Regularyzacja w uczeniu maszynowym

Regularyzacja to kluczowa technika ograniczająca przeuczenie modeli poprzez dodatkowe kary na wagi lub modyfikacje procesu uczenia. Poznaj najpopularniejsze metody, które pomogą Ci stworzyć bardziej stabilne i efektywne modele.

Przygotowanie danych do eksperymentów

Generowanie danych treningowych

Tworzymy prostą zależność liniową z szumem: y = 1.8x - 0.2 + szum gaussowski. Dane dzielimy na zbiór treningowy (30 próbek) i walidacyjny (10 próbek).

Ustawiamy ziarno losowe torch.manual_seed(42), aby wyniki były powtarzalne i możliwe do porównania między różnymi technikami regularyzacji.

Funkcje pomocnicze

- describe_model - wypisuje parametry i błąd MSE
- plot_regression - wizualizuje dane i dopasowanie
- Dane: 40 punktów w zakresie [-1, 1]

Regularyzacja L1

Selekcja cech przez zerowanie wag

Regularyzacja L1 dodaje do funkcji straty sumę wartości bezwzględnych wag:

$$L = L_{danych} + \lambda \sum_{i} |w_i|$$

Kara zachęca model do wyzerowania nieistotnych parametrów, co naturalnie prowadzi do selekcji cech. Współczynnik λ kontroluje siłę regularyzacji.

Regularyzacja L2

Stabilizacja przez weight decay

Regularyzacja L2 karze duże wartości wag poprzez dodanie kwadratu norm wag do funkcji straty:

$$L = L_{danych} + \lambda \sum_{i} w_i^2$$

Implementacja w PyTorch

W PyTorch wygodnie używa się parametru weight_decay w optymalizatorze, który automatycznie dodaje karę L2 i stabilizuje proces uczenia.

W przeciwieństwie do L1, regularyzacja L2 nie wyzerowuje parametrów, lecz równomiernie je zmniejsza.

Zalety L2

- Gładkie, stabilne dopasowanie
- Odporność na punkty odstające
- Zachowanie wszystkich cech
- Łatwa implementacja

Porównanie wyników L1 vs L2

Regularyzacja L1

Waga: 1.92

Bias: -0.11

MSE: 0.1231

Selekcja cech przez zerowanie nieistotnych wag

Regularyzacja L2

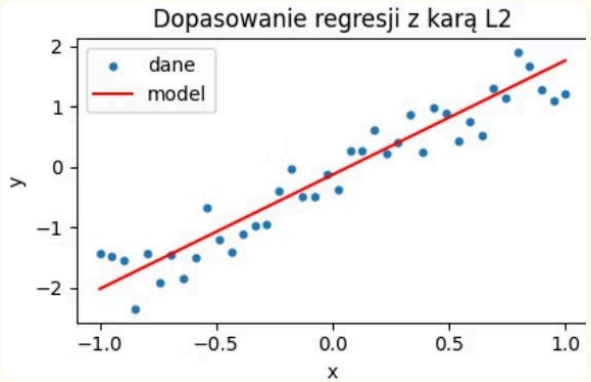
Waga: 1.89

Bias: -0.13

MSE: 0.1199

Równomierne zmniejszanie wszystkich wag

Weight decay ściąga wagi w stronę zera, dlatego linia regresji jest gładka i bardziej odporna na pojedyncze punkty odstające. L2 daje stabilne, pełne dopasowanie bez zerowania parametrów.


Dopasowanie regresji z karą L2
dane
model
2
1
0
−1
−2
y
−1.0
−0.5
0.0
0.5
1.0
x

Dropout

Losowe wyłączanie neuronów podczas treningu

Jak działa Dropout?

01

Tryb treningowy

Losowo wyłącza aktywacje neuronów (np. 50% wejść ustawia na zero), co zapobiega nadmiernemu poleganiu na pojedynczych cechach

02

Skalowanie wartości

Pozostałe aktywacje są skalowane, aby zachować oczekiwaną sumę wyjść niezależnie od losowania maski

03

Tryb ewaluacji

Warstwa przepuszcza sygnał bez zmian, wszystkie neurony są aktywne, co zapewnia stabilne predykcje

Dropout tworzy efekt zespołu modeli (ensemble), gdzie każda iteracja trenuje nieco inną architekturę sieci.

Wizualizacja działania Dropout

Dane wejściowe

```
tensor([[1., 1., 1., 1.],
        [1., 1., 1., 1.],
        [1., 1., 1., 1.]])
```

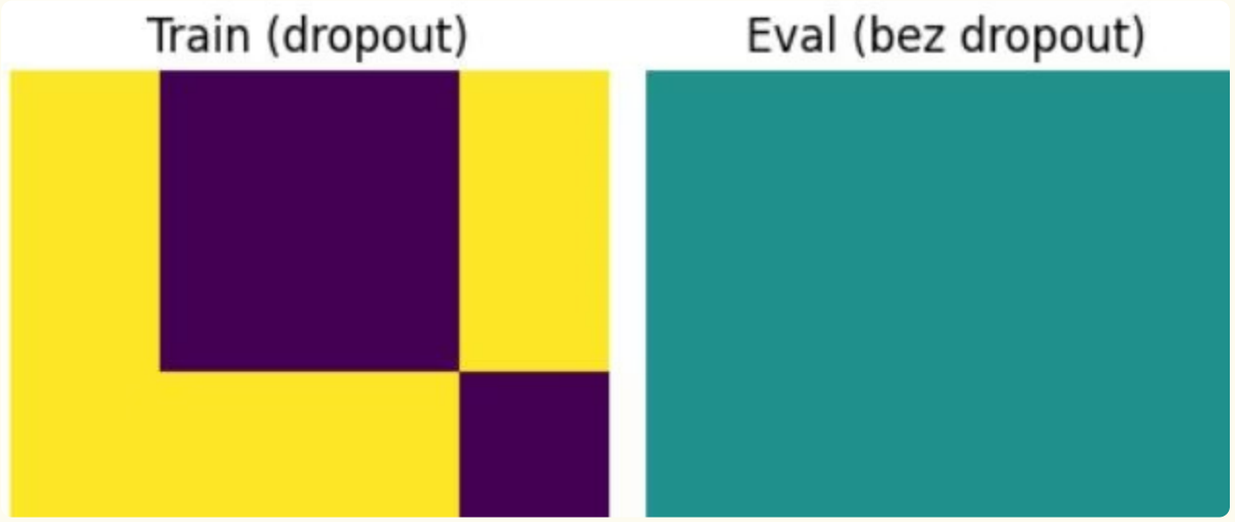
Tryb train (p=0.5)

```
tensor([[2., 0., 0., 2.],
        [2., 0., 0., 2.],
        [2., 2., 2., 0.]])
```

Tryb eval

```
tensor([[1., 1., 1., 1.],
        [1., 1., 1., 1.],
        [1., 1., 1., 1.]])
```

Podczas uczenia połowa elementów zostaje wyzerowana - sieć nie może polegać na pojedynczych aktywacjach.



Kluczowe wnioski z Dropout

Losowość w treningu

Maska dropout jest generowana losowo w każdej iteracji, co zmusza sieć do uczenia się redundantnych reprezentacji

Ochrona przed przeuczeniem

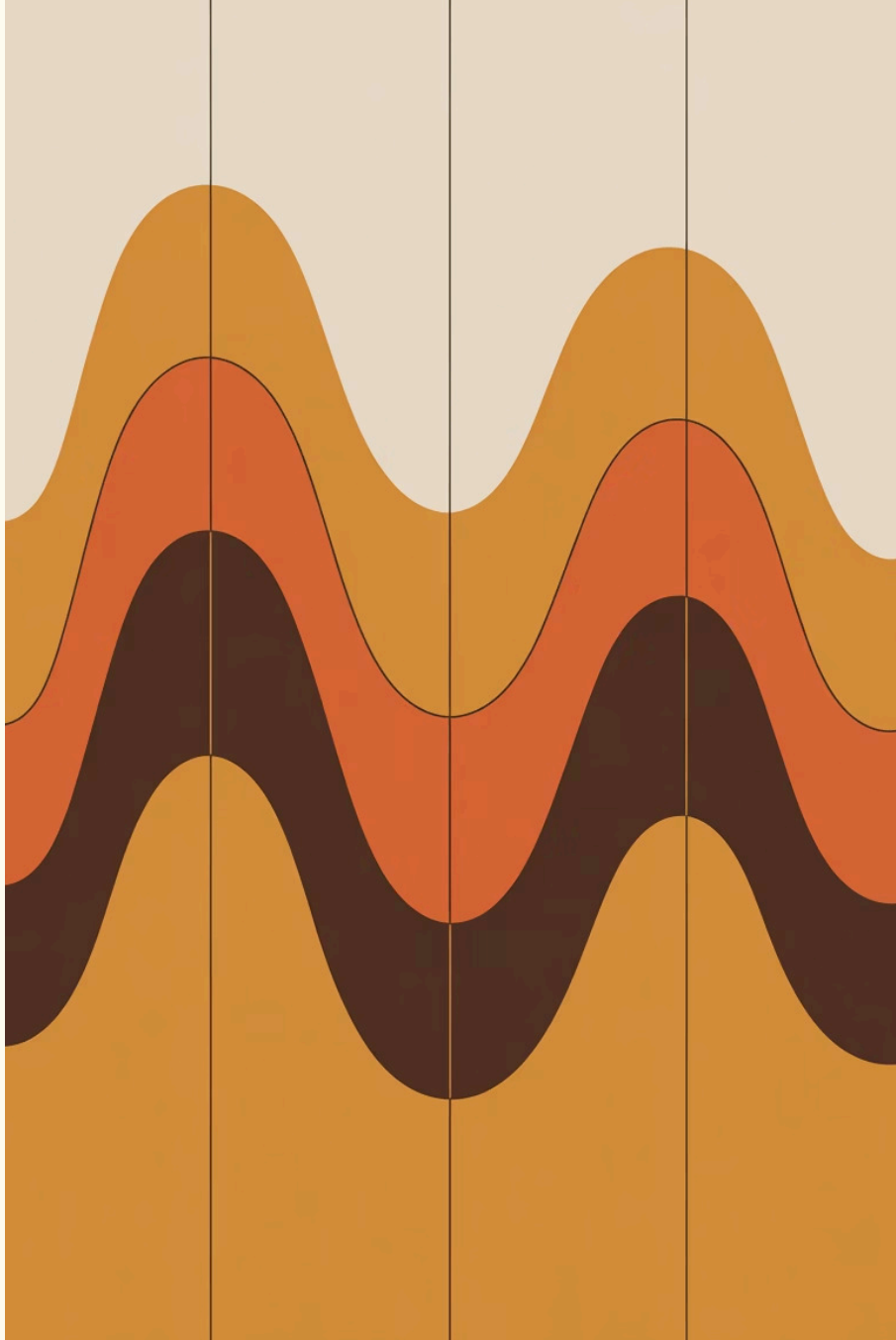
Niemożność polegania na konkretnych neuronach redukuje ko-adaptację i zwiększa generalizację modelu

Stabilne predykcje

W trybie ewaluacji wszystkie neurony są aktywne, a aktywacje skalowane dla spójności z treningiem

Early Stopping

Zatrzymanie treningu we właściwym momencie



Mechanizm Early Stopping

Early stopping monitoruje stratę walidacyjną i kończy trening, gdy brak poprawy przez określoną liczbę epok (patience). Dzięki temu model przestaje się uczyć zanim zacznie zapamiętywać szum.

Monitorowanie

Obliczanie straty walidacyjnej po każdej epoce

Zapisywanie

Przechowywanie najlepszego stanu wag modelu

Zatrzymanie

Przerwanie po wyczerpaniu patience

Przywrócenie

Powrót do najlepszych wag

Wyniki Early Stopping

1

Zatrzymanie po epoce 54

Mechanizm wykrył brak poprawy przez 3 kolejne epoki i przerwał trening

2

Najlepsza strata: 0.1276

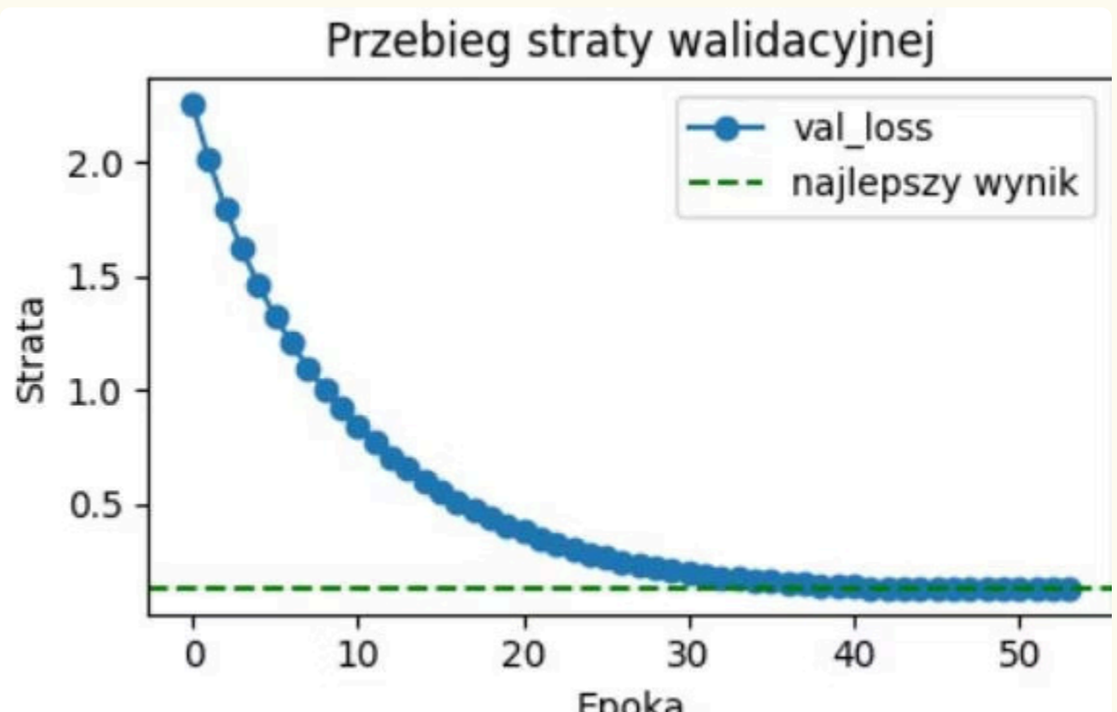
Model osiągnął minimum straty walidacyjnej w epoce 51

3

Przywrócenie wag

Finalny model używa parametrów z epoki o najniższej stracie walidacyjnej

Strata walidacyjna stabilizuje się po kilku epokach. Przywrócenie najlepszych wag zapewnia, że finalny model działa w punkcie minimalnej straty walidacyjnej, nawet jeśli ostatnia epoka była gorsza.



Parametry Early Stopping

Patience

Wartość: 3 epoki

Liczba epok bez poprawy, po której następuje zatrzymanie treningu

Tolerancja

Wartość: 1e-4

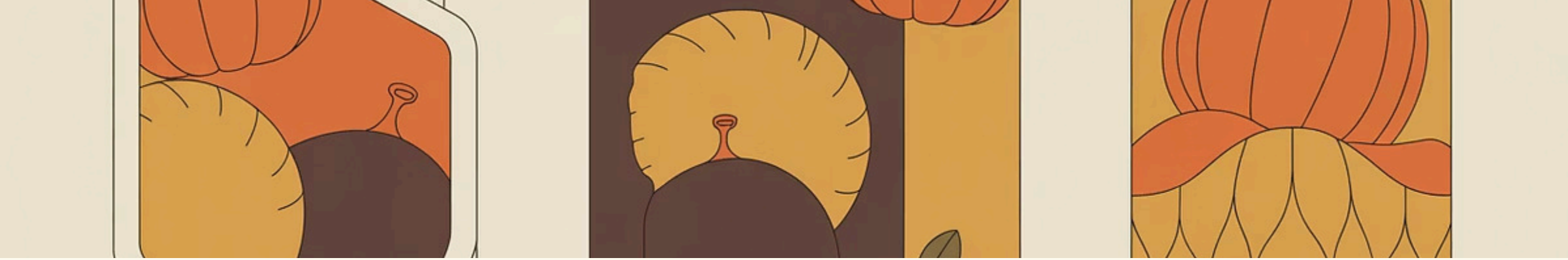
Minimalna poprawa uznawana za znaczącą przy porównywaniu strat

Maksimum epok

Wartość: 100

Górny limit iteracji, jeśli patience nie zostanie wyczerpana

Najlepsza praktyka: Zawsze przechowuj kopię najlepszych wag modelu, aby móc do nich wrócić po zakończeniu treningu. Historia strat walidacyjnych pomaga zdiagnozować problemy z uczeniem.



Augmentacja danych

Zwiększanie różnorodności zbioru treningowego

Techniki augmentacji

Augmentacja danych tworzy dodatkowe próbki poprzez modyfikację istniejących wejść. Dzięki temu model widzi więcej wariantów i uczy się cech odpornych na drobne zmiany.

Odbicie poziome

torch.flip() - lustrzane odbicie obrazu względem osi pionowej

Rotacja o 90°

torch.rot90() - obrót obrazu o wielokrotność 90 stopni

Szum gaussowski

Dodanie losowego szumu o rozkładzie normalnym do pikseli

Przykład augmentacji obrazu 4×4

Oryginał

```
tensor([[ 0.,  1.,  2.,  3.],
        [ 4.,  5.,  6.,  7.],
        [ 8.,  9., 10., 11.],
        [12., 13., 14., 15.]])
```

Rotacja o 90°

```
tensor([[ 3.,  7., 11., 15.],
        [ 2.,  6., 10., 14.],
        [ 1.,  5.,  9., 13.],
        [ 0.,  4.,  8., 12.]])
```

Odbicie poziome

```
tensor([[ 3.,  2.,  1.,  0.],
        [ 7.,  6.,  5.,  4.],
        [11., 10.,  9.,  8.],
        [15., 14., 13., 12.]])
```

Z szumem (σ=0.2)

Wartości nieznacznie zmodyfikowane przez dodanie losowego szumu gaussowskiego

Korzyści z augmentacji

Zwiększenie zbioru danych

Każda transformacja generuje nowy wariant bez zbierania dodatkowych próbek

Odporność na zmiany

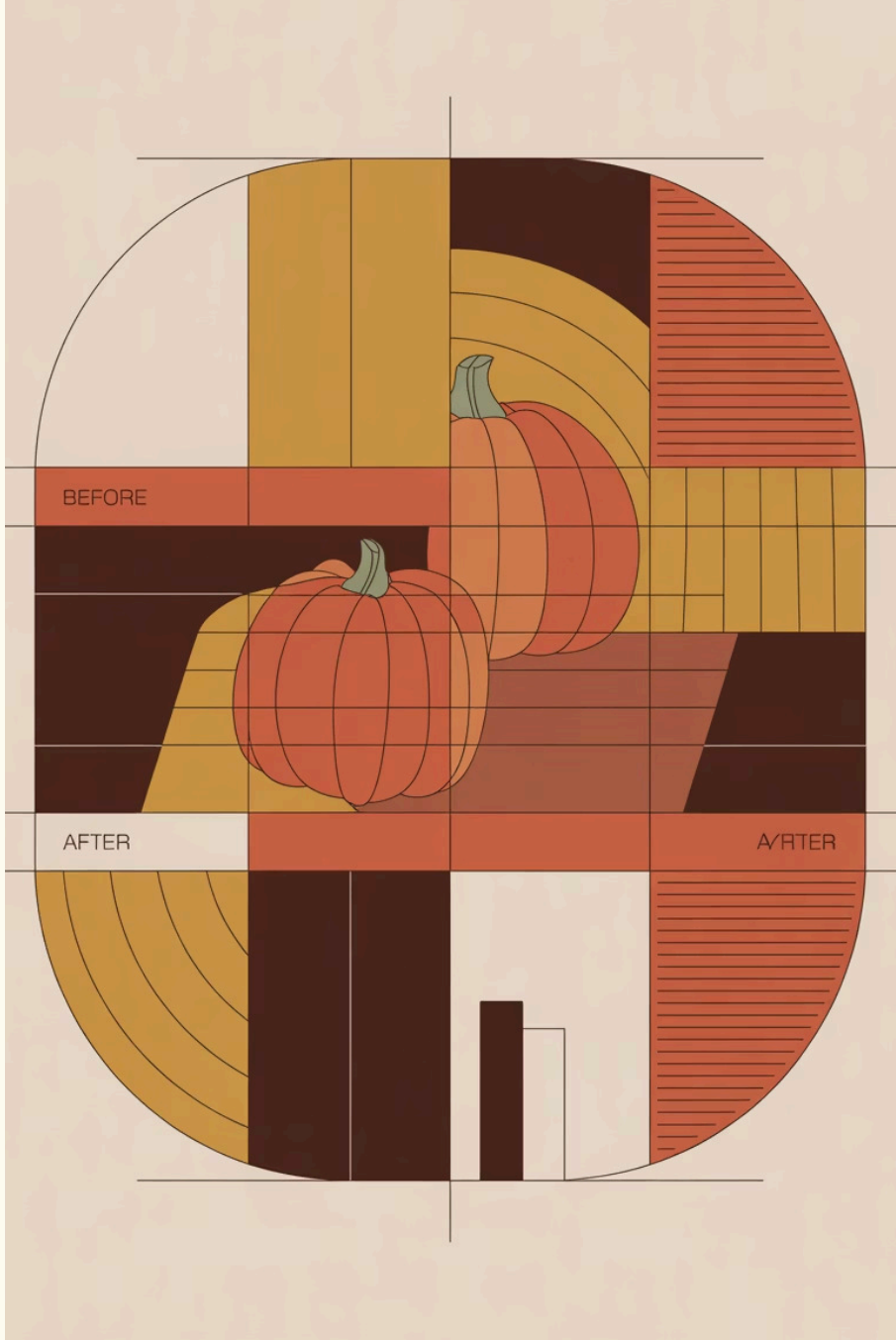
Model uczy się niezmienniczości względem transformacji geometrycznych i szumu

Regularyzacja na poziomie danych

Nawet niewielkie modyfikacje działają jak regularyzacja, chroniąc przed przeuczeniem

Lepsza generalizacja

Różnorodność treningowa przekłada się na lepsze wyniki na nowych danych



Podsumowanie technik regularyzacji

1

L1 - Selekcja cech

Zerowanie nieistotnych wag przez karę na wartości bezwzględne. Idealna do redukcji wymiarowości.

2

L2 - Stabilizacja wag

Równomierne zmniejszanie wszystkich parametrów przez weight decay. Gładkie, odporne dopasowanie.

3

Dropout - Redundancja

Losowe wyłączanie neuronów zmusza sieć do uczenia się wielu ścieżek. Efekt ensemble.

4

Early Stopping - Timing

Monitorowanie walidacji i zatrzymanie przed przeuczeniem. Proste i skuteczne.

5

Augmentacja - Różnorodność

Transformacje danych zwiększają zbiór treningowy i uczą niezmienniczości.

Wybór odpowiedniej techniki

Różne techniki regularyzacji sprawdzają się w różnych scenariuszach. Często najlepsze rezultaty daje ich kombinacja.

Małe zbiory danych

Augmentacja + Early Stopping

Głębokie sieci

Dropout + L2 (weight decay)

Uniwersalne podejście

L2 + Early Stopping

Wiele cech

L1 dla selekcji cech

Eksperymentuj z różnymi kombinacjami i monitoruj wyniki na zbiorze walidacyjnym, aby znaleźć optymalne rozwiązanie dla swojego problemu.