Raport

Jakub Kural

Wstęp

Tematem projektu jest rozpoznawanie mowy. Moim zadaniem było nauczenie modelu, który z 1-sekundowego klipu audio jest w stanie rozpoznać jedno z 10 słów kluczowych: *down*, *go*, *left*, *no*, *off*, *on*, *right*, *stop*, *up*, *yes* lub jeden z dwóch stanów specjalnych: ciszę lub nieznane wyrażenie.

Dane

W projekcie użyłem zbioru danych *speech_commands*. Dane podzielone są na 3 zbiory: *train*, *validation* oraz *test* o rozmiarach kolejno: 85511, 10102 i 4890. Zbiory *train* oraz *validation* są niezbalansowane ze znaczą przewagą nieznanych wyrażeń, lecz grupy pomiędzy zbiorami są w jednakowych proporcjach. Celem takiego zabiegu jest uzyskanie wysokiej precyzji na znanych poleceniach. Zbiór *test* jest zbalansowany.

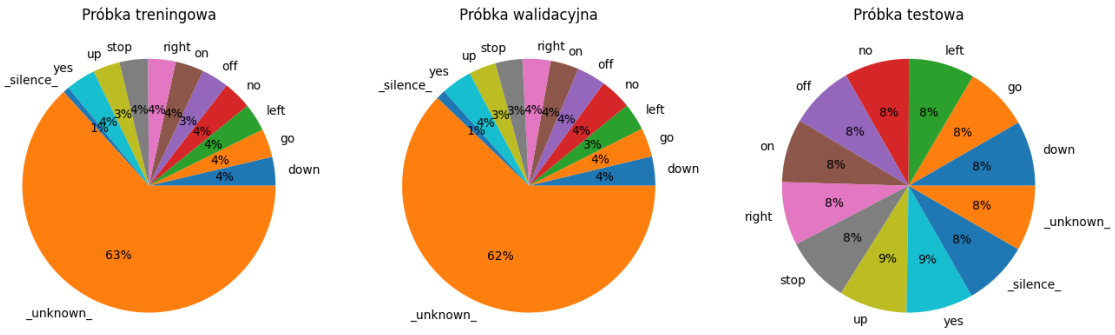
Model

Do rozwiązania problemu wykorzystałem sieć konwolucyjną. Składają się na nią warstwy:

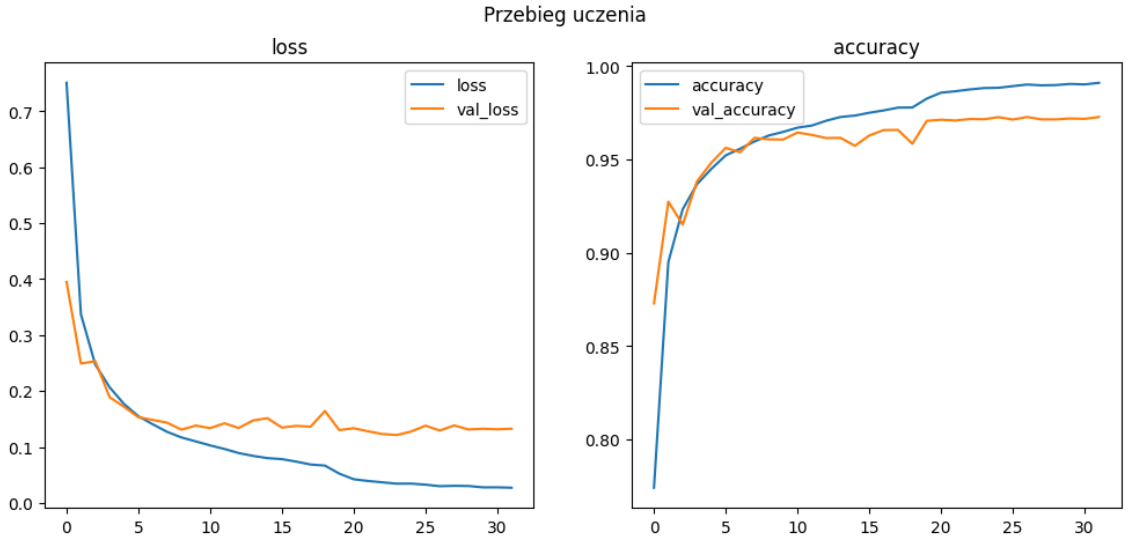
- Conv2D: 32 filtry, kernel [11, 41], stride [2, 2]
- Conv2D: 32 filtry, kernel [11, 21], stride [1, 2]
- Conv2D: 32 filtry, kernel [11, 21], stride [1, 2]
- MaxPool2D
- Dense: 128 jednostek
- Dense: 12 jednostek

Do aktywacji używana jest ReLU. Do regularyzacji używam Dropout i BatchNormalization. Jako funkcję straty wykorzystałem entropię krzyżową

Klipy audio przekształcane są na spektrogramy w skali Mela, które można traktować jako obrazy przedstawiające dźwięk, a następnie normalizowane. Użycie warstw konwolucyjnych pozwala na naukę rozpoznawania słów niezależnie od miejsca pojawienia się ich w spektrogramie, zmniejsza liczbę parametrów w porównaniu do sieci gęstej oraz wydobywa coraz bardziej szczegółowe cechy charakterystyczne spektrogramu. Pozwala to zredukować wpływ szumów oraz cech głosu osoby mówiącej na wynik przewidywania.



Rysunek 1: Podział danych



Rysunek 2: Krzywe uczenia

Wyniki modelu

dataset	accuracy	precision	recall	loss
train	0.99	0.99	0.99	0.03
validation	0.97	0.96	0.95	0.13
test	0.95	0.96	0.95	0.25

Model dobrze radzi sobie w rozpoznawaniu słów kluczowych, o czym świadczą wysokie *accuracy*, *precision* i *recall*. W wynikach można jednak zauważyć skłonność do overfittingu - metryki na zbiorze treningowym są znacznie lepsze niż na zbiorach walidacyjnym i testowym. Następnym krokiem w rozwoju projektu może być zmiejszenie liczby uczonych parametrów, co powinno zredukować overfitting i przyśpieszyć naukę. Można to osiągnąć, na przykład, poprzez zmniejszenie rozmiarów kerneli w warstwach konwolucyjnych, zwiększenie parametru *stride* lub dodanie warstw MaxPool2D pomiędzy warsty konwolucyjne.