1 Ekstrakcja cech z obrazów

1.1 Opis działania programu

Pierwszym etapem programu, po wczytaniu odpowiednich plików, jest wstępne przetwarzanie obrazu. Obraz zostaje zamieniony na czarno-biały, następuje usuwanie szumów oraz przeprowadzona zostaje dylatacja obrazu. Następnie uruchamiane są wszystkie funkcje przetwarzające. Wyniki funkcji przetwarzających zostają zapisane do pliku w formacie *json*.

1.2 Wykrywanie liścia

Funkcja *get_leaf* wyszukuje liść na obrazie. Na początku zostają wyodrębnione regiony obrazu, których powierzchnia jest większa niż 1000. Ma to na celu usunięcie wszelkich artefaktów. Następnie zostaje wybrany obiekt, który leży najbliżej środka - większość liści znajduje się na środku zdjęcia. Funkcja zwraca informacje o powierzchni liścia, liczbę pixeli wypukłego wielokąta pokrywającego obiekt oraz średnicę koła o takiej samej liczbie pixeli jak obiekt. Są to cechy uzyskiwane dzięki *skimage.measure.regionprobs*.

1.3 Sposób wyliczania cech

1. Powierzchnia ogonka liścia
   Na początek zostaje przeprowadzona erozja obiektu w celu otrzymania liścia bez ogonka. Następnie liść bez ogonka jest odejmowany od liścia z ogonkiem w celu uzyskania samego ogonka liścia. Kolejny etap polega na odtworzeniu oryginalnej wielkości ogonka, co zostało przeprowadzone przy wykorzystaniu otwarcia binarnego. Funkcja zwraca największą powierzchnię - czyli powierzchnię ogonka liścia.

2. Obwód obiektu
   Funkcja używa otwarcia binarnego w celu usunięcia ogonka liścia. Ogonek jest usuwany, aby sztucznie nie zawyżać długości obwodu (małe liście mogą mieć długie ogonki). Funkcja zwraca sumę liczby pixeli w obwodzie obiektu.

3. Szkielet obrazu
   W tym etapie zostaje zastosowana wbudowana funkcja, która zwraca sumę liczby pixeli w szkielecie.

4. Długość dłuższej osi obiektu na obrazie
   Na początek zostało zastosowane otwarcie binarne w celu usunięcia ogonka. Brak usunięcia ogonka może zaburzać uzyskiwany wynik, tak samo jak w przypadku obwodu liścia. Zwracana jest najdłuższa oś największego obiektu na obrazie.

5. Długość krótszej osi obiektu na obrazie
   Na początek zostało zastosowane otwarcie binarne w celu usunięcia ogonka. Brak usunięcia ogonka może zaburzać uzyskiwany wynik, tak samo jak w przypadku obwodu liścia. Zwracana jest najkrótsza oś największego obiektu na obrazie.

6. Odległość każdego punktu leżącego na konturze obiektu od jego środka
   Na początek zostało zastosowane otwarcie binarne (w celu usunięcia ogonka) oraz znajdowanie konturu obiektu. Zostają wyodrębnione regiony obrazu, których powierzchnia jest większa niż 1000. Następnie zostają wyodrębnione największe regiony obrazu. Zwracana jest średnia odległość wszystkich punktów konturu od środka.

1.4 Uzasadnienie wyboru cech

Cechy, które zostały wybrane są najbardziej intuicyjnym wyborem przy przetwarzaniu liści. Uzasadnienie poszczególnych cech:

- obwód obiektu daje nam pogląd na kształt liścia, ponieważ jest on od niego zależny. Im bardzie postrzępiony, nieregularny obiekt tym obwód jest większy,
- szkielet obiektu został wybrany z nadzieją, że będzie pokrywał się z unerwieniem liścia,
- długość dłuższej oraz krótszej osi obiektu na obrazie może wskazywać, czy liść jest wydłużony, czy spłaszczony,
- średnia odległość wszystkich punktów leżących na konturze obiektu od jego środka, również wskazuje mniej więcej jaki jest kształt liścia,
- powierzchnia ogonka liścia oraz powierzchnia liścia są intuicyjnym wyborem.

2 Wybór i uczenie klasyfikatorów

2.1 Eksperymenty obliczeniowe - domyślne parametry oraz wszystkie cechy

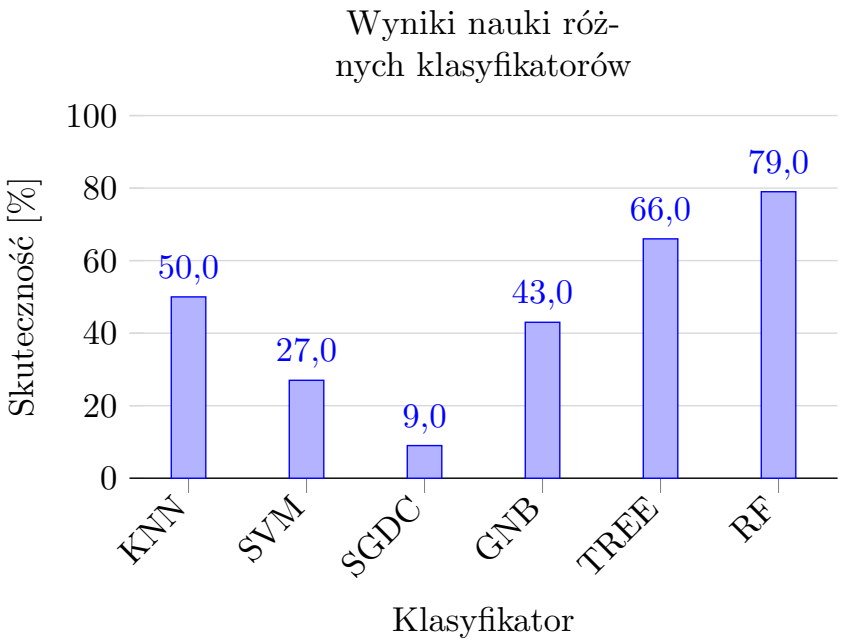
Pierwszy etap polegał na sprawdzeniu jak domyślne parametry klasyfikatorów oraz wszystkie cechy będą wpływać na jakość rozpoznawania gatunku liścia. Do nauki użyto następujących klasyfikatorów z biblioteki *scikit-learn*:

1. algorytm k najbliższych sąsiadów (*KNeighborsClassifier*, *KNN*),
2. maszyna wektorów nośnych (ang. *C-Support Vector Classification*, *SVM*),
3. gradient prosty (*SGDClassifier*, *SGDC*),
4. naiwny klasyfikator bayesowski (*GaussianNB*, *GNB*),
5. drzewa decyzyjne (*DecisionTreeClassifier*, *TREE*),
6. losowy las decyzyjny (*RandomForestClassifier*, *RF*).

W celu ocenienia jakości nauki zbiór wejściowy został wymieszany, a następnie podzielono go na dwa podzbiory:

- uczący, złożony z 700 liści,
- sprawdzający, złożony ze 100 liści.

Wyniki nauki zostały przedstawione na wykresie 1. Każdy słupek stanowi średnią z 100 iteracji nauki klasyfikatora i sprawdzenia błędów. Można zaobserwować, że najgorzej poradził sobie klasyfikator *SGDC*, a najlepiej - klasyfikator *RF*. Ze względu na fakt, że wynik klasyfikatora *RF* znacząco odbiega od wyników pozostałych klasyfikatorów zdecydowano, że tylko jego parametry będą testowane i polepszane.



Rysunek 1: Wykres przedstawiający jakość uzyskiwanych rozwiązań w zależności od danego klasyfikatora.

2.2 Eksperymenty obliczeniowe - testowanie cech

W ramach tego testu sprawdzono, który parametry są informatywne, a które należy usunąć, ze względu na małą różnorodność. Spróbowano następujących rozwiązań:

- usunięcie cech, które mają niską wariancję - będą to cechy nieinformatywne, gdyż nie wpłyną na klasyfikację - żadna cecha nie została usunięta,
- użycie testu χ^2 do wybrania najlepszych czterech cech - niestety nie można użyć tego testu, gdyż w danych znajdują się wartości ujemne, a ten test wymaga wartości dodatnich,
- estymatora opartego na drzewie (*ExtraTreesClassifier*) pozwalają na odrzucenie nieistotnych cech - żadna cecha nie została usunięta.

Na podstawie wyników powyższych metod pozostawiono wszystkie cechy dla każdego liścia. Rozmiar danych się nie zmienił i wynosi 800 wierszy oraz 9 kolumn. Przyjmując założenie, że powyższe metody zostały użyte poprawnie, można założyć, że cechy obliczone w pierwszym etapie zostały dobrze wybrane oraz poprawnie obliczone.

2.3 Eksperymenty obliczeniowe - testowanie parametrów

Jak już wspomniano wcześniej, na obecnym etapie zostaną przetestowane parametry tylko losowego lasu decyzyjnego ze względu na bardzo dobre wyniki podstawowe. Niektóre artykuły [1] wskazują, że jest to dobra decyzja, gdyż w tego typu projektach właśnie ten klasyfikator radzi sobie najlepiej. Następujące parametry zostały poddane testowaniu:

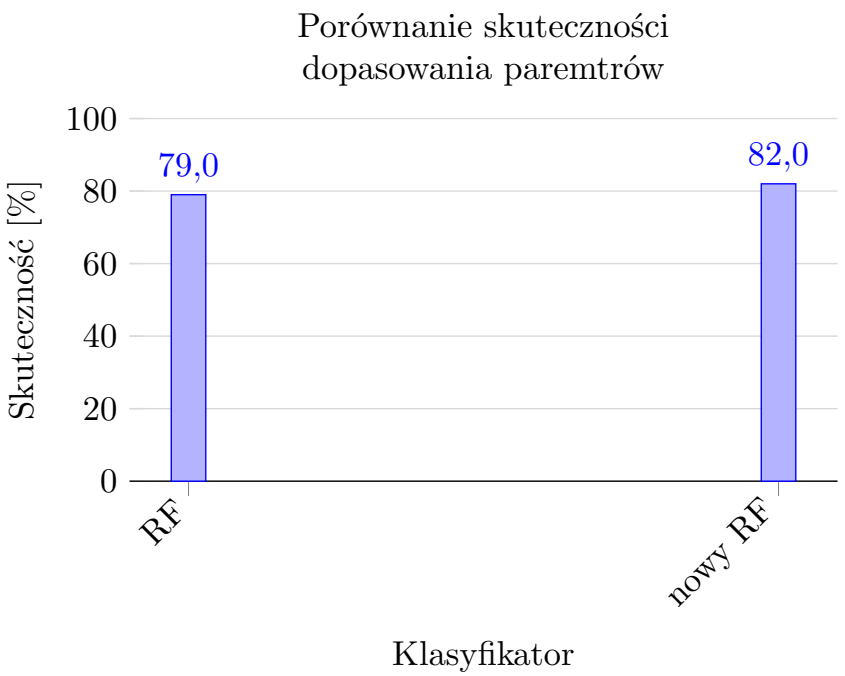
- *n_estimators* - liczba drzew w lesie,
- *max_features* - liczba cech uwzględniana podczas poszukiwania najlepszego podziału,
- *max_depth* - maksymalna liczba liści w drzewie,
- *min_samples_split* - minimalna liczba próbek potrzebna do podziału węzła,

- *min_samples_leaf* - minimalna liczba próbek potrzebna, aby uznać je za węzeł,
- *bootstrap* - czy metoda boostrap ma zostać użyta do wyboru próbek.

Pierwszy etap polegał na użyciu funkcji *RandomizedSearchCV*, która pozwalała na losowanie parametrów z podanych zakresów. Taka decyzja jest podyktowana faktem, że sprawdzenie wszystkich możliwych kombinacji parametrów byłoby bardzo czasochłonne. Dzięki użyciu tej metody można losowo sprawdzić parametry, a następnie wybrać te, które prawdopodobnie najbardziej wpływają na uzyskiwane wyniki. Uzyskane wartości dla parametrów zostały przedstawione w tabeli 1. Następnie przetestowano klasyfikator, w celu sprawdzenia, czy z takimi parametrami wyniki są rzeczywiście lepsze, co zostało przedstawione na wykresie 2. Uzyskane wyniki dla „nowego RF" (klasyfikatora z dopasowanymi parametrami) są nieznacznie (o 3 punkty procentowe) lepsze, w stosunku do domyślnych parametrów.

n_estimators	1000
max_features	sqrt
max_depth	20
min_samples_split	2
min_samples_leaf	1
boostrap	True

Tablica 1: Uzyskane najlepsze losowe parametry.



Rysunek 2: Wykres przedstawiający porównanie między klasyfikatorami bez dopasowanych parametrów z dopasowanymi parametrami.

Następny etap polegał na sprawdzeniu wszystkich możliwych kombinacji parametrów z danego zakresu w celu uzyskania rzeczywiście jak najlepszych parametrów. W tabeli 2 przedstawiono wybrane wartości dla poszczególnych parametrów. Natomiast w tabeli 3 przedstawiono uzyskane najlepsze parametry zwrócone przez funkcję *GridSearchCV*. Porównując tabelę 1 z tabelą 3 można stwierdzić, że losowe parametry były bardzo zbliżone do najlepszych parametrów. Jedyna różnica była w liczbie drzew, która ostatecznie jest niższa, co wpływa na czas nauki.

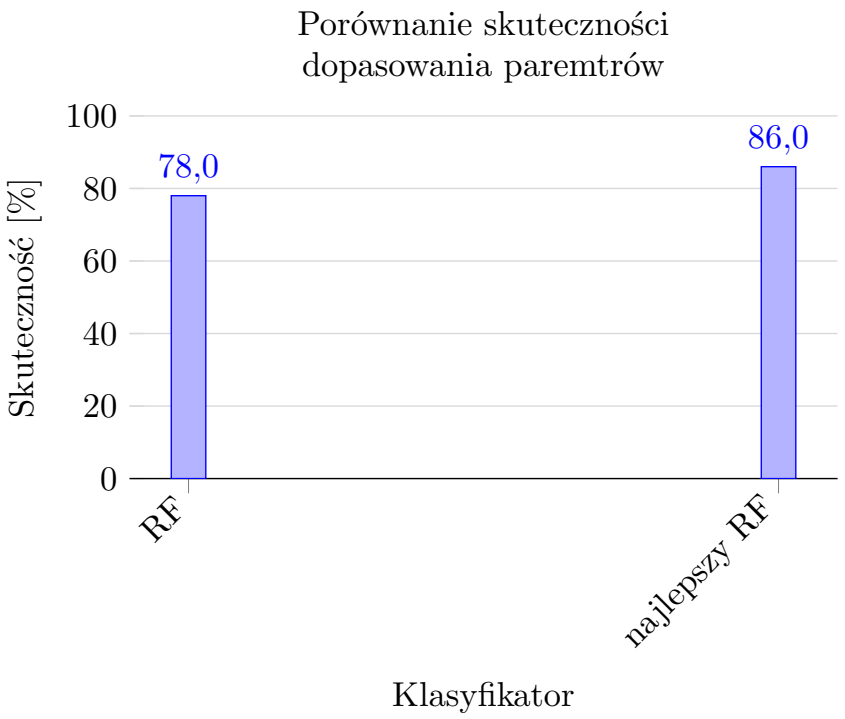
n_estimators	500, 1000, 1500, 2000
max_features	sqrt
max_depth	10, 20, 40, 100
min_samples_split	2, 4, 6
min_samples_leaf	1, 2, 3
boostrap	True

Tablica 2: Wartości parametrów użyte do kombinacji.

n_estimators	500
max_features	sqrt
max_depth	20
min_samples_split	2
min_samples_leaf	1
boostrap	True

Tablica 3: Najlepsze wartości parametrów zwrócone przez funkcję *GridSearchCV*.

Ostatni etap polegał na porównaniu wyników klasyfikatora z domyślnymi parametrami oraz wyników klasyfikatora z najlepszymi parametrami. Wynik został przedstawiony na wykresie 3. Można zauważyć, że wybrane parametry wpłynęły na uzyskane wyniki i taki klasyfikator jest o 8 punktów procentowych lepszy od klasyfikatora z domyślnymi parametrami. Tak uzyskany klasyfikator został zapisany do pliku z wykorzystaniem funkcji *dump*.



Rysunek 3: Wykres przedstawiający porównanie między klasyfikatorami bez dopasowanych parametrów z najlepszymi parametrami.

2.4 Program klasyfikujący

Do działania programu klasyfikującego jest potrzebny plik *clf_file.pkl* - do niego został zapisany klasyfikator. Program działa w następujący sposób: w linii komend oprócz nazwy skryptu należy podać ścieżkę do folderu zawierającego obrazy (folder nie może być zagnieżdżony). Program najpierw wczytuje wszystkie obrazy, następnie poddaje je procesowi ekstrakcji cech, a następnie klasyfikator przewiduje gatunek liścia dla każdego obrazu. Wynik działania programu zostaje wyświetlony w linii poleceń w formacie: nazwa_pliku.jpg gatunek.

Literatura

[1] Leaf classification project. [on-line]
`https://inside-techlabs.medium.com/leaf-classification-project-89bb5c7577f3`.