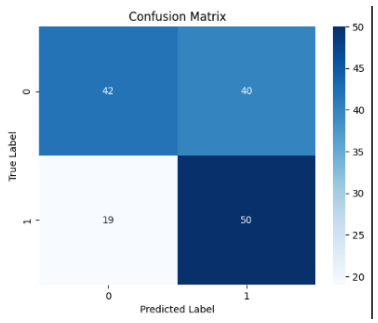
Wykres funkcji celu modelu bez MLP dla osadzenia 1

Wyniki:

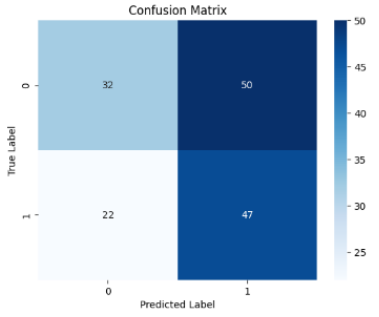
	Rozmiar osadzeń								
	64			1			2		
	PPV	TPR	F1	PPV	TPR	F1	PPV	TPR	F1
Klasa	MLP								
0	0.70	0.51	0.59	0.56	0.39	0.46	0.60	0.57	0.59
1	0.56	0.74	0.64	0.47	0.63	0.53	0.52	0.55	0.53
ACC	0.62			0.50			0.56		
	Liniowy								
0	0.73	0.45	0.55	0.69	0.51	0.59	0.59	0.39	0.47
1	0.55	0.79	0.65	0.55	0.72	0.63	0.48	0.68	0.57
ACC	0.68			0.61			0.52		



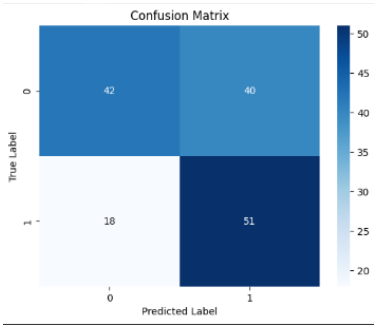
Rys. Macierzy pomyłek dla osadzenia 64 i predykatora liniowego



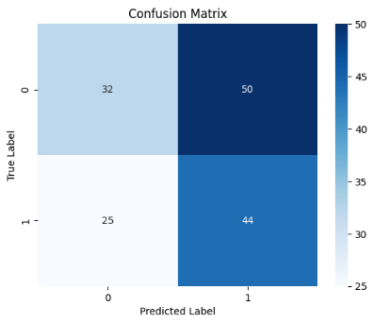
Rys. Macierzy pomyłek dla osadzenia 1 i predykatora liniowego



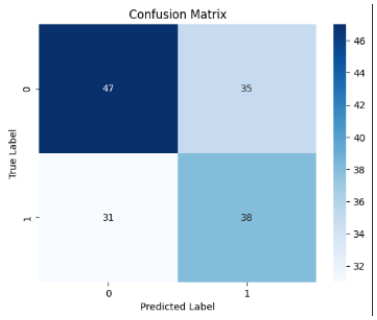
Rys. Macierzy pomyłek dla osadzenia 2 i predykatora liniowego



Rys. Macierzy pomyłek dla osadzenia 64 i predykatora nieliniowego

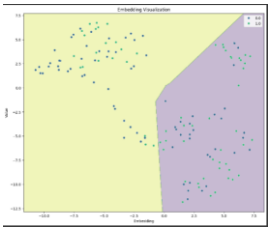


Rys. Macierzy pomyłek dla osadzenia 1 i predykatora nieliniowego

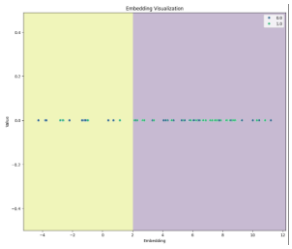


Rys. Macierzy pomyłek dla osadzenia 2 i predykatora nieliniowego

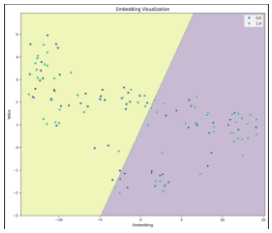
Jak widać z danych zawartych w powyższej tabeli, najlepsze wyniki otrzymano dla rozmiaru osadzenie 64. Zastosowanie MLP jako klasyfikatora nieznacznie w tym przypadku pogorszyło wyniki. Natomiast dla osadzeń 1 i 2 model klasyfikował z dokładnością zbliżoną do klasyfikacji losowej. Poniżej zaprezentowano wizualizację osadzeń.



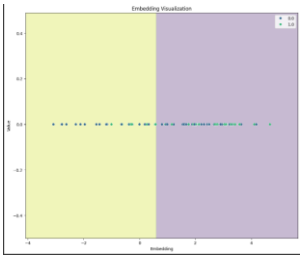
Rys Wizualizacja osadzenia rozmiaru 2 dla sieci z MLP



Rys Wizualizacja osadzenia rozmiaru 1 dla sieci z MLP



Rys Wizualizacja osadzenia rozmiaru 2 dla sieci z kl. liniowym



Rys Wizualizacja osadzenia rozmiaru 1 dla sieci z kl. liniowym

<u>Predykator wartości momentu dipolowego:</u>

W celu znalezienia optymalnego modelu wybrano do testów sieci składające się odpowiednio z warstwy typu *TransformerConv* oraz *GATConv.* Każda z wariantów sieci posiadała 11 kanałów wejściowych, 64 jako rozmiar osadzenia i wyjściową warstwę liniową. Pomiędzy warstwami grafowymi zastosowano funkcję *relu.* Poniżej zebrano wyniki wartości miary MAE na zbiorze walidacyjnym każdej z wariatów sieci przy danej liczbie warstw.

Liczba warstw	*TransformerConv*	*GATConv*
1	*0.893*	*0.956*
2	*0.712*	*0.894*
6	*0.529*	*0.770*
8	*0.652*	*0.530*

Tab 5. Tabela wartości MAE dla różnej ilości warstw sieci

Następnie dla architektury złożonej z 6 warstw *TransformerConv,* wykorzystującej cechy krawędzi, która uzyskała najmniejszą wartość błędu średnio-kwadratowego, szukano optymalnej liczby neuronów dla nieliniowego predykatora.

l. neuronów ukrytych	2	4	8	32	64
MSE	1.157	0.751	0.742	0.552	0.713

Tab 6. Wartości miary MAE dla różnych wartości rozmiaru osadzeń

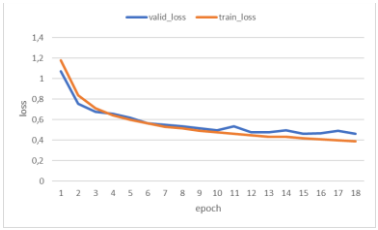
Jak wynika z powyższej tabeli, najdokładniej przewidywała wartości sieć złożona z 32 neuronów ukrytych i taką architekturę stosowano do dalszych eksperymentów.

Optymalne architektury sieci:

```
RGNN(
  (convs): ModuleList(
    (0): TransformerConv(11, 128, heads=1)
    (1-3): 3 x TransformerConv(128, 128, heads=1)
    (4): TransformerConv(128, 1, heads=1)
  )
  (out): Sequential(
    (0): Linear(in_features=1, out_features=1, bias=True)
  )
```

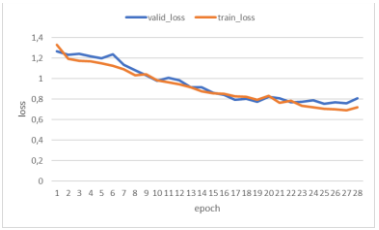
```
RGNN(
  (convs): ModuleList(
    (0): TransformerConv(11, 128, heads=1)
    (1-3): 3 x TransformerConv(128, 128, heads=1)
    (4): TransformerConv(128, 64, heads=1)
  )
  (out): Sequential(
    (0): Linear(in_features=64, out_features=32, bias=True)
    (1): ReLU()
    (2): Linear(in_features=32, out_features=1, bias=True)
  )
```

Zbiór Q9 został podzielony na treningowy, testowy i walidacyjny w stosunku 0.8, 0.1, 0.1. Jako funkcję straty wykorzystano MSELoss. W procesie nauki wykorzystano optymalizator Adam z learning_rate równym 0.001. Aby zapobiec przeuczeniu zastosowano mechanizm *early stopping* w oparciu o zbiór walidacyjny. Nauka odbywała się na 100 epokach.



Rys. Wykres funkcji celu modelu z MLP dla osadzenia 64

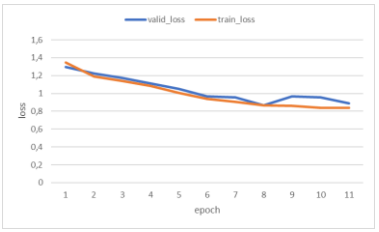
Rys. Wykres funkcji celu modelu z MLP dla osadzenia 2



Rys. Wykres funkcji celu modelu z MLP dla osadzenia 1

Wykres funkcji celu modelu bez MLP dla osadzenia 64

Wykres funkcji celu modelu bez MLP dla osadzenia 2



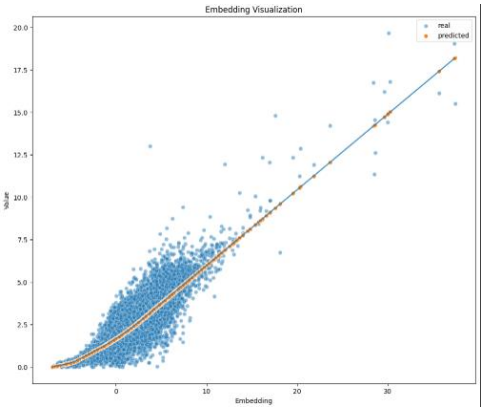
Wykres funkcji celu modelu bez MLP dla osadzenia 1

Wyniki:

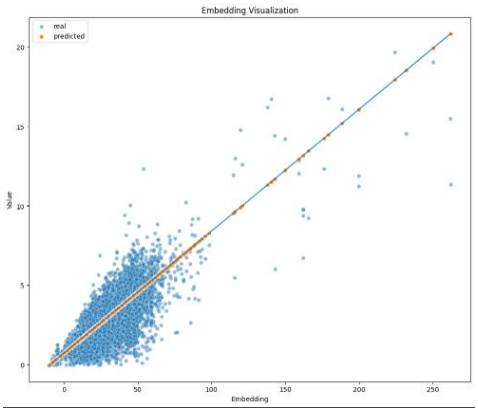
	Linera			MLP		
r. osadzeń	1	2	64	1	2	64
MAE	0.832	0.538	0.431	0.533	0.508	0.468

Tab 7 wyników metryki MAE na zbiorze testowym

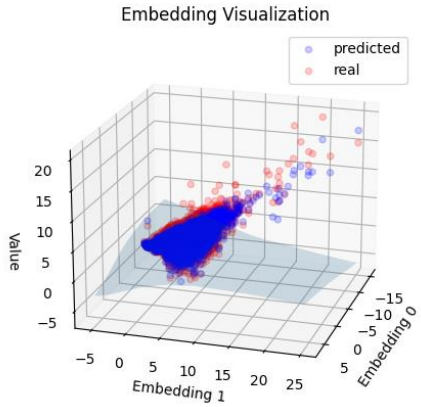
Błąd zarówno dla sieci z liniową warstwą jak i MLP wybranej architektury był zbliżony. Jednakże, to dla wariantu architektury z predykartorem liniowym i osadzeniem 64 okazał się najlepszy, a tym samo wartości przewidywane najmniej różniły się od tych rzeczywistych. Poniżej zaprezentowano wizualizację osadzeń w przypadku wymiaru 1 i 2.



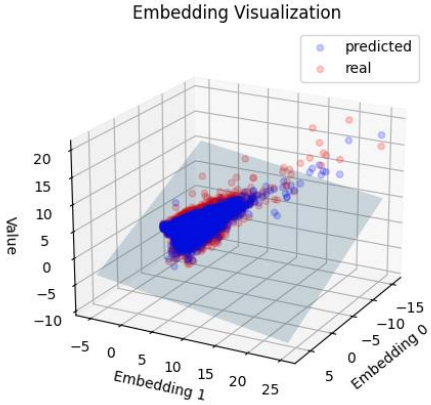
Rys. Wizualizacja osadzeń o wymiarze 1 i nieliniowego predykatora



Rys. Wizualizacja osadzeń o wymiarze 1 i liniowego predykatora



Rys. Wizualizacja osadzeń o wymiarze 2 i nieliniowego predykatora



Rys. Wizualizacja osadzeń o wymiarze 2 i liniowego predykatora