

Metody cutout w połączeniu z soft labelingiem - badanie skuteczności

Katarzyna Rogalska, Barbara Seweryn, Zuzanna Sieńko, Urszula Szczęsna, Michał Witecki

Wstęp

Celem projektu było przeprowadzenie eksperymentów mających na celu zweryfikowanie skuteczność augmentacji danych w treningu modeli. Badaną metodą augmentacji były różne sposoby cutoutu zdjęć, czyli wycinania losowych obszarów danego obrazu w celu wprowadzenia dodatkowej trudności dla modelu. Wycinanie polegało na zastąpieniu pewnej części pikseli pikselami czarnymi lub o losowych kolorach. Ta część badania inspirowana była [opublikowanymi artykułami](#) dotyczącymi różnych metod cutoutu i ich modyfikacjach. Z dołączonych papierów wynika, że taki sposób augmentacji danych poprawia naukę modeli, stąd motywacja na zweryfikowanie tej tezy dla różnorodnych sposobów cutoutu w naszym projekcie.

Wyżej wspomniane metody połączono także z mechanizmem soft labeling zmieniającym etykiety klas proporcjonalnie do zakrytego pola. Inutycyjnie takie działanie powinno poprawić wyniki modelu na zbiorze walidacyjnym, ponieważ zostałby on lepiej nauczony rozpoznawania obrazów obraczonych szumem, zatem była to druga hipoteza, która została poddana ekspermentom w tym projekcie.

Opis metod cutout

Zaimplementowano 5 rodzajów cutoutu - RandomPixelsCutout, RandomSquaresCutout, SquareCutout, CircleCutout, PolygonCutout oraz zmianę wejściowych etykiet na Soft Labels skalowane proporcjonalnie do powierzchni niezamaskowanego obszaru obrazu. Wszystkie metody cutoutu umożliwiają konfigurację rozmiaru i koloru wycięcia.

Opis klas implementujących cutout

RandomPixelsCutout

Klasa **RandomPixelsCutout** implementuje metodę losowego zakrywania pikseli obrazu. Transformacja ta polega na losowym wyborze pikseli w obrazie i nadpisaniu ich wartością czarną lub losowym kolorem – w zależności od ustawienia parametru color.

Parametry klasy:

- **max_cutout_size** (*float*): Maksymalny procent pikseli, które mogą zostać zakryte (wartość z zakresu 0–1).
- **color** (*bool*): Jeśli **False**, zakryte piksele przyjmują kolor czarny (0,0,0). Jeśli **True**, każdy zakryty piksel otrzymuje losowy kolor RGB.

Metoda **__call__**:

1. Obliczana jest losowa liczba pikseli do zakrycia – maksymalnie **max_cutout_size * liczba_pikseli**.
2. Wybierane są losowe współrzędne (x, y) dla tych pikseli.

3. W zależności od ustawienia `color`, piksele są nadpisywane kolorem czarnym lub losowymi wartościami RGB.
4. Obliczana jest nowa wartość etykiety (*soft label*).

Transformacja zwraca nowy obraz oraz zaktualizowaną etykietę.



`RandomPixelsCutout, max_cutout_size = 0.3,
color = False`

`RandomPixelsCutout, max_cutout_size = 0.3,
color = True`

RandomSquaresCutout

Klasa `RandomSquaresCutout` implementuje wariant metody cutoutu polegający na losowym zakrywaniu obrazu kwadratowymi obszarami o jednakowym rozmiarze.

Parametry klasy:

- `max_number_of_squares` (*int*): Maksymalna liczba kwadratów, które mogą zostać nałożone na obraz.
- `max_size_ratio` (*float*): Maksymalny stosunek długości boku kwadratu do wymiaru obrazu (wartość z zakresu 0–0.1). Wyższe wartości są niedozwolone i spowodują zgłoszenie wyjątku.
- `color` (*bool*): Określa sposób zakrycia. Jeśli `False`, piksele w obrębie wycinka zostaną zamienione na czarne (0,0,0). Jeśli `True`, każdy piksel wewnątrz kwadratu otrzyma losowy kolor RGB.

Metoda `__call__`:

1. Losowany jest rozmiar kwadratu na podstawie proporcji `max_size_ratio`.
2. Dla każdego z `max_number_of_squares` generowane są losowe współrzędne lewego górnego rogu, w taki sposób, by kwadrat mieścił się w granicach obrazu.
3. Kwadratowe obszary są nakładane na obraz i wypełniane odpowiednim kolorem.
4. Obliczana jest nowa wartość etykiety (*soft label*).

Transformacja zwraca zmodyfikowany obraz oraz nową etykietę.



```
RandomSquaresCutout, max_number_of_squares  
= 10, max_size_ratio = 0.1, color = False
```

RandomSquaresCutout, max_number_of_squares = 50, max_size_ratio = 0.1, color = True

SquareCutout

Klasa `SquareCutout` implementuje metodę zakrywania jednego, kwadratowego obszaru obrazu o ustalonym rozmiarze. W odróżnieniu od wersji losowej (`RandomSquaresCutout`), ta metoda zawsze aplikuje dokładnie jedno wycięcie, którego rozmiar określany jest bezpośrednio przez użytkownika.

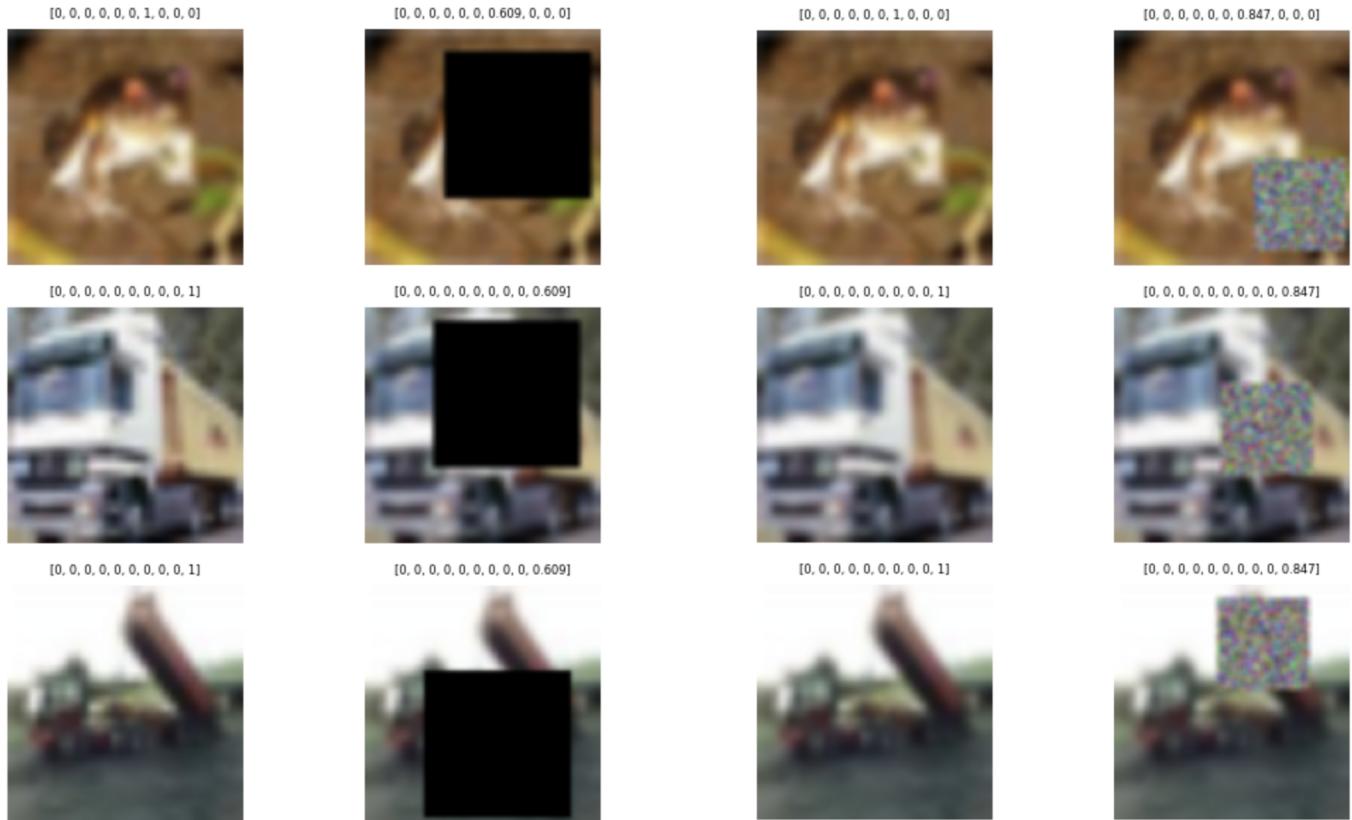
Parametry klasy:

- **size** (*int*): Rozmiar wycinanego kwadratu w pikselach (bok kwadratu). Wartość ta musi być mniejsza niż najmniejszy z wymiarów obrazu.
 - **color** (*bool*): Określa sposób zakrycia. Jeśli *False*, piksele w obrębie wycinka zostaną zamienione na czarne (0,0,0). Jeśli *True*, każdy piksel wewnętrz kwadratu otrzyma losowy kolor RGB.

Metoda call:

1. Sprawdzana jest poprawność rozmiaru wycinka względem wymiarów obrazu – w przypadku błędu zgłoszany jest wyjątek.
 2. Losowo wybierane są współrzędne lewego górnego rogu kwadratu.
 3. Kwadratowy obszar o zadanym rozmiarze jest nadpisywany kolorem czarnym lub losowymi wartościami RGB.
 4. Na podstawie liczby zmodyfikowanych pikseli (czyli `size2`) obliczana jest nowa etykieta (*soft label*).

Transformacja zwraca zmodyfikowany obraz oraz zaktualizowaną etykietę.



RandomSquareCutout, size = 80, color = False

RandomSquareCutout, size = 50, color = True

CircleCutout

Klasa `CircleCutout` implementuje metodę zakrywania obrazu za pomocą pojedynczego kołowego wycięcia o zmiennym promieniu.

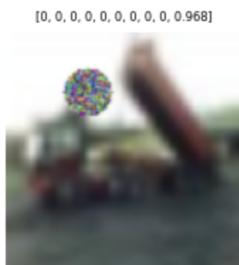
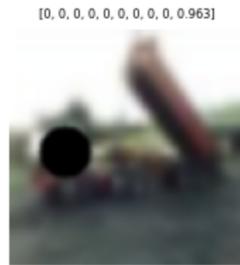
Parametry klasy:

- `radius` (`int` lub `None`): Promień kołowego wycięcia w pikselach. Jeśli `None`, promień jest wybierany losowo z zakresu od 5 do wartości wyliczonej na podstawie rozmiarów obrazu i `max_size_ratio`.
- `max_size_ratio` (`float`): Maksymalny rozmiar wycięcia wyrażony jako ułamek mniejszego wymiaru obrazu (wartość z zakresu 0–1). Promień koła nie może przekroczyć połowy tej wartości.
- `color` (`tuple RGB` lub `None`): Kolor wycięcia. Domyślnie czarny (`0,0,0`), jeśli nie zostanie podany.
- `random_color` (`bool`): Jeśli `True`, piksele wewnętrznych koła otrzymują losowe kolory RGB zamiast jednolitego koloru.

Metoda `__call__`:

1. Ustala promień wycięcia na podstawie parametru `radius` lub losowo w dopuszczalnym zakresie.
2. Losowo wybiera środek koła tak, aby mieściło się ono w obrębie obrazu.
3. Tworzy maskę pikseli należących do koła według równania okręgu.
4. Nakłada maskę na obraz, zamieniając piksele na czarne lub losowe kolory RGB.
5. Oblicza nową etykietę (*soft label*).

Transformacja zwraca zmodyfikowany obraz oraz zaktualizowaną etykietę.



CircleCutout, max_size_ratio = 0.3, color = False

CircleCutout, max_size_ratio = 0.3, color = True

PolygonCutout

Klasa `PolygonCutout` implementuje metodę zakrywania obrazu za pomocą wycięcia w kształcie wielokąta o losowej liczbie wierzchołków.

Parametry klasy:

- `max_vertices` (`int`): Maksymalna liczba wierzchołków wielokąta (domyślnie 12).
- `min_vertices` (`int`): Minimalna liczba wierzchołków wielokąta (domyślnie 3, czyli trójkąt).
- `max_size_ratio` (`float`): Maksymalny rozmiar wycięcia wyrażony jako ułamek mniejszego wymiaru obrazu (wartość z zakresu 0–1).
- `color` (`tuple` RGB lub `None`): Kolor wycięcia. Domyślnie czarny (`0, 0, 0`), jeśli nie zostanie podany.
- `random_color` (`bool`): Jeśli `True`, piksele wewnętrznych wielokąta otrzymują losowe kolory RGB zamiast jednolitego koloru.

Metoda `__call__`:

1. Losuje liczbę wierzchołków wielokąta z zakresu od `min_vertices` do `max_vertices`.
2. Losowo wyznacza środek wielokąta oraz promień bazowy na podstawie `max_size_ratio`.
3. Generuje punkty wierzchołków wielokąta, rozmieszczone wokół środka z losowym przesunięciem kąta i odległości.
4. Sprawdza dla każdego piksela, czy znajduje się wewnątrz wielokąta, wykorzystując algorytm ray-casting.
5. Piksele znajdujące się wewnątrz wielokąta są zamieniane na wybrany kolor lub losowe kolory RGB.
6. Oblicza nową etykietę (`soft label`).

Transformacja zwraca zmodyfikowany obraz oraz zaktualizowaną etykietę.



PolygonCutout, max_size_ratio = 0.3,
max_vertices = 12, min_vertices = 8, color = False

PolygonCutout, max_size_ratio = 0.3,
max_vertices = 12, min_vertices = 8, color = True

SoftLabelsDataset

Klasa `SoftLabelDataset` rozszerza istniejący zbiór danych o dodatkową możliwość transformacji obrazów za pomocą cutoutu oraz modyfikacji etykiet na *soft labels*.

Parametry klasy:

- `dataset`: oryginalny zbiór danych, który ma być przetwarzany.
- `pipeline_before_cutout`: sekwencja transformacji wykonywanych na obrazach przed zastosowaniem cutoutu.
- `pipeline_after_cutout`: sekwencja transformacji wykonywanych po cutoucie.
- `num_classes (int)`: liczba klas w zbiorze danych, używana do kodowania one-hot etykiet.
- `cutout_transform`: obiekt transformacji wykonujący cutout. Jeśli `None`, to cutout nie jest wykonywany, a zbiór danych nie jest rozszerzany.

Metody:

- `__len__`: zwraca rozmiar zbioru danych. Jeśli dostępny jest transformator cutoutu, zwraca podwójną długość oryginalnego zbioru (oryginalne + przekształcone obrazy).
- `__getitem__`: dla danego indeksu zwraca parę (`obraz, etykieta`).
 - Dla indeksów odpowiadających transformacjom cutoutu, obraz jest przetwarzany przez `cutout_transform`, a etykieta jest modyfikowana na *soft label*.
 - W przeciwnym wypadku zwraca oryginalny obraz i etykietę (zakodowaną one-hot).
 - Transformacje zdefiniowane w `pipeline_before_cutout` są wykonywane zawsze przed cutoutem, a `pipeline_after_cutout` zawsze po nim (lub zamiast, jeśli cutout nie jest stosowany).

Transformacja zwraca przetworzony obraz oraz odpowiednio zmodyfikowaną etykietę.

Opis wykorzystanych danych

W ramach projektu wykorzystano dwa zestawy danych obrazowych: **CIFAR-10** oraz **Fashion-MNIST**. Oba zbiorы należą do najczęściej stosowanych benchmarków w zadaniach klasyfikacji obrazów.

CIFAR-10

Zbiór danych **CIFAR-10** zawiera **60 000 kolorowych obrazów RGB** o wymiarach **32×32 piksele**, podzielonych na **10 klas**:

- samolot
- samochód
- ptak
- kot
- jeleń
- pies
- żaba
- koń
- statek
- ciężarówka

Podział zbioru:

- **50 000 obrazów treningowych**
- **10 000 obrazów testowych**

Fashion-MNIST

Zbiór danych **Fashion-MNIST** to nowoczesna alternatywa dla klasycznego MNIST. Zawiera **obrazy przedstawiające elementy odzieży** w odcieniach szarości o wymiarach **28×28 pikseli**, które są podzielone na **10 klas ubrań i akcesoriów**:

- T-shirt/top
- Spodnie
- Sweter
- Sukienka
- Płaszcz
- Sandały
- Koszula
- Sneakersy
- Torba
- Botki

Podział zbioru:

- **60 000 obrazów treningowych**
- **10 000 obrazów testowych**

Opis modeli

Model bazowy

W eksperymencie jako model bazowy wykorzystano konwolucyjną sieć neuronową (CNN), zaprojektowaną do klasyfikacji obrazów wejściowych o takim samym kształcie jak próbki w zbiorze `X_train`. Liczba neuronów w warstwie wyjściowej (`k`) odpowiada liczbie klas i została ustalona na podstawie kształtu etykiet treningowych (`k = Y_train.shape[1]`). Architektura modelu została zbudowana w oparciu o klasyczne podejście z warstwami konwolucyjnymi, normalizacją i warstwami gęstymi.

Szczegóły architektury:

1. Wejście:

- Obraz wejściowy o kształcie zgodnym z próbki w `X_train`.

2. Pierwszy blok konwolucyjny:

- `Conv2D(32, (3, 3), activation='relu', padding='same')`
- `BatchNormalization()`
- `Conv2D(32, (3, 3), activation='relu', padding='same')`
- `BatchNormalization()`
- `MaxPooling2D((2, 2))`

3. Drugi blok konwolucyjny:

- `Conv2D(64, (3, 3), activation='relu', padding='same')`
- `BatchNormalization()`
- `Conv2D(64, (3, 3), activation='relu', padding='same')`
- `BatchNormalization()`
- `MaxPooling2D((2, 2))`

4. Trzeci blok konwolucyjny:

- `Conv2D(128, (3, 3), activation='relu', padding='same')`
- `BatchNormalization()`
- `Conv2D(128, (3, 3), activation='relu', padding='same')`
- `BatchNormalization()`
- `MaxPooling2D((2, 2))`

5. Warstwy gęste:

- `Flatten()`
- `Dropout(0.25)`
- `Dense(1024, activation='relu')`
- `Dropout(0.25)`

6. Warstwa wyjściowa:

- `Dense(k, activation='softmax')`

Wyniki modelu bazowego dla zbioru CIFAR-10

Model bazowy osiągnął następujące wyniki po 6 epokach treningu:

- **Accuracy (zbiór treningowy):** 0.8461
- **Accuracy (zbiór walidacyjny):** 0.8033

Wyniki modelu bazowego dla zbioru Fashion-MNIST

Model bazowy osiągnął następujące wyniki po 6 epokach treningu:

- **Accuracy (zbiór treningowy):** 0.9446
- **Accuracy (zbiór walidacyjny):** 0.9276

Wyniki te stanowiły za punkt odniesienia dla oceny wpływu technik **Cutout** oraz **soft labels** na jakość klasyfikacji. W kolejnych sekcjach przedstawiono, w jaki sposób te metody wpływają na skuteczność modelu w kontekście obu zbiorów danych.

Analiza wyników

Wyniki eksperymentów z podziałem na zbiory danych zostały przedstawione w formie tabelarycznej poniżej.

Zbiór danych: CIFAR

Cutout Type	Max Size / Param	Soft / Normal	Val Acc	Train Acc
base model	-	-	0.8033	0.8461
Polygon	0.2	Normal	0.8023	0.8661
Polygon	0.1	Normal	0.7989	0.8687
Polygon	0.05	Normal	0.7833	0.8705
Polygon	0.05	Soft	0.7709	0.8715
Pixels	0.1	Normal	0.8010	0.8619
Pixels	0.05	Normal	0.8123	0.8656
Pixels	0.05	Soft	0.4791	0.5017
Squares	25, ratio = 0.1	Normal	0.7601	0.8584
Squares	10, ratio = 0.1	Normal	0.8055	0.8627
Squares	10, ratio = 0.05	Normal	0.8135	0.8637
Squares	10, ratio = 0.05	Soft	0.7912	0.8646
Square	5	Normal	0.8119	0.8747
Square	10	Normal	0.8010	0.8633
Square	5	Soft	0.7460	0.8117

Dla powyższego zbioru **obniżenie** accuracy na zbiorze treningowym w porównaniu do modelu bazowego zostało zanotowane tylko w przypadku:

- Pixels(0.05, soft)
- Square(5, soft)

Jeśli chodzi o zbiór walidacyjny, **obniżenie** accuracy zaobserwowane został dla:

- Pixels(0.05, soft)
- Squares(25, 0.1, normal)
- Squares(10, 0.05, soft)
- Square(5, normal)

Eksperyment, który wypadł zdecydowanie najgorzej to cutout Pixels(0.05, soft) z accuracy treningowym na poziomie zaledwie 0.5017, a walidacyjnym 0.4791. Jest to bardzo niski i niepokojący wynik, który w porównaniu z innymi eksperymentami z soft labels, wypada drastycznie gorzej.

Zbiór danych: Fashion MNIST

Cutout Type	Max Size / Param	Soft / Normal	Val Acc	Train Acc
base model	-	-	0.9276	0.9446
Polygon	0.2	Normal	0.9238	0.9464
Polygon	0.01	Normal	0.9274	0.9461
Polygon	0.05	Normal	0.9064	0.9472
Polygon	0.05	Soft	0.9193	0.9473
Pixels	0.1	Normal	0.9244	0.9471
Pixels	0.05	Normal	0.9265	0.9481
Pixels	0.05	Soft	0.7149	0.6800
Squares	25, ratio = 0.1	Normal	0.9200	0.9451
Squares	10, ratio = 0.1	Normal	0.9269	0.9476
Squares	10, ratio = 0.05	Normal	0.7253	0.8099
Squares	10, ratio = 0.05	Soft	0.7701	0.7879
Square	5	Normal	0.9262	0.9471
Square	10	Normal	0.9211	0.9475
Square	5	Soft	0.8063	0.7243

Dla zbioru Fashion MNIST **obniżenie** accuracy na zbiorze treningowym w porównaniu do modelu bazowego zostało zanotowane w przypadku:

- Pixels(0.05, soft)
- Squares(10, 0.05, normal)
- Squares(10, 0.05, soft)
- Square(5, soft)

Jeśli chodzi o zbiór walidacyjny, obniżenie accuracy zaobserwowane zostało **dla każdej z technik.**

W obydwu eksperymentach cutout Pixels(0.05, soft) spowodowało obniżenie accuracy zarówno na zbiorze treningowym jak i walidacyjnym.

Dodatkowe eksperymenty i testy

Opisane powyżej wyniki jasno wskazują, że zastosowana augmentacja w znacznej większości przypadków nie tylko nie poprawia wyników, lecz je dodatkowo pogarsza. Największym zaskoczeniem jednak był fakt, że wpływ użycia **soft labels** jest tak negatywny:

W zbiorze CIFAR zmienienie etykiet skutkowało każdorazowym obniżeniem accuracy, przy porównaniu do odpowiadającego mu - stosującego ten sam cutout z identycznymi parametrami. Największy spadek zanotowano dla Pixels(0.05, soft), gdzie wynik jest prawie dwukrotnie niższy.

Patrząc na zbiór Fashion Mnist wyniki walidacji modelów wspomaganych augmentacją są w każdym przypadku gorsze niż model bazowy. Jeśli chodzi o zachowanie rezultatów po zmianie etykiet to sytuacja jest trochę inna niż w przypadku poprzedniego zbioru. Modele z soft labels w dwóch przypadkach pogarszają, a w dwóch przypadkach polepszają wyniki w porównaniu z ich odpowiadającymi modelami. Jednak obniżenia są zdecydowanie bardziej znaczące - średni wzrost accuracy walidacji w tym przypadku to 2,9%, podczas gdy średni spadek to aż 16,6%. Można więc zatem uznać, że działanie soft labels nie jest pozytywne.

Odpowiedź na pytanie dlaczego tak się dzieje nie jest znana. Etykiety obrazów są zmieniane o małą frakcją, co nie powinno, aż tak pogarszać przewidywań modelów. Jednak możliwe, że nawet takie zmiany nie pozwalają na poprawne działanie tego typu sieci neuronowych. Szukając odpowiedzi na to pytanie, zostały wyróżnione dwa prawdopodobne scenariusze, które mogą tak wpływać na stosowaną konwolucyjną sieć neuronową:

- Występowanie niedokładności numerycznych przy liczeniu nowej etykiety opartej na wyciętym/zakrytym polu
- Etykiety jednak są zmniejszane o zbyt dużą liczbę i model z tym sobie nie radzi

Eksperiment: Soft Labels nie zależą od wyciętego pola.

Aby zweryfikować prawdziwość tych teorii przeprowadzony został eksperiment mający na celu wytrenowanie modeli korzystając ze stałych labeli, zmniejszanych o znacznie mniejsze wartości. Przebadano to obydwie te teorie naraz, nie pozwalając na wyżej wspomniane niedokładności numeryczne, przy jednoczesnym pomniejszaniu etykiet o niezbyt duże wartości.

Każda etykieta obrazu objętego augmentacją, jest zamieniana na normalną etykietę [0 0 ... 0 1 0 ... 0] gdzie zamiast wartości jeden będzie początkowo wartość 0.9999. Następnie te same eksperymenty zostaną przeprowadzone dla 0.999 oraz 0.99 w celu zobaczenia czy dalsze obniżanie wartości w soft labels przekłada się na pogorszenie wyników.

Wyniki eksperymentów z podziałami na zbiory danych przedstawiono poniżej:

Zbiór danych: CIFAR

Cutout	Soft label	Max size	Acc Val + (old)	Acc train + (old)
Random Squares	0.9999	10, ratio =0.05	0.7859 (0.7912)	0.8684 (0.8646)
Random Squares	0.999	10, ratio =0.05	0.7937 (0.7912)	0.8653 (0.8646)
Random Squares	0.99	10, ratio =0.05	0.8161 (0.7912)	0.8646 (0.8646)
Square	0.9999	5	0.8048 (0.7460)	0.8686 (0.8117)
Square	0.999	5	0.7845 (0.7460)	0.8686 (0.8117)
Square	0.99	5	0.8146 (0.7460)	0.8660 (0.8117)
Random Pixels	0.9999	0.05	0.8087 (0.4791)	0.8609 (0.5017)
Random Pixels	0.999	0.05	0.8065 (0.4791)	0.8669 (0.5017)
Random Pixels	0.99	0.05	0.8172 (0.4791)	0.8686 (0.5017)
Polygon	0.9999	0.05	0.8082 (0.7709)	0.8679 (0.8715)
Polygon	0.999	0.05	0.8044 (0.7709)	0.8727 (0.8715)
Polygon	0.99	0.05	0.8149 (0.7709)	0.8716 (0.8715)

Dla powyższego zbioru, accuracy zbioru walidacyjnego na stale ustalonych soft labels jest w większości przypadków wyższe, niż dla etykiet zależących od pola wycięcia.

Warto zauważenia jest fakt, że dla wycięcia, które sprawiało największe problemy - Random Pixels(0.05, soft) - udało się znacznie poprawić wynik i wyrównać go z wynikami innych modeli. Można wnioskować, że w tym przypadku, któraś z postawionych tez rzeczywiście powodowała obniżenie rezultatów.

Ciekawe również jest to, że dla każdego kolejnego wycięcia najlepszy wyniki otrzymuje się dla miękkich etykiet ustalonych na wartość 0.99. Zmiananie etykiet o mniejsze wartości niż pola wycięte rzeczywiście poprawiły rezultaty, chociaż powyższy trend idzie w drugą stronę. Możliwe, że w eksperymencie nie zostały uwzględnione wystarczające duże wartości, które hipotetycznie doprowadziłyby do obniżenia tych wyników. Możliwa jest również trzecia przyczyna poprzednich gorszych wyników:

Stabilność wartości etykiet sama w sobie zapewnia lepsze rezultaty.

Zbiór danych: Fashion MNIST

Cutout	Soft label	Max size	Acc Val + (old)	Acc train + (old)
Random Squares	0.9999	10, ratio =0.05	0.9291 (0.7701)	0.9501 (0.7879)
Random Squares	0.999	10, ratio =0.05	0.9267 (0.7701)	0.9476 (0.7879)
Random Squares	0.99	10, ratio =0.05	0.8445 (0.7701)	0.8548 (0.7879)
Square	0.9999	5	0.9262 (0.8063)	0.9505 (0.7243)

Cutout	Soft label	Max size	Acc Val + (old)	Acc train + (old)
Square	0.999	5	0.9267 (0.8063)	0.9491 (0.7243)
Square	0.99	5	0.8494 (0.8063)	0.8568 (0.7243)
Random Pixels	0.9999	0.05	0.9237 (0.7149)	0.9470 (0.6800)
Random Pixels	0.999	0.05	0.9226 (0.7149)	0.9460 (0.6800)
Random Pixels	0.99	0.05	0.8349 (0.7149)	0.8355 (0.6800)
Polygon	0.9999	0.05	0.9232 (0.9193)	0.9481 (0.9473)
Polygon	0.999	0.05	0.9033 (0.9193)	0.9479 (0.9473)
Polygon	0.99	0.05	0.8416 (0.9193)	0.8582 (0.9473)

Z kolei dla zbioru Fashion MNIST wyniki zdają się jednak potwierdzać tezę o zbyt dużym pomniejszaniu wartości etykiet. W każdym modelu accuracy na podzbiorze walidacyjnym jest ewidentnie gorsza dla 0.99 niż dla 0.999/0.9999.

Mogłoby to wynikać z różnic w wielkościach zbiorów - dla większego objętościowo zbioru Fashion MNIST obniżenie wartości etykiet do 0.99 już robi różnicę, podczas gdy dla mniejszego zbioru CIFAR ta różnica byłaby widoczna dopiero po kolejnym obniżeniu wartości etykiety. Hipoteza ta miałaby podstawę w tym, że przy zbiorze z większą liczbą obrazów treningowych model automatycznie patrzy więcej razy na soft label. To z kolei prowadzi do sytuacji, że mniejsza zmiana w wartości soft labelu wywoła większą różnicę w działaniu modelu.

Następnie porównanie modeli korzystających z testowanych etykiet versus modeli z labelami zależącymi od pola. Jedynie dla cutout Polygon modele ze stałymi labelami nie wyszły lepiej, chociaż i tutaj różnica nie jest znacząca. Oczywiście, jak poprzednio, znaczy to znaczną poprawę w działaniu Random Pixels.

Ostatnim krokiem eksperymentu była próba zmodyfikowania klas cutoutu tak, aby miękkie etykiety zależące od pola wyciętej powierzchni były zaokrąglone do kilku miejsc po przecinku. Zgodnie z powyższymi eksperymentami wyeliminowałoby to hipotezę o problemach numerycznych związanych z dużą częścią dziesiętną. Po przetrenowaniu kilka próbnych modeli nie zaobserwowano jednak poprawy względem początkowo zaprezentowanej tabeli, dlatego na tym etapie zakończono dalsze badania tego zjawiska.

Podsumowanie

W przeprowadzonym eksperymencie udało się zaimplementować 5 różnych metod augmentacji obrazów typu Cutout. Każda z metod przyjmowała parametry określające, czy wycięta część obrazu ma być czarna czy kolorowa oraz czy podczas przeprowadzenia augmentacji etykieta klasy ma zostać zmodyfikowana zgodnie z ideą Soft Labelingu. Działanie wspomnianych metod przetestowano na 2 zbiorach danych trenując sieć neuronową w różnych konfiguracjach łącznie 30 razy. W pierwszej kolejności zwrócono uwagę na fakt dość nieznacznych zmian w wynikach treningu oraz walidacji po zastosowaniu metod Cutoutu bez zmiany etykiet. Rzadko kiedy wartości accuracy były gorsze niż w modelu bazowym - zazwyczaj augmentacja poprawiała wyniki o kilka setnych procenta lub wcale na nie nie wpływała. W dalszej analizie napotkano anomalie związaną ze znacznym pogorszeniem accuracy modeli uczonych na danych z miękkimi etykietami, co poddano dalszym testom i eksperymentom w celu odnalezienia przyczyny takiego zjawiska. Finalnie nie udało się jednoznacznie odpowiedzieć na pytanie, dlaczego Cutout w połączeniu z Soft Labeling daje tak niestabilne

wyniki z różnicami względem modelu bazowego dochodzącymi czasem nawet do 20 punktów procentowych. Możliwe przyczyny tego zjawiska, które udało się sformułować to zaszyte w klasie sieci neuronowej problemy numeryczne lub stabilność całkowitych etykiet sama w sobie zapewniająca lepsze rezultaty. Na tym etapie projekt został zakończony. W przyszłości rozwój badania mógłby być nastawiony na więcej eksperymentów, użycie modeli z kilkukrotnym wywoływaniem jednej konfiguracji oraz dogłębną diagnostyką napotkanej anomalii związanej z miękkimi etykietami, które intuicyjnie powinny pozytywnie wpływać na testowanie modeli.

Bibliografia

1. ["Colorful Cutout"](#)
2. ["Benefits of Cutout and Cutmix"](#)