

Tytuł: Klasyfikacja Pokémonów po typach na podstawie ich statystyk

Data: 14.06.2025

Autor: Maksymilian Janica

1. Cel projektu

Celem niniejszego projektu jest zbadanie, czy na podstawie statystyk bazowych Pokémona (takich jak HP, Atak, Obrona, itd.) można przewidzieć jego typ (lub typy). Projekt opiera się na zastosowaniu sieci neuronowych do klasyfikacji typów Pokémonów w różnych wariantach zestawu danych.

2. Motywacja

Typy Pokémonów odgrywają kluczową rolę w rozgrywce – zarówno ofensywnie, jak i defensywnie. W społeczności fanów Pokémonów panuje przekonanie, że istnieją wyraźne zależności pomiędzy typem Pokémona a jego statystykami bazowymi. Przykładowe powszechnie przyjmowane założenia to:

- Pokemony typu **Steel** cechują się wysokimi statystykami obronnymi, ale niższymi ofensywnymi.
- Pokemony typu **Psychic** często mają wysokie statystyki specjalnego ataku i obrony.

Celem eksperymentu jest sprawdzenie, czy tego typu przekonania znajdują odzwierciedlenie w danych i czy możliwa jest skuteczna klasyfikacja typu Pokémona wyłącznie na podstawie jego statystyk.

3. Opis Danych

Dane wykorzystane w projekcie zostały pobrane z portalu Kaggle:

<https://www.kaggle.com/datasets/abcsds/pokemon>

Zbiór danych zawiera informacje o **721 Pokémonach** oraz ich cechach i statystykach. Każdy wiersz reprezentuje jednego Pokémona i zawiera następujące kolumny:

- # – Numer identyfikacyjny Pokémona (zgodny z numeracją w grach).

- **Name** – Nazwa Pokémona.
- **Type 1** – Główny (pierwszy) typ Pokémona.
- **Type 2** – Dodatkowy (drugi) typ Pokémona, jeśli występuje (może być pusty).
- **Total** – Suma wszystkich statystyk bazowych.
- **HP** – Punkty życia (Health Points).
- **Attack (Atk)** – Statystyka odpowiadająca za fizyczne obrażenia.
- **Defense (Def)** – Statystyka odpowiadająca za redukcję fizycznych obrażeń.
- **Special Attack (Sp. Atk)** – Statystyka odpowiadająca za specjalne (niefizyczne) obrażenia.
- **Special Defense (Sp. Def)** – Statystyka odpowiadająca za redukcję obrażeń specjalnych.
- **Speed (Spd)** – Statystyka określająca kolejność działania w turze (wyższa wartość = pierwszy ruch).
- **Generation** – Generacja, w której dany Pokémon zadebiutował.
- **Legendary** – Wartość logiczna (True/False) określająca, czy Pokémon jest legendarny.

Przykład:

#	Name	Type 1	Type 2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Generation	Legendary
49	Venomoth	Bug	Poison	450	70	65	60	90	75	90	1	False

4.Założenia badawcze

W ramach przeprowadzonego projektu przyjęto kilka kluczowych założeń, które następnie poddano weryfikacji eksperymentalnej:

- 1. Pokemony różnych typów wykazują różnice w rozkładzie statystyk.**
Zakłada się, że typ Pokémona wiąże się z określonym profilem statystyk bazowych. Przykładowo, Pokemony typu Steel mają zwykle wyższe statystyki defensywne, podczas gdy typ Psychic może charakteryzować się wysokimi wartościami Sp. Atk i Sp. Def. Celem jest sprawdzenie, czy te obserwacje mają odzwierciedlenie w danych.
- 2. Typ Pokémona można skutecznie przewidzieć na podstawie jego statystyk bazowych.**
Założono, że jeśli różnice między typami są wystarczająco wyraźne, to model oparty na sieciach neuronowych powinien być w stanie nauczyć się wzorców i przypisać odpowiedni typ (lub pierwszy typ) na podstawie samych statystyk (takich jak HP, Attack, Defense, itd.).

3. Kombinacje typów są traktowane niezależnie od kolejności.

W przypadku Pokémonów o dwóch typach przyjęto, że kolejność typów nie ma znaczenia. Oznacza to, że Pokémon typu *Grass/Poison* jest traktowany tak samo jak *Poison/Grass*. To uproszczenie umożliwia spójne grupowanie danych i ułatwia analizę modeli klasyfikacyjnych.

5. Warianty eksperymentów klasyfikacyjnych

W celu zbadania zależności między statystykami bazowymi a typem Pokémona, przeprowadzono cztery niezależne eksperymenty klasyfikacyjne. Każdy z nich różnił się zakresem danych oraz podejściem do reprezentacji typów Pokémonów. Celem było określenie, w których warunkach sieć neuronowa najlepiej radzi sobie z przewidywaniem typu.

Opisane poniżej warianty zostały szczegółowo przeanalizowane w dalszych podrozdziałach:

◊ Wariant 1 – Klasyfikacja po pierwszym typie (Type 1)

Uwzględniono wszystkie Pokemony, niezależnie od tego, czy mają jeden czy dwa typy. W przypadku Pokémonów dwutypowych, analizie poddano wyłącznie pierwszy typ (Type 1). To podejście traktuje problem jako klasyczną klasyfikację wieloklasową.

◊ Wariant 2 – Klasyfikacja tylko Pokémonów monotypowych

W tym przypadku z danych **usunięto wszystkie Pokemony posiadające Type 2**, pozostawiając jedynie Pokemony monotypowe.

Celem było zbadanie, czy uproszczenie problemu (brak dodatkowego typu) i ograniczenie do mniejszego, czystszego zbioru danych przełoży się na lepsze wyniki klasyfikacji. Tak jak wcześniej, klasyfikowano według Type 1.

◊ Wariant 3 – Klasyfikacja tylko Pokémonów dwutypowych (połączone typy)

Ten wariant obejmował wyłącznie Pokemony posiadające oba typy (Type 1 oraz Type 2).

Każda unikalna kombinacja dwóch typów została potraktowana jako **jedna klasa** – np.

`Grass_Poison`, `Water_Ice`, `Bug_Flying`. Kolejność typów nie jest zachowywana (czyli `Grass_Poison` = `Poison_Grass`).

Ze względu na dużą liczbę możliwych kombinacji, klasyfikację ograniczono do `TOP_K_TYPES` najczęstszych par typów, co pozwoliło na zmniejszenie niebalansowania danych i bardziej stabilne uczenie.

❖ Wariant 4 – Klasyfikacja wszystkich Pokémonów po kombinacji typów (mono + dual)

W tym podejściu każdy Pokémon – niezależnie od tego, czy posiada jeden czy dwa typy – został zaklasyfikowany do unikalnej klasy na podstawie **połączenia Type 1 i Type 2**. Pokémonom monotypowym przypisywano sztuczną wartość "None" jako Type 2, co pozwalało na ich uwzględnienie jako np. `Fire_None`, `Psychic_None`.

Kombinacje typów były traktowane jako pojedyncze klasy, a klasyfikacja miała formę **wieloklasową**. Ograniczono liczbę klas do najczęstszych (`TOP_K_TYPES`), by poprawić równowagę danych i skuteczność modelu.

5.1 Klasyfikacja po pierwszym typie Pokémona (Type 1)

Opis podejścia

W tym wariantie analizowano wszystkie dostępne Pokemony, niezależnie od tego, czy posiadają jeden czy dwa typy. Do klasyfikacji wykorzystywano wyłącznie **pierwszy typ (Type 1)**, który według założeń stanowi główną cechę charakteryzującą Pokémona.

Wariant ten pozwala uprościć problem do klasyfikacji **wieloklasowej z jedną etykietą** i objąć możliwie największy zbiór danych. W celu zmniejszenia niebalansowania klas, klasyfikację ograniczono do **{TOP_K_TYPES} najczęściej występujących typów**.

Przygotowanie danych

- Dane wczytano z oryginalnego zbioru zawierającego 721 Pokémonów.
- Zbiór danych został przefiltrowany tak, aby uwzględnić tylko rekordy, których Type 1 należy do {TOP_K_TYPES} najczęstszych.

- Wybrane cechy wejściowe: Total, HP, Attack, Defense, Sp. Atk, Sp. Def, Speed.
- Cechy zostały wstandaryzowane przy użyciu StandardScaler.
- Etykiety zakodowano numerycznie za pomocą LabelEncoder.
- Podział danych: 70% – trening, 30% – test.
- Użyto `class_weight='balanced'` do kompensacji różnic liczebności między typami.

Wariant TOP_K_TYPES = 4

Architektura modelu

W ramach eksperymentu z ograniczoną liczbą klas (TOP_K_TYPES = 4) poszukiwano możliwie najprostszego, stabilnego modelu, który nie będzie podatny na przeuczenie.

Poniższa tabela przedstawia architekturę najbardziej optymalnego modelu:

warstwa	Liczba neuronów	Funkcja aktywacji	Dropout
Dense	8	{aktywator}	2.0
Dense	4	{aktywator}	-
Dense	num_classes	softmax	-

Zwiększenie liczby warstw lub neuronów skutkowało pogorszeniem modelu, co zostanie przedstawione w dalszej części raportu na podstawie poniższego modelu.

warstwa	Liczba neuronów	aktywator	dropout
Dense	8	{aktywator}	0.2
Dense	4	{aktywator}	-
Dense	2	{aktywator}	-
Dense	num_classes	softmax	-

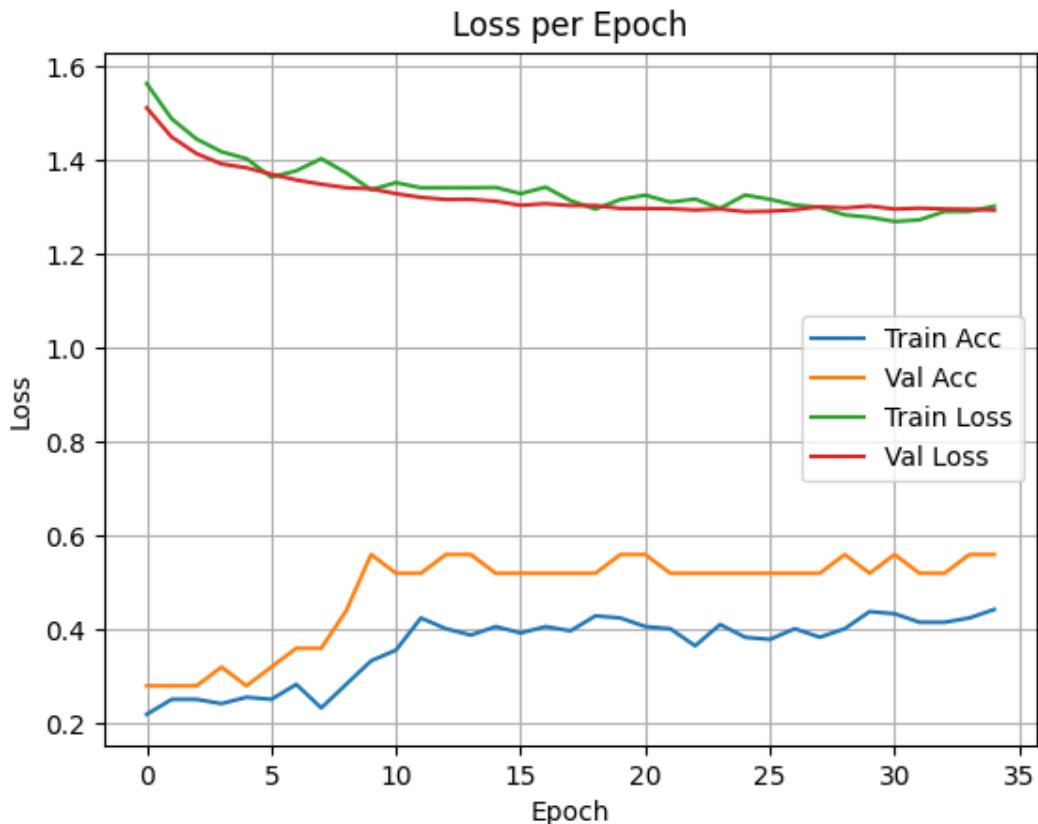
Porównanie funkcji aktywacji

Model przetestowano z trzema różnymi funkcjami aktywacji:

- ReLU
- Tanh
- ELU

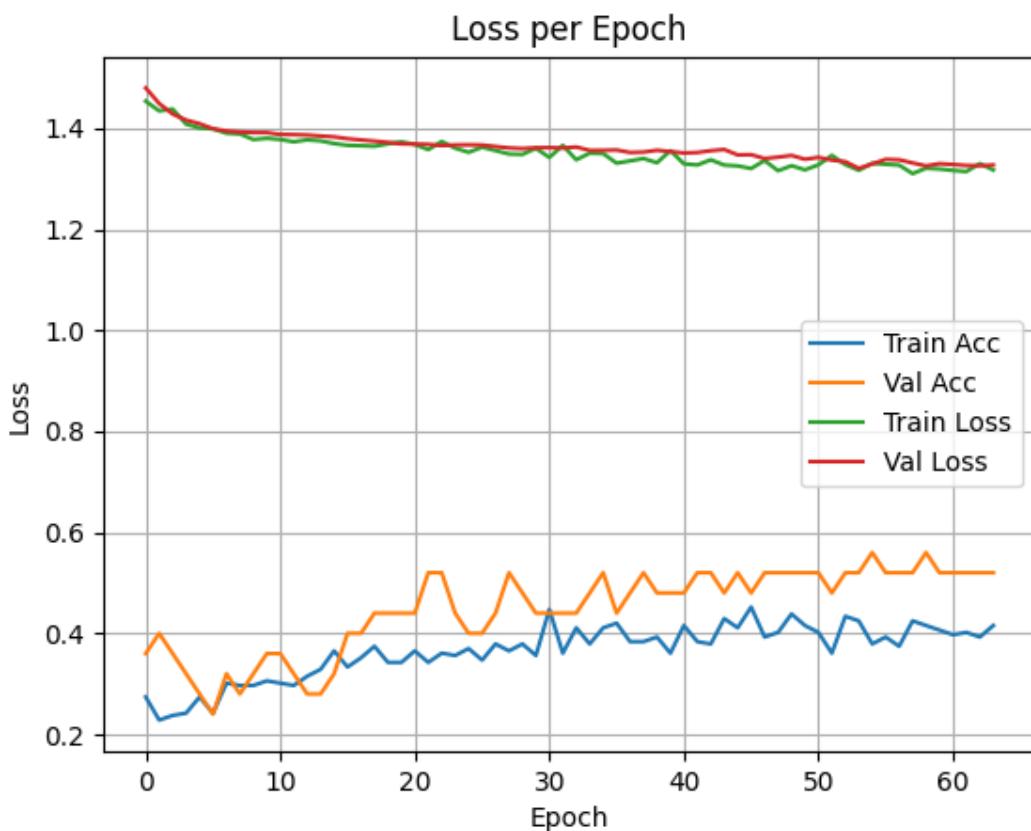
Dla każdego przypadku przeanalizowano przebieg dokładności i funkcji straty w czasie treningu.

ReLU



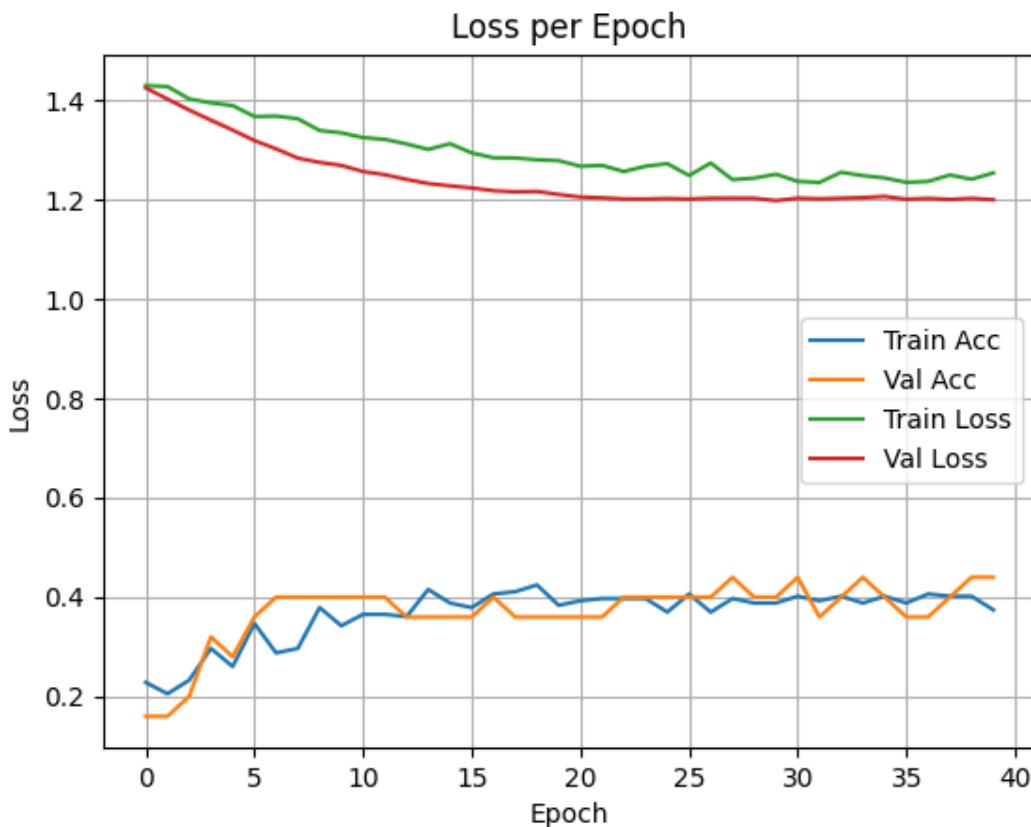
- *Opis przebiegu treningu:*
 - Z powyższego wykresu wynika, że przebieg funkcji straty (loss) jest stabilny, a krzywe dla zbioru treningowego i walidacyjnego są do siebie zbliżone, co sugeruje prawidłowe uczenie się modelu bez wyraźnych oznak przeuczenia.
 - Zauważalnie wyższa dokładność (accuracy) na zbiorze walidacyjnym niż na treningowym może natomiast wskazywać, że skuteczność modelu **jest wrażliwa na skład danych** – tzn. konkretne Pokemony obecne w zbiorze walidacyjnym mogą ułatwiać klasyfikację. Taka sytuacja może być też efektem **losowego podziału danych** i powinna zostać potwierdzona powtórzonymi eksperymentami

Wariant z dodatkową warstwą neuronów:



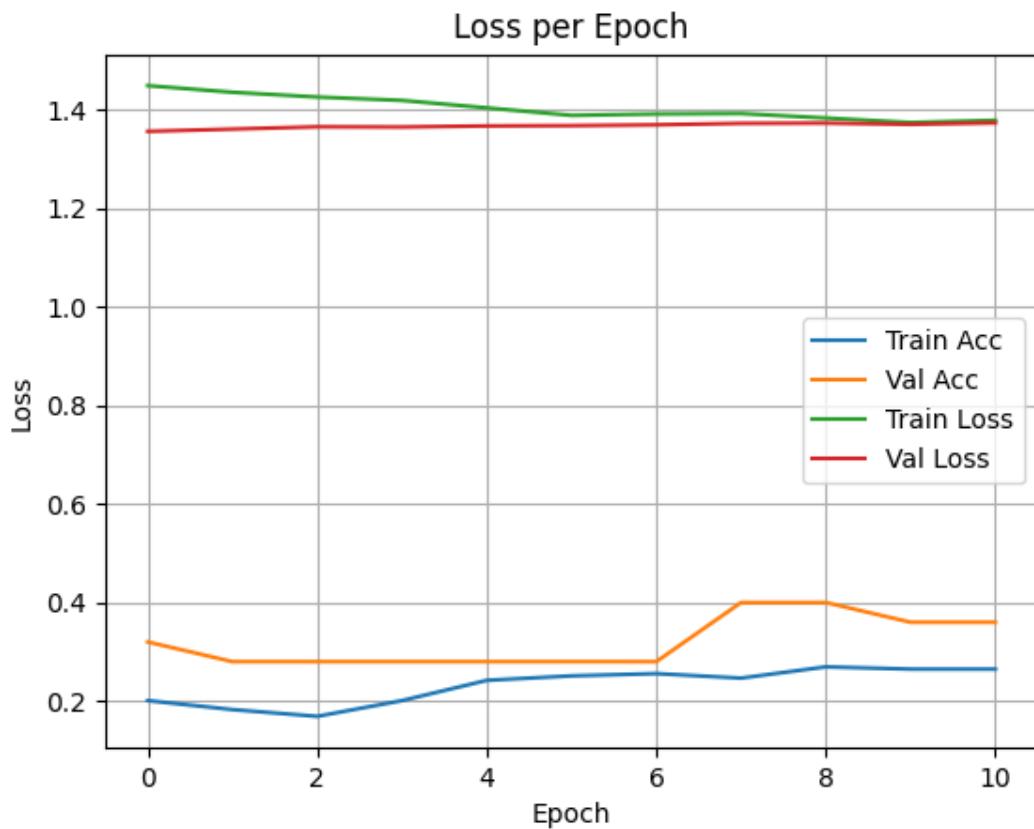
- *Opis przebiegu treningu:*
 - Z powyższego wykresu wynika, że – podobnie jak w wariantie bez dodatkowej warstwy neuronowej – przebieg funkcji straty (loss) pozostaje stabilny, a krzywe dla zbioru treningowego i walidacyjnego są do siebie zbliżone, co sugeruje prawidłowy proces uczenia się bez wyraźnych oznak przeuczenia.
 - Warto jednak zauważyć, że wartości straty w tym wariantie są **wyraźnie wyższe** niż w modelu bazowym, co może świadczyć o **gorszej jakości dopasowania** modelu do danych.
 - Podobnie jak wcześniej, dokładność (accuracy) na zbiorze walidacyjnym przewyższa dokładność na zbiorze treningowym, jednak tym razem jest **mniej stabilna** – widoczne są większe wahania między epokami. Może to sugerować, że model jest bardziej wrażliwy na konkretne przypadki w zbiorze walidacyjnym, a jego ogólna zdolność do generalizacji jest niższa niż w wariantie bez dodatkowej warstwy.

Tanh



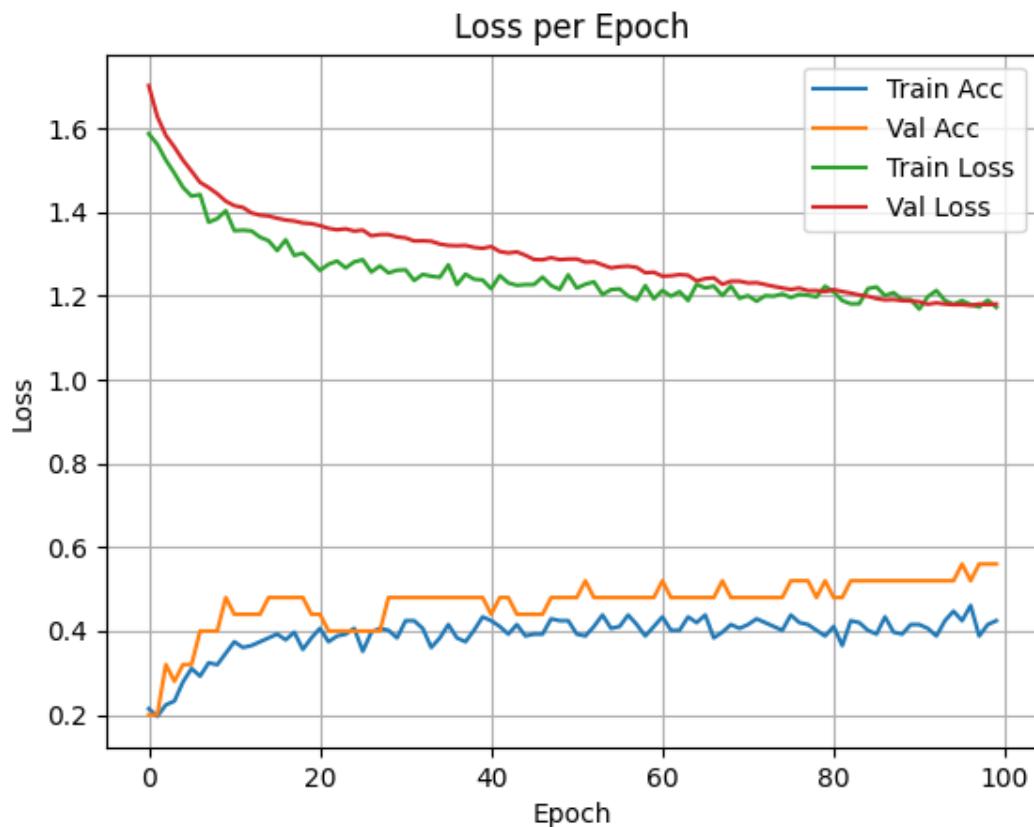
- W przypadku funkcji aktywacji tanh zaobserwowano, że przebieg funkcji straty (loss) na zbiorze walidacyjnym jest **bardziej stabilny** niż w przypadku ReLU, a jej wartości są **niższe zarówno w trakcie treningu, jak i po jego zakończeniu**. Może to sugerować, że tanh pozwala modelowi lepiej dopasować się do danych walidacyjnych pod względem funkcji kosztu.
- Pomimo niższej wartości straty, dokładność klasyfikacji (accuracy) osiągnięta przez model była **istotnie niższa – o około 20 punktów procentowych w porównaniu do ReLU**. Może to wskazywać, że funkcja tanh sprzyja bardziej „zachowawczemu” uczeniu, minimalizując stratę kosztem zdolności modelu do podejmowania zdecydowanych decyzji klasyfikacyjnych.
- Różnica między loss a accuracy w tym przypadku podkreśla, że **niska funkcja straty nie zawsze przekłada się bezpośrednio na wysoką skuteczność klasyfikacji**, szczególnie przy zastosowaniu softmax i strat typu crossentropy, gdzie wartości predykcji mogą być poprawne, ale zbyt „rozmyte”, by zapewnić wysokie accuracy.

Wariant z dodatkową warstwą neuronów:



Na podstawie wykresu funkcji straty można zauważyc, że:

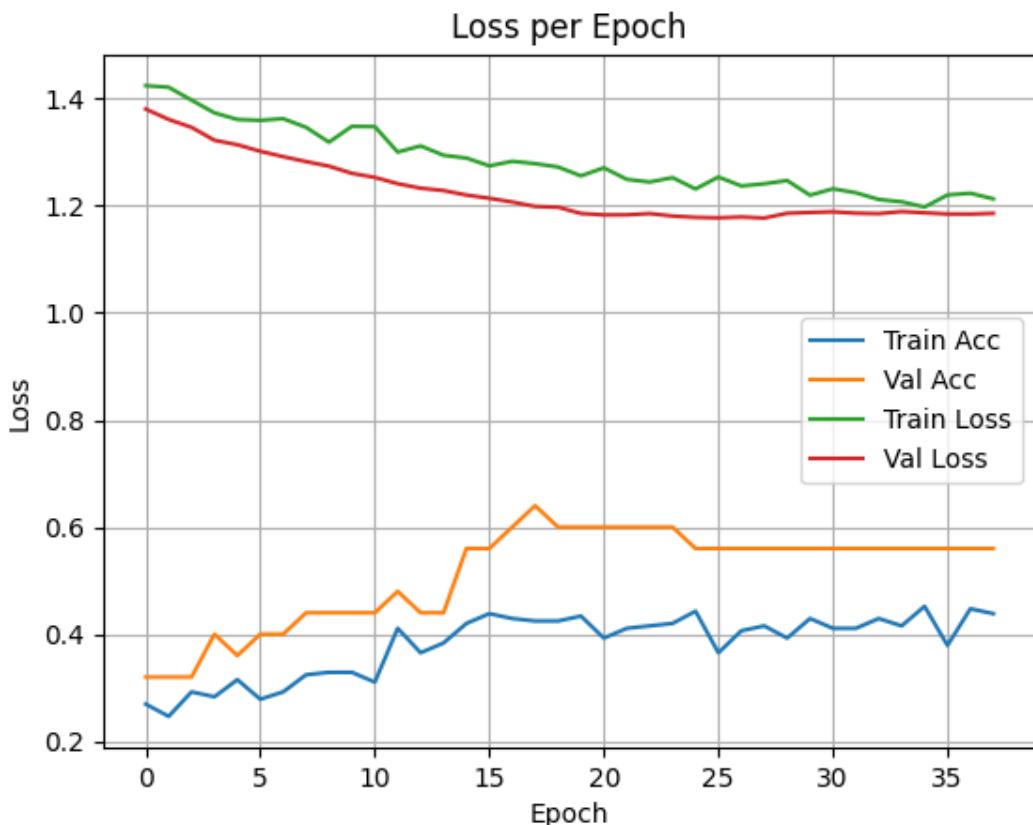
- **Strata walidacyjna (val_loss) delikatnie rośnie** w trakcie treningu, co może wskazywać na początek przeuczenia.
- Jednocześnie **strata testowa lekko maleje, a następnie stabilizuje się na poziomie zbliżonym do walidacyjnej**, co sugeruje względnie ustabilizowane, choć nieoptymalne, dopasowanie do danych testowych.



Na podstawie wykresu można zauważyć, że:

- Zarówno **funkcja straty dla zbioru treningowego (Train Loss)**, jak i **walidacyjnego (Val Loss)** systematycznie maleją przez większość epok, co świadczy o prawidłowym przebiegu procesu uczenia.
- Pod koniec treningu wartości Train Loss i Val Loss **zblążają się do siebie i stabilizują na podobnym poziomie** (~1.2), co sugeruje, że model **nie przeuczył się** i generalizuje stosunkowo dobrze.
- **Dokładność (Accuracy)** rośnie w pierwszych kilkunastu epokach, a następnie stabilizuje się:
 - Na zbiorze treningowym (linia niebieska) osiąga wartość około **0.42**,
 - Na zbiorze walidacyjnym (linia pomarańczowa) osiąga wartość około **0.47–0.48**, utrzymując się lekko powyżej treningowej.

Wariant z dodatkową warstwą neuronową:



- Na podstawie wykresu można zauważyć, że funkcja straty (loss) zarówno dla zbioru treningowego, jak i walidacyjnego **systematycznie maleje przez cały przebieg treningu**, co sugeruje, że model uczy się stabilnie i bez oznak przeuczenia. W końcowej fazie wartości Train Loss i Val Loss **zblążają się do siebie i stabilizują na poziomie około 1.2**, co wskazuje na dobre dopasowanie.
- Dokładność (accuracy) rośnie przez pierwsze kilkanaście epok, a następnie stabilizuje się – osiągając około **43% dla treningu i ~55% dla walidacji**, z widoczną przewagą na zbiorze walidacyjnym. Może to świadczyć o lekkiej zależności wyników od konkretnego podziału danych, jednak brak gwałtownych skoków sugeruje ogólnie dobre zachowanie modelu.
- Zastosowanie funkcji aktywacji ELU w połączeniu z głębszą siecią pozwoliło utrzymać stabilność uczenia, jednak **dalsze zwiększenie liczby warstw nie przełożyło się na znaczący wzrost dokładności**, co może sugerować, że istotniejsze są cechy danych niż sama złożoność modelu.

Wnioski ogólne

Przeprowadzone eksperymenty pozwoliły zaobserwować istotne zależności między funkcją aktywacji, architekturą modelu a jego skutecznością w zadaniu klasyfikacji typu Pokémona na podstawie statystyk.

1. **Najlepsze rezultaty** uzyskano przy zastosowaniu funkcji aktywacji ELU, która zapewniała stabilne uczenie, niski poziom funkcji straty oraz względnie wysoką dokładność predykcji. Jej działanie było zauważalnie lepsze niż ReLU oraz Tanh, zwłaszcza pod względem zbieżności i braku przeuczenia.
2. Wprowadzenie **dodatkowej warstwy neuronowej** nie poprawiło jakości klasyfikacji – pomimo spadku funkcji straty, dokładność modelu ustabilizowała się na poziomie około **40–45%**. W niektórych przypadkach pogorszyło to zdolność generalizacji, co może sugerować, że zwiększenie złożoności modelu przy tej liczbie danych i klas było nieoptymalne.
3. Niska różnica między `train loss` a `val loss` we wszystkich testach wskazuje, że zastosowane techniki regularizacji (dropout, wagi klas) skutecznie ograniczyły przeuczenie, jednak **górny pułap dokładności (~50%)** może wynikać z ograniczeń samego zbioru danych (niewielka liczba przykładów per klasa, niewielka rozpiętość statystyk w obrębie typów).
4. W niektórych przypadkach dokładność walidacyjna przewyższała treningową, co może sugerować **niereprezentatywny podział danych** lub silne zróżnicowanie między klasami.

Wariant TOP_K_TYPES = 8

W ramach eksperymentu z ograniczoną liczbą klas (`TOP_K_TYPES = 8`) poszukiwano możliwie najprostszego, stabilnego modelu, który nie będzie podatny na przeuczenie.

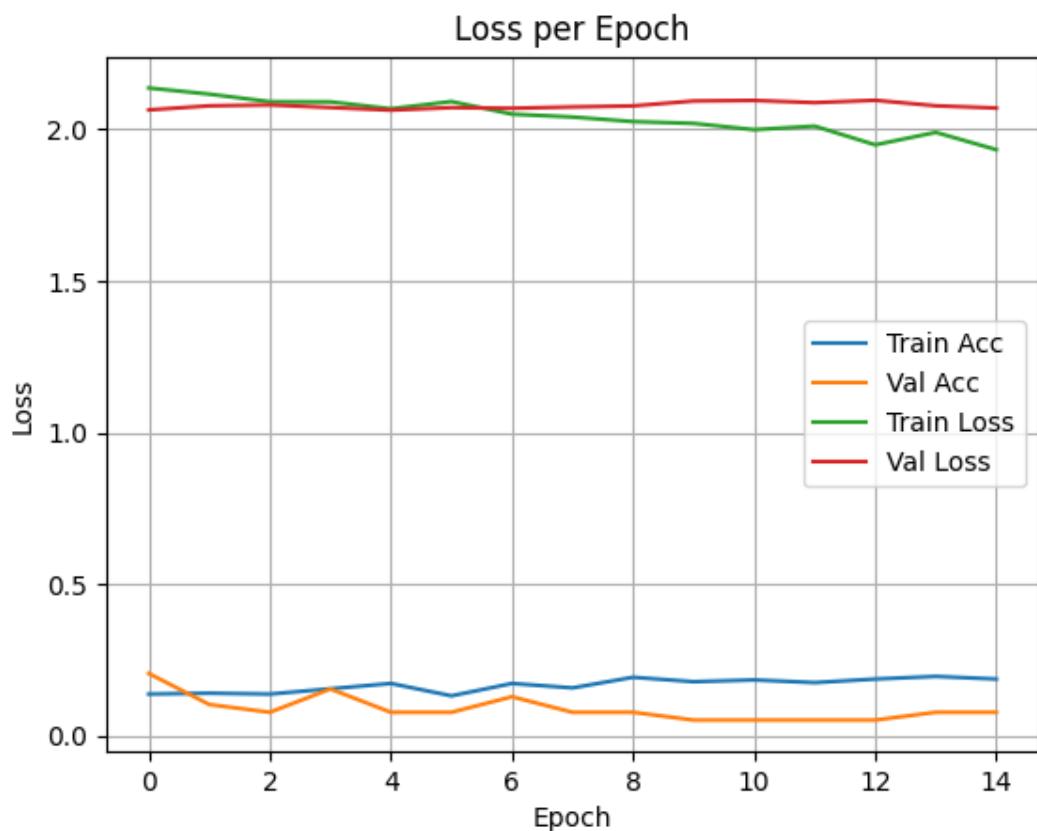
Poniższa tabela przedstawia architekturę najbardziej optymalnego modelu:

Warstwa	Liczba neuronów	aktywator	Dropout
Dense	64	{aktywator}	0.4
Dense	32	{aktywator}	0.3
Dense	16	{aktywator}	0.2
Dense	8	{aktywator}	-
Dense	Num_classes	softmax	-

W tym przyadku testowanao też jak zwiększenie szerokości wpływa na model, dlatego dla najlepszego z poprzedniego eksperimentu (elu) zrobiono wariant z podwojona ilością neuronów na każdej warstwie

Warstwa	Liczba Neuronów	aktywator	Dropout
Dense	128	{aktywator}	0.4
Dense	64	{aktywator}	0.3
Dense	32	{aktywator}	0.2
Dense	16	{aktywator}	-
	Num_classes	softmax	-

Relu



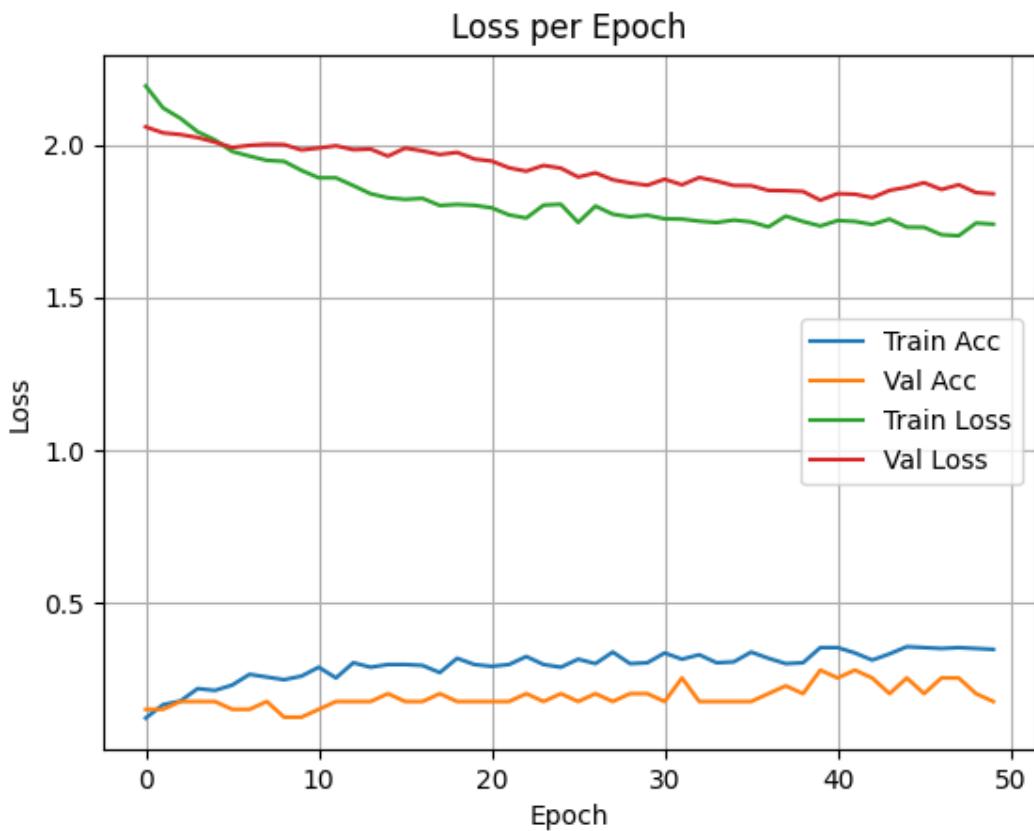
- Opis przebiegu treningu:
 - W przypadku funkcji aktywacji ReLU oraz ograniczenia klasyfikacji do 8 najczęściej występujących typów Pokémonów, model **nie wykazał oznak efektywnego uczenia**.
 - Wykres funkcji straty (loss) pokazuje, że zarówno Train Loss, jak i Val Loss **pozostają na stosunkowo wysokim poziomie (~2.0)** i **nie wykazują wyraźnej tendencji spadkowej**, co oznacza, że model **nie**

zdołał dopasować się ani do danych treningowych, ani do walidacyjnych.

- Równocześnie, wartości dokładności (accuracy) pozostają bardzo niskie — w obu przypadkach poniżej **20%**, a przebieg jest stosunkowo płaski, bez wyraźnej poprawy w czasie.

W związku z **niską skutecznością uzyskaną przy zastosowaniu wcześniejszej ustalonej optymalnej architektury**, zrezygnowano z przeprowadzenia eksperymentu z modelem o **zwiększonej szerokości warstw**, uznając, że dalsze zwiększanie złożoności mogłoby jedynie nie przynieść istotnej poprawy wyników.

Tanh

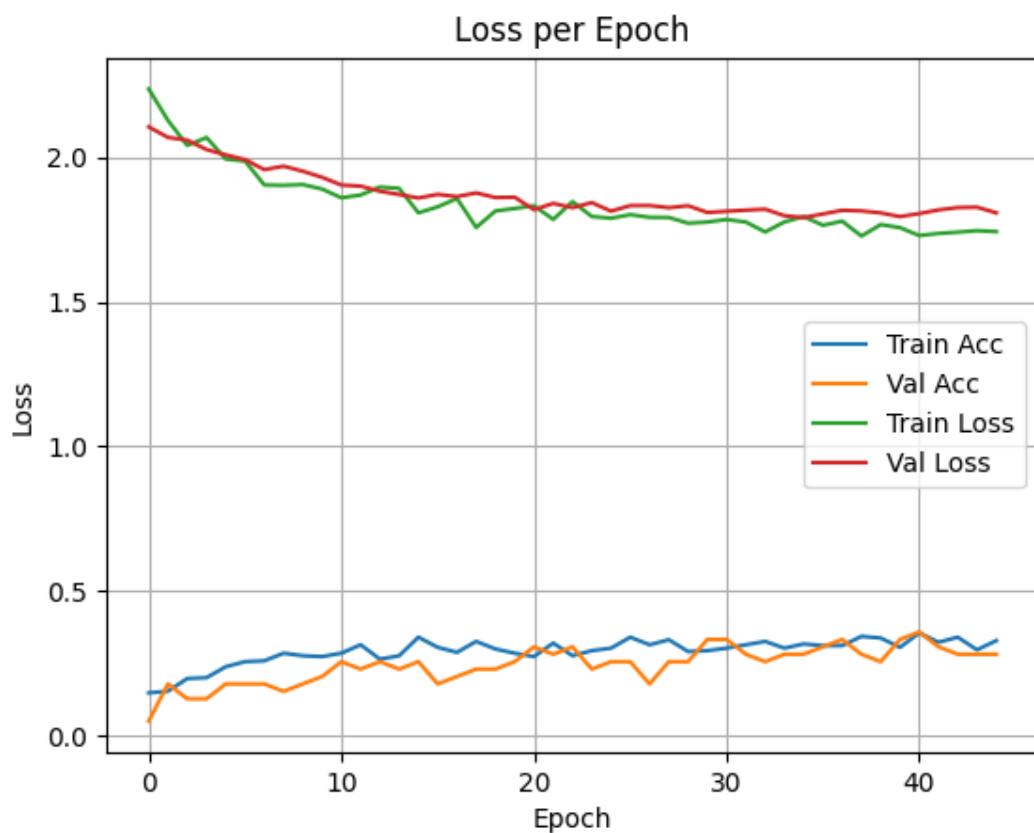


- Opis treningu modelu:
 - W przypadku funkcji aktywacji Tanh oraz ograniczenia klasyfikacji do 8 najczęściej występujących typów Pokémonów, model **nie wykazał oznak efektywnego uczenia**.
 - Zaobserwowano istotną różnicę pomiędzy stratą treningową a walidacyjną, co sugeruje przeuczenie modelu. Model dobrze dopasował się do danych treningowych, jednak nie uogólnia skutecznie na danych walidacyjnych.

- W przypadku przedstawionego modelu **dokładność na zbiorze treningowym wyraźnie przewyższa dokładność walidacyjną**, osiągając poziom około **35%**, podczas gdy walidacyjna nie przekracza **25–30%**. Taka różnica sugeruje, że model ma trudności z uogólnieniem wiedzy poza dane treningowe, co może wskazywać na **niewystarczające dopasowanie do zbioru walidacyjnego**. Jedną z możliwych przyczyn jest **duża zmienność lub niestabilność danych**, np. silne różnice pomiędzy rozkładami typów Pokémonów w zbiorach treningowym i walidacyjnym, brak reprezentatywności danych lub ich niezbalsansowanie.

W związku z **niską skutecznością uzyskaną przy zastosowaniu wcześniejszej ustalonej optymalnej architektury**, zrezygnowano z przeprowadzenia eksperymentu z modelem o **zwiększonej szerokości warstw**, uznając, że dalsze zwiększenie złożoności mogłoby jedynie pogłębić problem przeuczenia lub nie przynieść istotnej poprawy wyników.

Elu

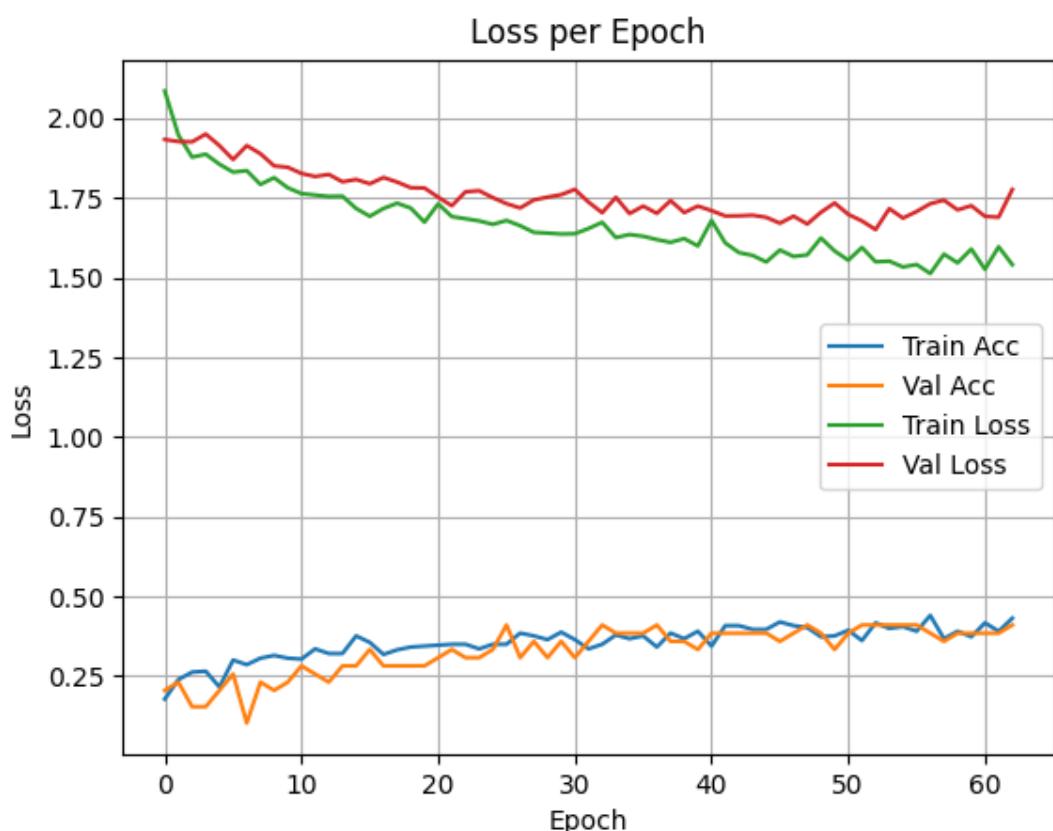


- Opis treningu modelu:
 - Krzywe funkcji straty w początkowej fazie treningu przebiegają prawidłowo — nie występuje między nimi duży rozrzut, co sugeruje

stabilne uczenie się modelu. Jednak około **40. epoki** można zaobserwować **pierwsze oznaki przeuczenia**: tempo spadku straty walidacyjnej (`val loss`) znaczco się zmniejsza, podczas gdy strata treningowa (`train loss`) nadal wyraźnie maleje. To rozbieżność typowa dla modelu, który zaczyna zapamiętywać dane treningowe kosztem generalizacji.

- Wartości dokładności (accuracy) przez większość treningu również wskazują na tę zależność — **dokładność walidacyjna utrzymuje się poniżej dokładności treningowej**, lecz pod koniec treningu obie stabilizują się na zbliżonym poziomie, wynoszącym około **30%**. Może to świadczyć o ograniczonej zdolności modelu do rozróżniania typów Pokémonów na podstawie samych statystyk, mimo względnie stabilnego procesu uczenia.

Wariant z większą szerokością modelu:



- Opis treningu modelu:
 - Już od początkowych epok można zaobserwować oznaki **przeuczenia** — krzywe straty dla zbioru treningowego i walidacyjnego rozchodzą się stosunkowo wcześnie, a **val loss utrzymuje się wyraźnie powyżej train loss** przez większość przebiegu treningu.

- Pomimo tego, dokładność modelu na zbiorze treningowym i walidacyjnym pozostaje **zblizona**, osiągając maksymalnie około **40%**. Taka sytuacja może wskazywać, że model był w stanie nauczyć się pewnych ogólnych wzorców, jednak **nadmiernie dopasował się do danych treningowych**, co ograniczyło jego zdolność generalizacji.
- Ze względu na **szybsze pojawienie się przeuczenia oraz brak wyraźnej przewagi dokładności**, model ten został oceniony jako **mniej efektywny niż przyjęta wcześniej wersja optymalna**.

Wnioski ogólne

5. Przeprowadzone eksperymenty pozwoliły zaobserwować istotne zależności między wyborem funkcji aktywacji, architekturą modelu a jego skutecznością w zadaniu klasyfikacji typu Pokémona na podstawie statystyk.
6. Najlepsze wyniki osiągnięto ponownie przy użyciu funkcji aktywacji **ELU** — głównie dlatego, że **ReLU wykazywało problemy z konwergencją** i model z jej użyciem **niemal w ogóle się nie uczył**, natomiast **tanh prowadziło do szybszego i wyraźniejszego przeuczenia** modelu, mimo początkowo obiecującego przebiegu funkcji straty.
7. Zwiększenie szerokości warstw neuronowych nie przełożyło się na poprawę skuteczności klasyfikacji. Chociaż w niektórych wariantach zaobserwowano wzrost dokładności do poziomu **około 40%**, towarzyszył temu również **szybszy wzrost straty walidacyjnej**, co wskazuje na **niedostateczne dopasowanie i przeuczenie modelu**.
8. Dodatkowo, w niektórych przypadkach dokładność na zbiorze walidacyjnym była wyższa niż na treningowym, co może świadczyć o **niereprezentatywnym podziale danych lub znacznym zróżnicowaniu rozkładów klas** pomiędzy zbiorami. Wskazuje to na potrzebę dalszej analizy struktury danych oraz ewentualnego zastosowania bardziej zaawansowanych technik podziału i balansu klas.

Wariant TOP_K_TYPES = 16

W ramach eksperymentu z ograniczoną liczbą klas (**TOP_K_TYPES = 16**) poszukiwano możliwie najprostszeego, stabilnego modelu, który nie będzie podatny na przeuczenie, jednakże patrząc tylko na najlepszy model z poprzednich eksperymentów (Elu).

Poniższa tabela przedstawia architekturę najbardziej optymalnego modelu:

Warstwa	Liczba Neuronów	Aktywator	Dropout
Dense	128	{aktywator}	0.5
Dense	64	{aktywator}	0.5
Dense	32	{aktywator}	0.4
Dense	16	{aktywator}	0.3
Dense	8	{aktywator}	-
Dense	Numb_classes	softmax	-

Przetostowano również jak wpłynie na model zwiększenie warstw lub szerokości modelu:

Głębszy model:

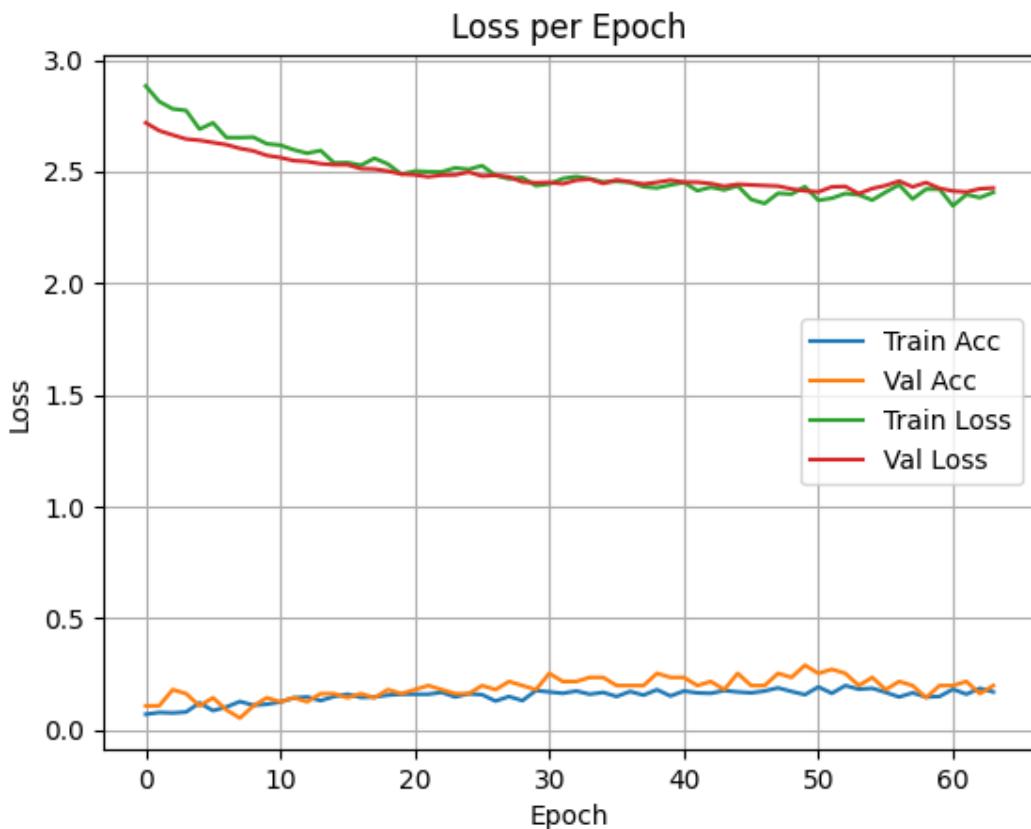
Warstwa	Liczba Neuronów	Aktywator	Dropout
Dense	256	{aktywator}	0.5
Dense	128	{aktywator}	0.5
Dense	64	{aktywator}	0.4
Dense	32	{aktywator}	0.4
Dense	16	{aktywator}	0.3
Dense	8	{aktywator}	-
Dense	Numb_classes	{aktywator}	-

Szerszy model:

Warstwa	Liczba Neuronów	Aktywator	Dropout
Dense	256	{aktywator}	0.5
Dense	128	{aktywator}	0.5
Dense	64	{aktywator}	0.4
Dense	32	{aktywator}	0.3
Dense	16	{aktywator}	-
Dense	Numb_classes	softmax	-

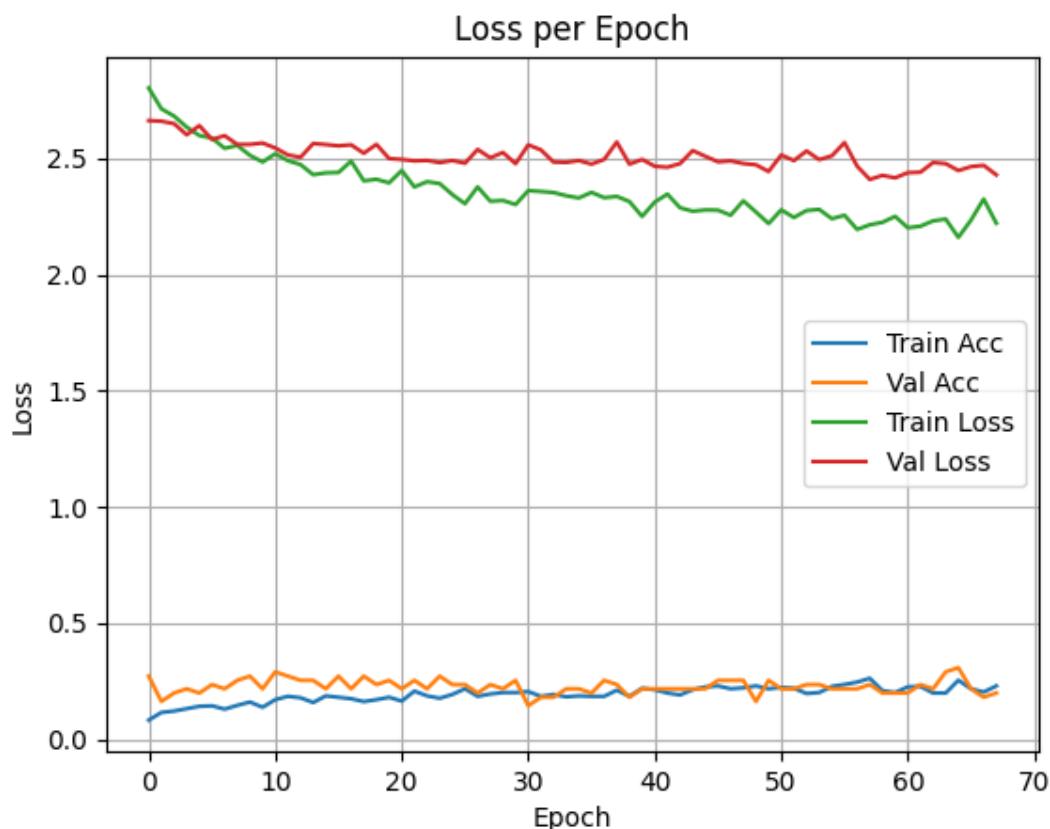
Opis nauki Modelu

Optymalny model:



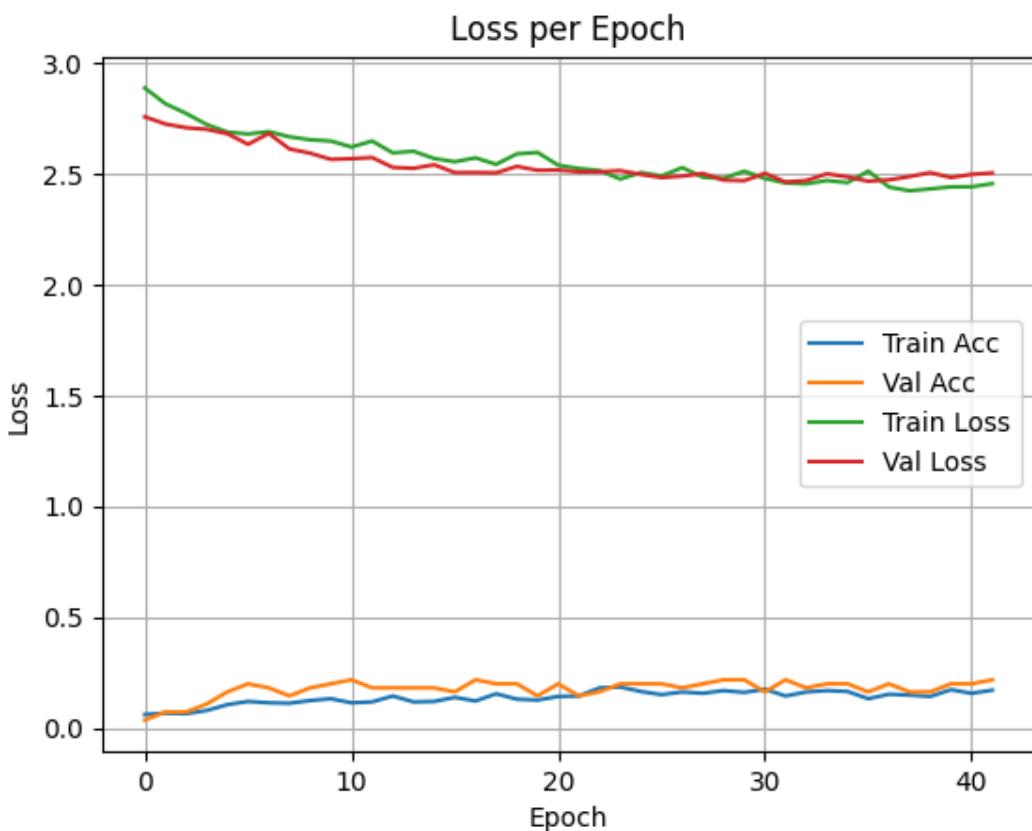
- Opis:
 - Krzywe funkcji straty dla zbioru treningowego i walidacyjnego **przebiegają bardzo blisko siebie**, co sugeruje stabilny proces uczenia się. Jednak osiągają one **relatywnie wysokie wartości (nieco poniżej 2.5)**, co może wskazywać na trudność zadania lub niedostateczne dopasowanie modelu. Pod koniec treningu krzywa straty dla zbioru treningowego zaczyna znajdować się **niżej niż walidacyjna**, co może świadczyć o **początkach przeuczenia**, które mogłyby się pogłębić przy dłuższym treningu.
 - Krzywe dokładności (accuracy) również wykazują **zbieżny przebieg**, jednak przez większość czasu dokładność walidacyjna utrzymuje się **nieznacznie powyżej dokładności treningowej**. Pod koniec procesu treningowego obie krzywe **przecinają się i stabilizują na poziomie około 20%**, co potwierdza ograniczoną zdolność modelu do skutecznej klasyfikacji przy zastosowanej konfiguracji.

Szerszy Model:



- Opis:
 - Krzywe funkcji straty dla zbioru treningowego i walidacyjnego **wyraźnie się rozchodzą**, co wskazuje na **przeuczenie modelu**. Strata treningowa systematycznie maleje, natomiast walidacyjna przestaje się poprawiać i utrzymuje się na wyraźnie wyższym poziomie.
 - W przypadku dokładności (accuracy), początkowo wartość dla zbioru walidacyjnego przewyższa dokładność treningową, jednak w trakcie treningu obie krzywe **zbliżają się do siebie i kończą na podobnym poziomie — około 25%**, co potwierdza ograniczoną zdolność modelu do skutecznej generalizacji mimo relatywnie stabilnego przebiegu dokładności.

Głębszy Model:



- Opis:
 - Krzywe funkcji straty dla zbioru treningowego i walidacyjnego **przebiegają stosunkowo blisko siebie i stabilizują się na poziomie około 2.5**, co sugeruje, że model nie wykazuje silnych oznak przeuczenia, ale również **nie osiąga dobrego dopasowania**.
 - Krzywe dokładności (accuracy) również pozostają zblżone przez cały okres treningu i kończą się na poziomie około **20%**, co wskazuje na **ograniczoną skuteczność modelu** w klasyfikacji. Pomimo stabilnego przebiegu uczenia, niska dokładność sugeruje, że model **nie zdołał uchwycić wystarczająco silnych zależności w danych wejściowych**.

Wnioski ogólne

9. Przeprowadzone eksperymenty z wykorzystaniem funkcji aktywacyjnej **ELU** – wcześniej najskuteczniejszej w wariantach z ograniczoną liczbą klas (**TOP_K_TYPES = 4 i 8**) – nie przyniosły zadowalających rezultatów przy większej liczbie typów.

10. Pomimo modyfikacji architektury modelu (zarówno poprzez zwiększanie szerokości, jak i głębokości sieci), **dokładność klasyfikacji oscylowała wokół poziomu 20%**, co jest znacznie poniżej oczekiwania i wyraźnie gorsze niż w prostszych wariantach (dla TOP_K_TYPES = 4 osiągnięto nawet 50% dokładności).
11. Wyniki te sugerują, że **podejście polegające na klasyfikacji wyłącznie pierwszego typu Pokémona (Type 1)**, niezależnie od istnienia drugiego typu, **nie odzwierciedla w pełni struktury i złożoności danych**, przez co **model traci informację niezbędną do poprawnej klasyfikacji**.

Możliwe przyczyny niskiej skuteczności tego podejścia

1. Utrata informacji o drugim typie (Type 2)

W przypadku Pokémonów złożonych z dwóch typów, pominięcie drugiego typu skutkuje zredukowaniem opisu jednostki – model nie ma dostępu do pełnej informacji, co może znacząco ograniczać jego zdolność rozróżniania klas.

2. Zacieranie różnic między klasami

Niektóre typy Pokémonów występują bardzo często w kombinacjach (np. Poison/Grass i Grass/Poison), co może prowadzić do sytuacji, w której różne przypadki mają bardzo podobne statystyki, ale różne etykiety – model nie jest w stanie ich rozdzielić na podstawie samych danych liczbowych.

3. Zbyt wiele klas przy niewystarczającej liczbie przykładów

Przy zwiększeniu liczby klas (TOP_K_TYPES > 8), dane stają się bardziej rozproszone – każdy typ ma relatywnie mniej przykładów, co może prowadzić do **niedostatecznego nauczenia się niektórych klas** oraz trudności z generalizacją.

4. Złożoność relacji statystyki → typ

Statystyki bazowe Pokémonów są **niewystarczająco unikalne dla jednego typu** – wiele różnych typów może mieć podobne profile statystyczne (np. wysoka Sp. Atk i niska Def), co utrudnia klasyfikację, szczególnie gdy jeden z typów jest pomijany.

5. Zakłócenia w strukturze etykiet

Część Pokémonów jedno-typowych i dwu-typowych została potraktowana w sposób jednolity (wszyscy klasyfikowani po Type 1), co może **wprowadzić chaos w strukturze klas** – niektóre klasy będą mieszać dane z bardzo różnych przypadków.

5.2. Klasyfikacja tylko Pokémonów jednotypowych (mono-typed)

Opis podejścia

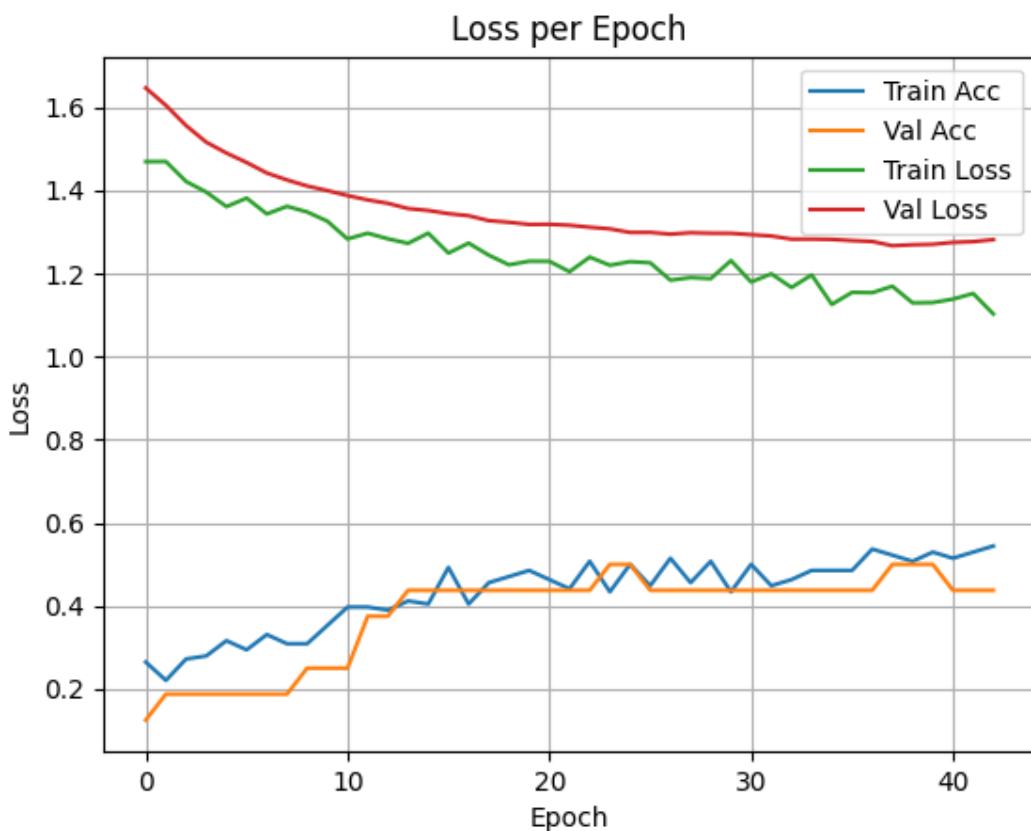
W tym wariantie zdecydowano się zbadać, czy model jest w stanie nauczyć się zależności między statystykami a typem Pokémona w przypadku ograniczenia zbioru wyłącznie do Pokémonów posiadających **jeden typ (mono-typed)**. Celem było sprawdzenie, czy usunięcie Pokémonów dwutypowych (które mogą zacierać granice między klasami) pozwoli osiągnąć lepsze wyniki klasyfikacji.

Do eksperymentu wykorzystano architekturę modelu, która wcześniej okazała się najskuteczniejsza – z funkcją aktywacji **ELU** – testowaną na zestawach z ograniczoną liczbą typów (TOP_K_TYPES = 4 i 8). Umożliwiło to ocenę, czy podobne zależności występują również w przypadku danych jednotypowych.

Przygotowanie danych

- Dane wczytano z oryginalnego zbioru zawierającego 721 Pokémonów.
- Zbiór danych został przefiltrowany tak, aby uwzględnić tylko rekordy, których Type 1 należy do {TOP_K_TYPES} najczęstszych.
- Wybrane cechy wejściowe: Total, HP, Attack, Defense, Sp. Atk, Sp. Def, Speed.
- Cechy zostały wstandaryzowane przy użyciu StandardScaler.
- Etykiety zakodowano numerycznie za pomocą LabelEncoder.
- Podział danych: 70% – trening, 30% – test.
- Użyto `class_weight='balanced'` do kompensacji różnic liczebności między typami.

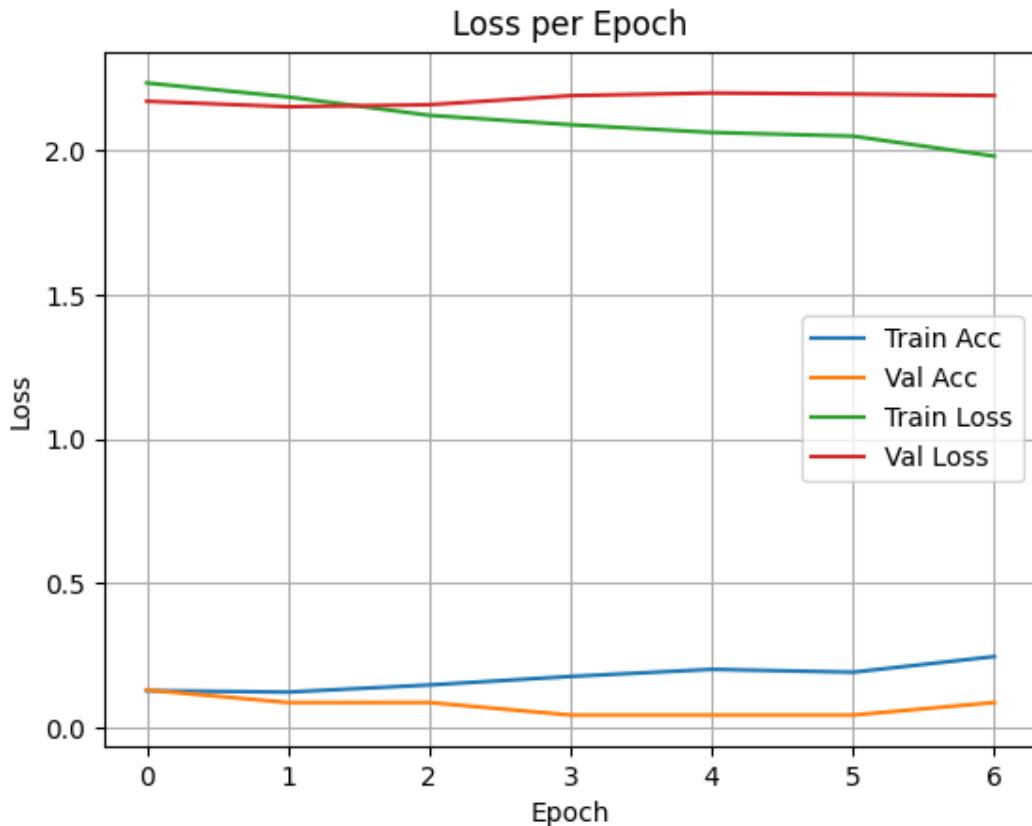
Wariant TOP_K_TYPES = 4



Opis przebiegu treningu modelu

- Już od początku procesu treningowego obserwowane jest wyraźne **przeuczenie modelu** – krzywa straty dla zbioru walidacyjnego (val loss) utrzymuje się znaczco powyżej krzywej dla zbioru treningowego i ta różnica z czasem się pogłębia.
- Pomimo przeuczenia, **dokładność modelu (accuracy)** zarówno na zbiorze testowym, jak i walidacyjnym przekracza poziom 40%, przy czym dokładność na zbiorze testowym przez większość czasu pozostaje wyższa od walidacyjnej. Może to sugerować, że model lepiej generalizuje na nieznanych danych testowych niż na wyodrębnionej części walidacyjnej, co może wynikać z niereprezentatywnego podziału danych

Wariant TOP_K_TYPES = 8



Opis przebiegu treningu – mono-typed, TOP_K_TYPES = 8

Krzywe straty (loss) od samego początku sugerują przeuczenie – **strata walidacyjna (val loss)** jest stale wyższa niż **strata treningowa (train loss)**, a różnica między nimi rośnie wraz z kolejnymi epokami. Model szybko dostosowuje się do danych treningowych, ale nie przenosi tej wiedzy skutecznie na dane walidacyjne.

Dokładność (accuracy) treningowa oraz walidacyjna są stosunkowo niskie i oscylują w zakresie **10–25%**, przy czym dokładność walidacyjna jest przez większość czasu niższa od treningowej. To wskazuje, że model ma trudności z uogólnieniem nauczonych wzorców.

Ogólnie, pomimo niewielkiej liczby epok, widoczne jest wyraźne przeuczenie i brak poprawy jakości klasyfikacji, co może sugerować zbyt mało danych, zbyt duży rozrzut wewnętrz klas, albo brak reprezentatywności statystyk bazowych dla rozróżniania typów Pokémonów w tym wariantie.

Wnioski

Eksperymenty przeprowadzone na zbiorach ograniczonych wyłącznie do Pokémonów posiadających tylko jeden typ (mono-typed) wykazały, że model opracowany w poprzednim wariantie – oparty na klasyfikacji na podstawie pierwszego typu (Type 1) – **nie poradził sobie skutecznie w tym scenariuszu.**

Pomimo wykorzystania najlepiej wcześniej sprawdzającej się architektury (z funkcją aktywacji ELU), model nie był w stanie osiągnąć satysfakcjonujących wyników.

Dokładność klasyfikacji utrzymywała się na niskim poziomie (ok. 20–40%), a wykresy strat wyraźnie wskazywały na problem przeuczenia. W niektórych przypadkach już po kilku epokach strata walidacyjna znacznie przewyższała stratę treningową, co świadczy o słabej zdolności modelu do generalizacji.

Te rezultaty sugerują, że w przypadku mono-typed Pokémonów **sam typ (Type 1) nie zawiera wystarczająco informacji powiązanych z rozkładem statystyk**, aby model mógł skutecznie rozróżniać klasy. Możliwe, że w przypadku Pokémonów jednego typu różnice w statystykach są zbyt subtelne lub niejednoznaczne, przez co model nie jest w stanie wyłapać spójnych wzorców.

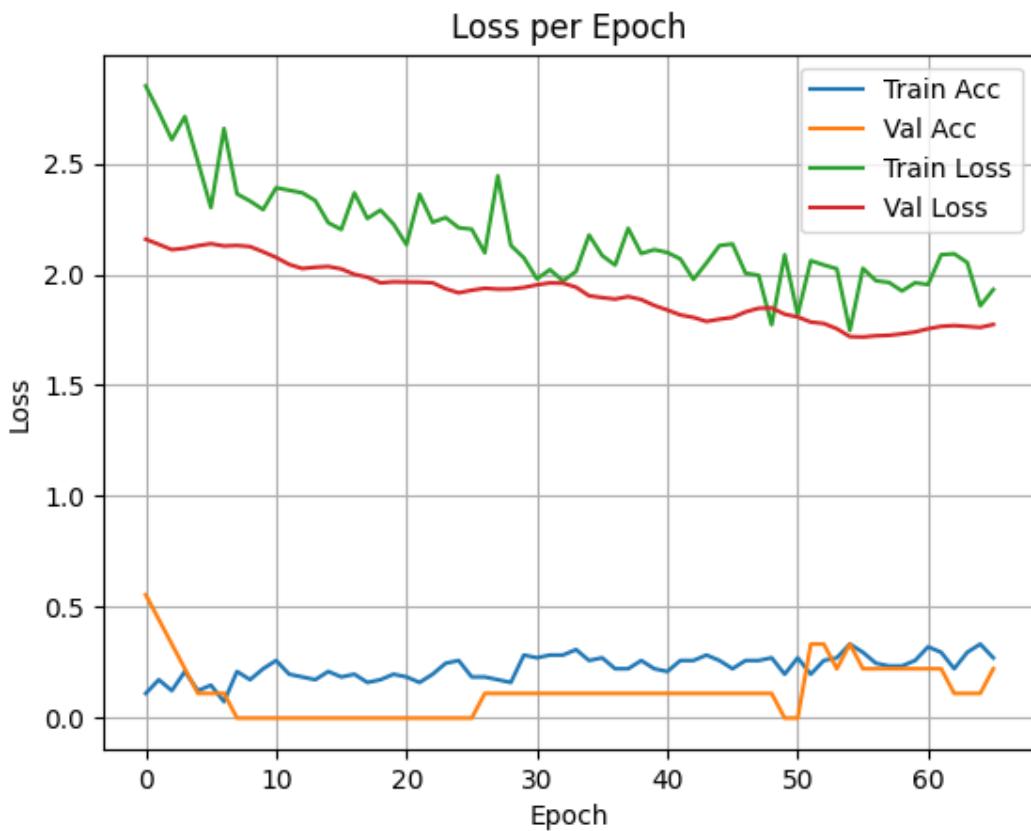
5.3. Wariant z tylko Pokémonami o dwóch typach (dual type)

Opis:

W tym wariantie zdecydowano się przetestować model z wariantu 1 na zbiorze zawierającym wyłącznie Pokemony posiadające dwa typy (dual typed). Celem było sprawdzenie, czy model z aktywacją ELU będzie w stanie nauczyć się zależności w bardziej złożonym przypadku dwóch typów, oraz czy osiągnie lepsze wyniki niż w przypadku wariantu 5.2 (tylko Pokemony jednotypowe).

Do eksperimentu użyto **dokładnie tej samej architektury modelu** co wcześniej – liczba warstw, liczba neuronów oraz wszystkie hiperparametry zostały zachowane bez zmian. Model wykorzystywał aktywację ELU, czyli funkcję, która wcześniej dawała najlepsze rezultaty.

Dodatkowo, testy przeprowadzono tylko na **próbce Pokémonów należących do 10 najczęstszych kombinacji dwóch typów**, co miało na celu zapewnienie odpowiednio licznego i reprezentatywnego zbioru danych do nauki i walidacji.



Opis treningu modelu:

W trakcie treningu modelu na zbiorze Pokémonów posiadających dwa typy (dual typed) zaobserwowano **stabilny spadek straty (loss)** zarówno na zbiorze treningowym, jak i walidacyjnym. Krzywe te pozostają względnie blisko siebie, co może sugerować, że model nie ulega łatwo przeuczeniu – przynajmniej pod względem funkcji kosztu.

Mimo pozytywnego wyglądu krzywych straty, dokładność (accuracy) pozostaje **niesatysfakcjonująca** – oscyluje wokół 20–30%, a przez dużą część treningu walidacyjna celność bywa niestabilna lub bliska零. Możliwe jest, że model zyskuje pewien potencjał reprezentacyjny, ale złożoność problemu oraz wysokie podobieństwo między niektórymi typami utrudniają mu wyraźne rozróżnienie klas.

Wnioski:

Model zastosowany w wariantie 5.3 (ten sam, co w wariantie bazowym z funkcją aktywacji ELU) wykazuje pewien **potencjał** – co widać po malejących krzywych straty, jednak mimo to **nie osiąga zadowalającej dokładności**. Może to wynikać z:

- dużej **złożoności kombinacji dwóch typów Pokémonów**, które mogą wprowadzać niejednoznaczność dla klasyfikatora,
- ograniczonej pojemności modelu, który może być zbyt płytki lub zbyt prosty, by uchwycić złożone zależności między cechami a etykietami dual types,
- możliwego **niedostatecznego zróżnicowania danych treningowych** – niektóre kombinacje mogą być dominujące lub bardzo rzadkie,
- potencjalnej potrzeby bardziej zaawansowanej architektury, np. uwzględniającej zależności nieliniowe lub kombinacje cech (interakcje).

5.4. Klasyfikacja wszystkich Pokémonów z uwzględnieniem dual types

Opis podejścia:

W tym wariantie zdecydowano się na objęcie klasyfikacją **całej dostępnej populacji Pokémonów**, niezależnie od tego, czy posiadają jeden czy dwa typy. W przypadku Pokémonów posiadających tylko jeden typ (tzw. *mono-typed*), jako drugi typ przypisano wartość "None", tworząc sztuczny typ złożony, np. "Bug_None", "Fire_None" itd.

Tym samym każda etykieta klasyfikacyjna przyjmuje postać pary typu głównego i drugorzędnego. W eksperymencie zastosowano tylko 20 najliczniejszych typów

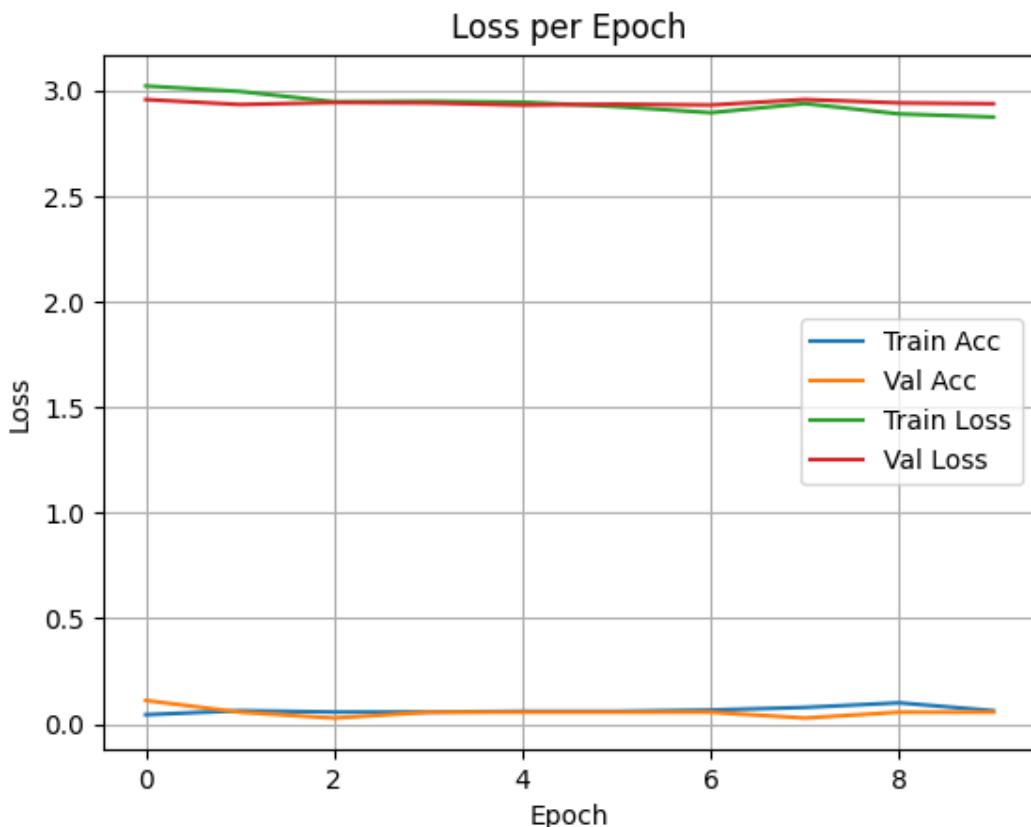
Przygotowanie danych:

- Dane zostały pobrane z oryginalnego zbioru Pokémonów (721 rekordów).
- W przypadku Pokémonów z jednym typem, jako drugi przypisano "None", tworząc pełne kombinacje dla każdej instancji.
- Wybrane cechy wejściowe: Total, HP, Attack, Defense, Sp. Atk, Sp. Def, Speed.
- Cechy zostały wystandardyzowane za pomocą StandardScaler.
- Etykiety zakodowano za pomocą LabelEncoder.
- Podział danych: 70% – trening, 30% – test.

- Użyto `class_weight='balanced'` w celu kompensacji nierównomiernych rozkładów klas.

Struktura modelu:

W tym wariantie ponownie zastosowano model z wariantu 1 (uznany za najbardziej skuteczny), czyli klasyczną sieć neuronową z funkcją aktywacji **ELU** i stałą architekturą (liczba warstw i szerokość sieci jak poprzednio).



Opis przebiegu treningu:

- **Krzywe straty (loss)** praktycznie się pokrywają – zarówno na zbiorze treningowym, jak i walidacyjnym. Ich przebieg przypomina **płaską linię** o wartości około **3.0**, z jedynie kilkoma bardzo delikatnymi załamaniami (dosłownie 3–5 punktów zmiany).
- **Dokładność (accuracy)** utrzymuje się przez cały okres treningu na bardzo niskim poziomie – **poniżej 10%**, co w przypadku 20 klas zbliża się do wartości losowej. Krzywe przebiegają praktycznie poziomo i wykazują minimalną poprawę.

Wnioski:

- Model **nie był w stanie nauczyć się sensownych reprezentacji** dla tak zróżnicowanych danych. Pomimo dużej liczby przykładów, klasyfikacja na 20 złożonych klas (kombinacje typów) okazała się zbyt trudna dla tej architektury.
- Niska dokładność oraz brak poprawy funkcji kosztu sugerują, że model **nie nauczył się żadnych istotnych zależności** między statystykami a przypisany podwójnym typem.
- Możliwe przyczyny niepowodzenia:
 - Zbyt **złożona przestrzeń etykiet (20 klas złożonych z kombinacji dwóch typów)** – niektóre z nich mogą być bardzo rzadko reprezentowane.
 - **Dodanie typu "None"** mogło wprowadzić dodatkową niejednoznaczność.
 - **Prosta architektura modelu** mogła być niewystarczająca dla uchwycenia interakcji między cechami a złożonymi klasami.
 - Wartość strat utrzymująca się na wysokim poziomie może wskazywać na **problem z optymalizacją** – być może wymagane byłoby inne podejście do reprezentacji etykiet (np. rozdzielne przewidywanie typu 1 i typu 2).

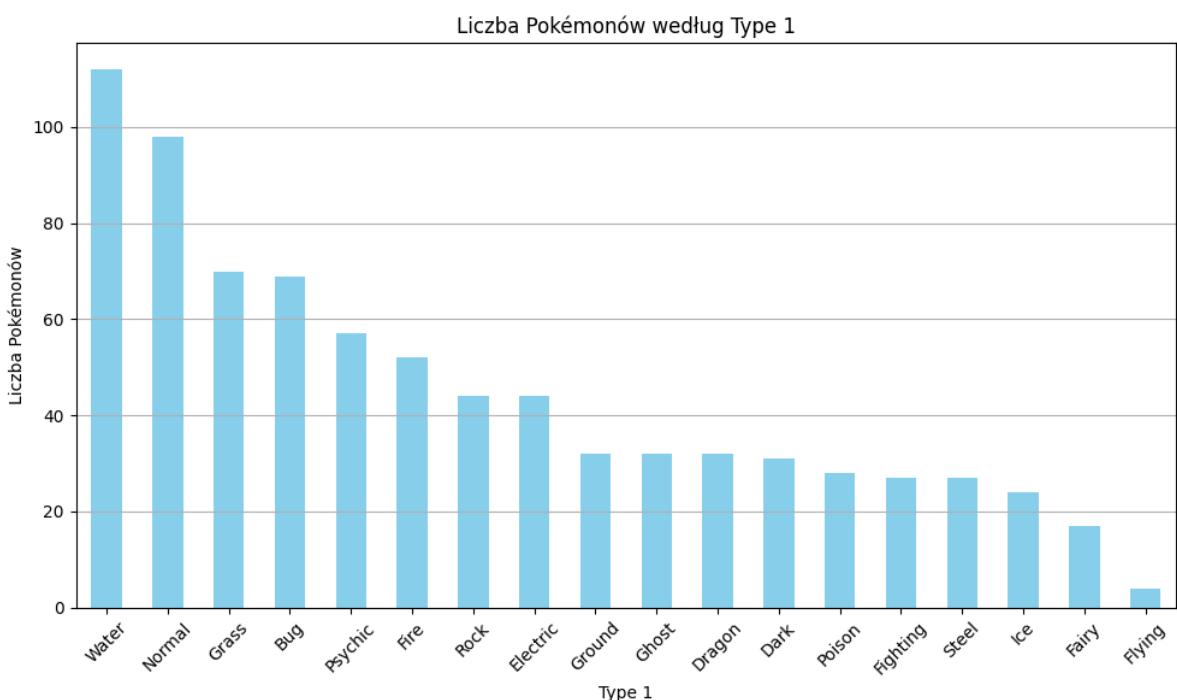
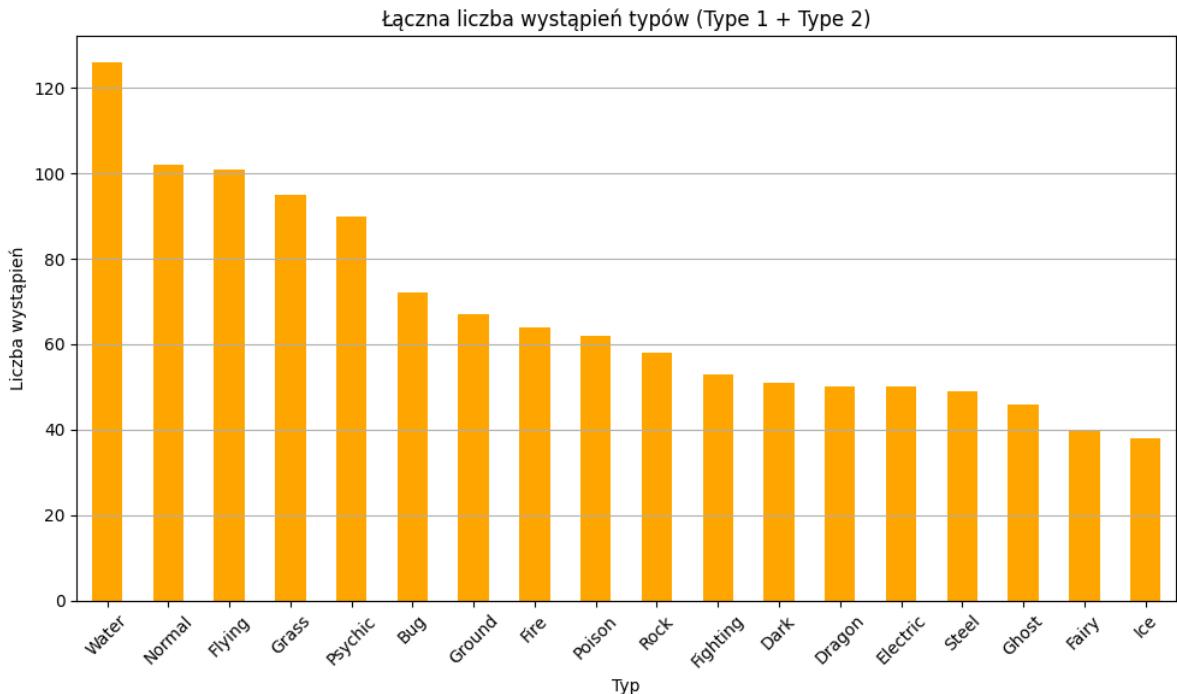
Wnioski końcowe

Pomimo wykorzystania najlepiej sprawdzającego się modelu z wariantu 1 (czyli z klasyfikacją po pierwszym typie i aktywacją ELU), wyniki osiągnięte w wariantach 2, 3 i 4 **nie były zadowalające**. W szczególności:

- **Wariant 2** (tylko Pokemony z jednym typem) – nie udało się nauczyć sensownych reprezentacji; model wykazywał oznaki przeuczenia, ale dokładność była niska.
- **Wariant 3** (tylko dual-typed) – pomimo niskiej dokładności, **krzywe straty sugerują potencjał** i możliwość dalszego ulepszania modelu.
- **Wariant 4** (wszystkie Pokemony, mono z przypisanym typem None) – model kompletnie nie poradził sobie z klasyfikacją aż 20 klas, dokładność była niemal losowa.

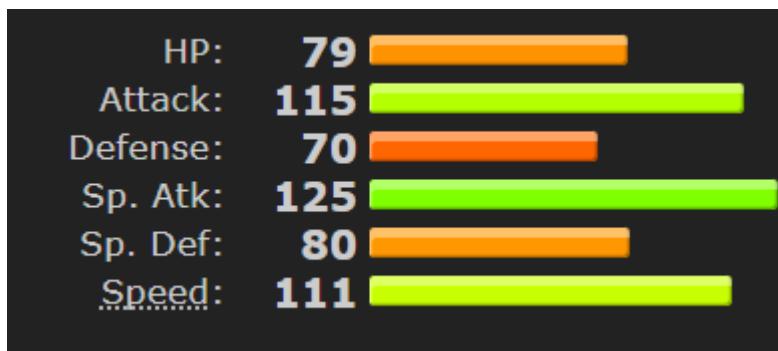
Obserwacje i potencjalne problemy:

- **Nierównomierność liczności typów:** Wśród mono-typed Pokémonów np. typ **Water** występuje ponad 120 razy, natomiast **Flying** tylko 4 razy – a w dodatku te 4 to dwie różne formy tego samego legendarnego Pokémona. Jednak gdy spojrzymy globalnie (typ 1 + typ 2), to Flying jest już **trzecim najczęstszym typem**. To pokazuje, że analizowanie tylko po pierwszym typie może prowadzić do błędnych wniosków. Poniższe wykresy przedstawiają wystąpienia typów najpierw w kombinacji a potem patrząc tylko na pierwszy typ pomaga to zwizualizować różnie w liczności typów w zależności do spojrzenia.

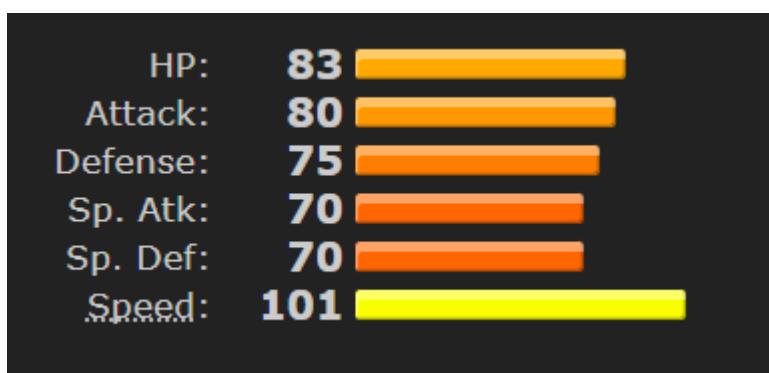


- **Wpływ Pokémonów legendarnych na wyniki:** Pokemony legendarne zazwyczaj charakteryzują się znacznie wyższymi statystykami bazowymi niż reszta, co może prowadzić do dużych odchyлеń i zakłóceń w procesie uczenia modelu. Ich obecność w zbiorze danych może zaburzać rozpoznawalne wzorce statystyczne charakterystyczne dla danej klasy typów.
- Przykład stanowią dwa Pokemony o tym samym typie głównym – **Flying**:
- **Tornadus** (legendarny) – ma wyższy współczynnik *Special Attack* niż *Attack*,
- **Pidgeot** (zwykły) – odwrotnie, *Attack* przewyższa *Special Attack*.

- Pomimo przynależności do tej samej klasy typów, znaczące różnice w profilach statystycznych mogą wprowadzać zamieszanie dla modelu uczącego się zależności między cechami a typem Pokémona. W takich przypadkach model może przypisać typ bardziej na podstawie „mocy” Pokémona niż jego rzeczywistej klasyfikacji typowej.



Statystyki Tornadusa.



Statystyki Pidgeota.

Statystyki zostały wzięte ze strony Smogon.com

- Typ jako narzędzie balansu:** Często Pokemony o bardzo silnych statystykach otrzymują typy, które dają im więcej **słabości (weaknesses)** jako formę równoważenia. Dwa Pokemony o bardzo podobnych statystykach mogą mieć kompletnie różne typy – co komplikuje naukę modelu.
- Wpływ fabularny i designowy:** Wybór typu dla Pokémona to często **decyzja oparta na fabule, klimacie czy pomyśle twórców**, a nie tylko statystykach.
Przykłady:
 - Pokémon nocny – dostaje typ **Dark**.
 - Pokémon stworzony z myślą o unikalności – np. **Lucario**, pierwszy Pokémon typu Fighting, który opiera się bardziej na **Special Attack** niż na zwykłym Attacku.

Wszystko to prowadzi do ważnej konkluzji:

Typ Pokémona to nie tylko funkcja jego statystyk, ale również wynik balansu rozgrywki, fabularnych założeń i kreatywnych decyzji twórców.

Z tego powodu, próba klasyfikacji typu wyłącznie na podstawie statystyk może być po prostu **niemożliwa do wykonania z wysoką skutecznością**.

Gdybym miał przeprowadzić eksperyment jeszcze raz...

Na pewno spróbowałbym:

- Skorzystać z **nowszego zbioru danych** – obecnie dostępne są Pokemony aż do **generacji 10**, podczas gdy mój zbiór kończył się na **gen 7**.
- Przetestować **kombinacje różnych funkcji aktywacyjnych** – np. ReLU w jednej warstwie, ELU w innej.
- Przeprowadzić testy **na pojedynczych generacjach** (np. tylko Gen 1), aby sprawdzić, czy uproszczenie domeny poprawia wyniki.
- Poświęcić na to po prostu **więcej czasu** – tuning hiperparametrów, inne architektury, augmentacja danych itd.

Weryfikat końcowy:

"Tak średnio bym powiedział."

Model z wariantu 1 był solidnym punktem wyjścia, ale ogólna jakość wyników jasno pokazuje, że **typy Pokémonów to zbyt złożony temat, by opierać klasyfikację wyłącznie na statystykach**. Potrzebne są inne cechy wejściowe lub zupełnie inne podejście.