Narzędzia do stylometrii (analiza stylu artykułów – analiza Autorstwa)

Cel projektu

Celem projektu, było stworzenie funkcji oraz przetestowanie ich działania w celu analizy tekstu pod względem użytego stylu. Analiza stylu dzieła jest w stanie nam udzielić informacji o autorze tekstu, przez co możemy wykryć czy dane dzieło zostało stworzone przez autora lub nawet czy fragment został stworzony przez kogoś innego. Mechanizm taki można wykorzystać w wszelakiego rodzaju oprogramowaniu służącym do wykrywania plagiatów oraz niesamodzielnej pracy.

Opis rozwiązania

Projekt można podzielić na 3, różne części w których zostały użyte różne funkcje służące do analizy podobieństwa bądź różnic w stylu autorów:

- imposters
- stylo
- rolling.classify

Imposters

Funkcja wykorzystująca klasyfikator nadzorowany przez uczenie maszynowe dostosowany do oceny autorstwa. Klasyfikator bazuje na najczęściej występujących słowach. Funkcja ta zwraca wartość określającą podobieństwo. Na gruncie teoretycznym każdy wynik powyżej 0,5 sugerowałby, że weryfikacja autorstwa danego kandydata przebiegła pomyślnie. Jednak wyniki takie jak 0,39 lub 0,63 należy uznać za podejrzane: wydają się sugerować, że klasyfikator miał problemy z podejmowaniem jednoznacznych decyzji.

Funkcja imposters bazuje na macierzy w której, wierszami są tytuły utworów, a kolumny to cechy (najczęściej słowa), wartości w macierzy to częstotliwości występowania danej cechy.

```
> print(galbraith)
                       the       and       to        of        a         was
coben_breaker        3.5921994 1.1751078 2.1627238 1.3757359 2.5185999 1.5023227
coben_dropshot       3.5878361 1.1785435 2.1221613 1.2685983 2.3753589 1.5674759
coben_fadeaway       3.9313925 1.4454976 2.2004062 1.2130445 2.3064771 1.3225006
coben_falsemove      3.625411  1.6133386 2.1335329 1.2366884 2.4009913 1.3753251
coben_goneforgood    3.8340306 1.816723  2.152941  1.1758076 1.961908  1.7326685
coben_nosecondchance 4.0982934 1.5889666 2.2712554 1.2058633 1.9921373 1.7577145
coben_tellnoone      4.1015556 1.7901359 2.0306373 1.2463421 2.1763598 1.4181288
galbraith_cuckoos    4.5230275 2.267404  2.4940058 2.1793966 2.1412831 1.6555098
lewis_battle         5.0507132 3.4045379 2.1384253 2.1384253 1.9598416 1.5110928
```

Początkowo funkcja imposters została przetestowana na spreparowanych danych, które zawierały 3000 najczęściej używanych słów z 26 książek, 5 autorów: Galbraith,

Rowling, Cobenm, Tolkien, Lewis. Z takiego zboru została wybrana książka Drużyna Pierścienia - Tolkiena i porównana do pozostałych 25 książek, aby stwierdzić podobieństwo tej książki do pozostałych.

```
> imposters(reference.set=galbraith[-c(24),], test=galbraith[24,])
No candidate set specified; testing the following classes (one at a time):
  coben   galbraith   lewis   rowling   tolkien


Testing a given candidate against imposters...

coben    0
galbraith        0
lewis    0.16
rowling          0
tolkien          1
    coben galbraith     lewis   rowling   tolkien
     0.00      0.00      0.16      0.00      1.00
```

Wynik jaki otrzymaliśmy jest dokładnie taki jaki się spodzielwaliśmy, czyli podobieństwo 1.0 (według algorytmu - praktycznie równe pewności) do autora powieści. Istnieje niewielkie podobieństwo do C.S. Lewis jest jest ono na tyle małe, że nie można stwierdzić tutaj jakiejkolwiek współpracy podczas tworzenia.

Kolejnym krokiem było przetestowania dostępnych metod obliczania dystansu, czyli mówiąc dokładniej to określenie podobieństwa między tekstami. Dostępne są algorytmy:

- Burrow's Delta (podstawowy)
- Argamon's Linear Delta
- Eder's Delta
- Eder's Simple Distance
- Canberra Distance
- Manhattan Distance
- Euclidean Distance
- Cosine Distance

```
imposters(reference.set=galbraith[-c(24),], test=galbraith[24,])
imposters(reference.set=galbraith[-c(24),], test=galbraith[24,], distance = "delta")
imposters(reference.set=galbraith[-c(24),], test=galbraith[24,], distance = "wurzburg")
imposters(reference.set=galbraith[-c(24),], test=galbraith[24,], distance = "minmax")
imposters(reference.set=galbraith[-c(24),], test=galbraith[24,], distance = "argamon")
imposters(reference.set=galbraith[-c(24),], test=galbraith[24,], distance = "eder")
imposters(reference.set=galbraith[-c(24),], test=galbraith[24,], distance = "manhattan")
imposters(reference.set=galbraith[-c(24),], test=galbraith[24,], distance = "euclidean")
imposters(reference.set=galbraith[-c(24),], test=galbraith[24,], distance = "cosine")
```

W naszym przypadku nie ma większych różnic, między algorytmami, ale w przypadku minmax czas obliczeń wydłuża się z kilku sekund do kilku minut.

```
    coben galbraith     lewis   rowling   tolkien
     0.00      0.00      0.24      0.00      1.00
> # Cosine Delta (aka Wurzburg Distance)
> imposters(reference.set=galbraith[-c(24),], test=galbraith[24,], distance = "wurzburg")
```

```
    coben galbraith     lewis   rowling   tolkien
      0.0       0.0       0.2       0.0       1.0
> # Ruzicka Distance (aka Minmax Distance)
```

Kolejnym etapem było wczytanie własnego korpusu. Aby dokonać analizy własnych utworów, należy najpierw wczytać tekst oraz przekształcić go w tokeny. Następnie pobrać najczęściej występujące tokeny. Ostatnim krokiem jest połączenie wyników w jeden macierz.

```
tokenized.texts = load.corpus.and.parse(files = "all")
features = make.frequency.list(tokenized.texts, head = 2000)
data = make.table.of.frequencies(tokenized.texts, features, relative = TRUE)
```

Wczytane zostały 6 książek 2 autorów: George R. R. Martin (Clash Of Kings, A Game of Thrones), William Shakespeare (Macbeth, Hamlet, Romeo And Juliet, King Lear).

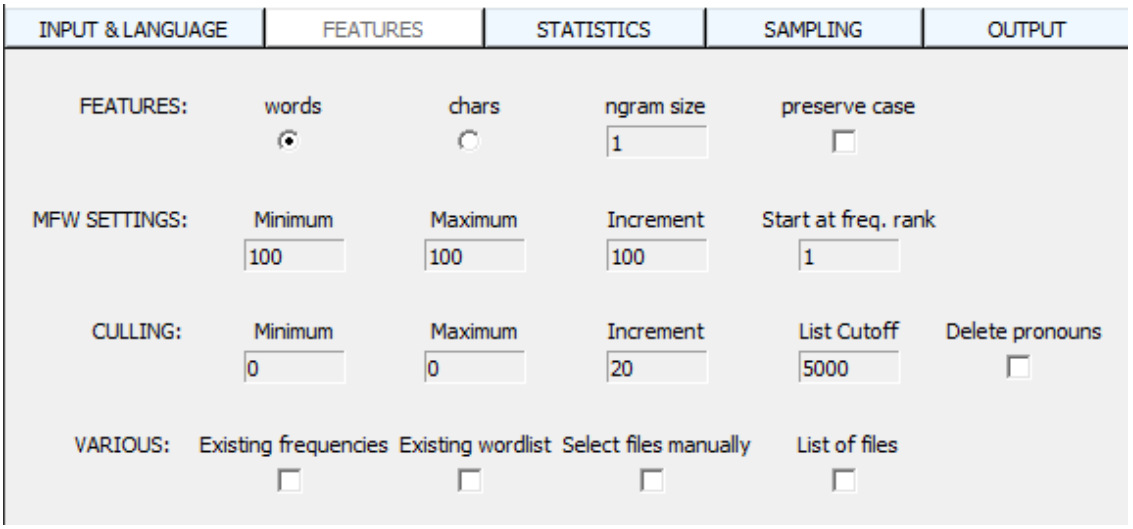
Dokonujemy analizę autorstwa książki Clash Of Kings w porównaniu do pozostałych książek. Wynik jest taki jaki oczekiwaliśmy czyli podobieństwo 1.0 do drugiej książki tego samego autora.

```
 game-of-thrones          hamlet       king-lear         macbeth romeo-and-juliet
            1.00            0.23            0.19            0.21            0.00
```

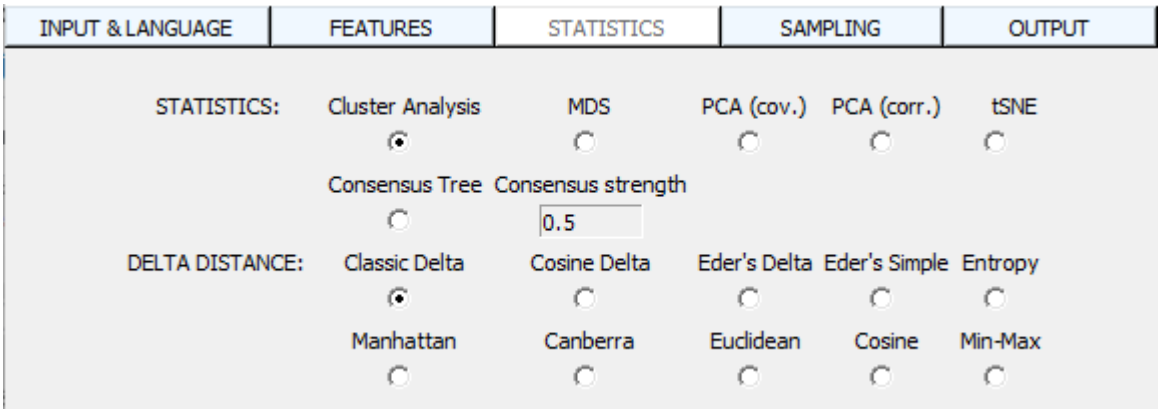
Stylo

Wywołując funkcję stylo zostanie otwarta aplikacja w trybie graficznym, która pozwala w prosty sposób konfigurować ustawienia na postawie, których chcemy dokonać analizy stylu artysty. Można także normalnie konfigurować wszystkie opcje w edytorze tekstowym zamiast gui. Zaletą takiej opcji jest większa możliwość konfiguracji ponieważ w trybie graficznym nie ma wszystkich opcji dostępnych. W aplikacji jest wiele opcji do konfiguracji, z czego najważniejsze są zakładki FEATURES, STATISTICS, SAMPLING.

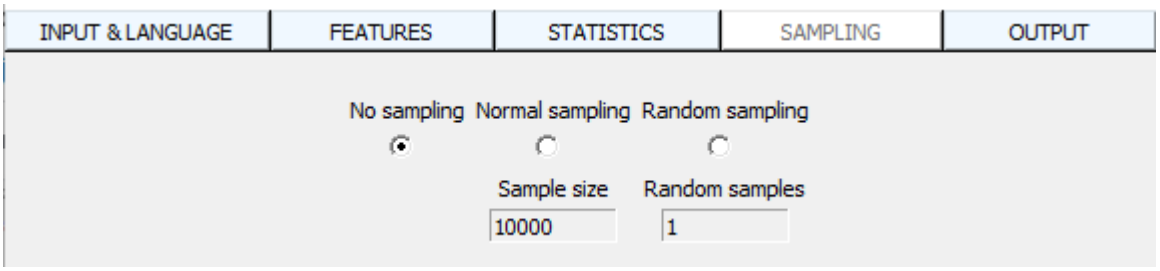
Opcje w zakładce pozwalają konfigurować, na jakiej zasadzie tworzone są cechy, ile ich ma być oraz możliwość na pomijanie pewnych cech, aby zwiększyć skuteczność. W naszym przypadku pozostawiamy opcję tak jak są domyślnie, gdyż chcemy podzielić tekst względem słów oraz nie chcemy niczego usuwać.



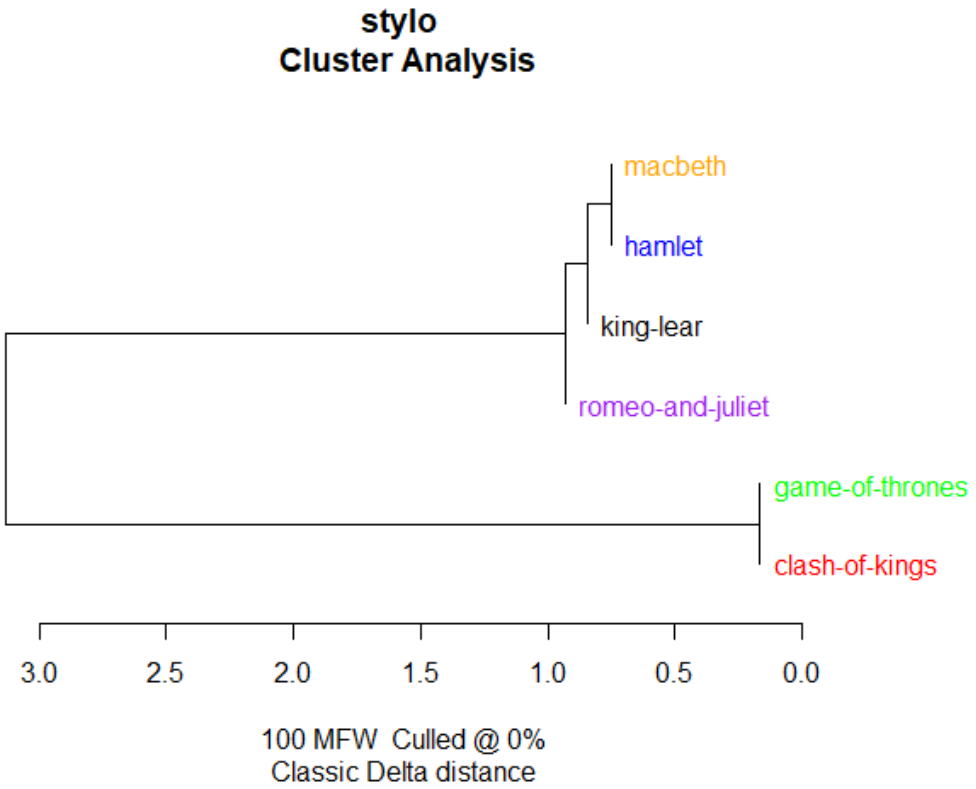
Opcje w zakładce pozwalają wybór, jaką statystykę chcemy sprawdzić oraz algorytm obliczania podobieństwa między tekstami. W tym przypadku nie będziemy zmieniać funkcji służącej do obliczania dystansu, ale będziemy testować różne funkcje statystyk.



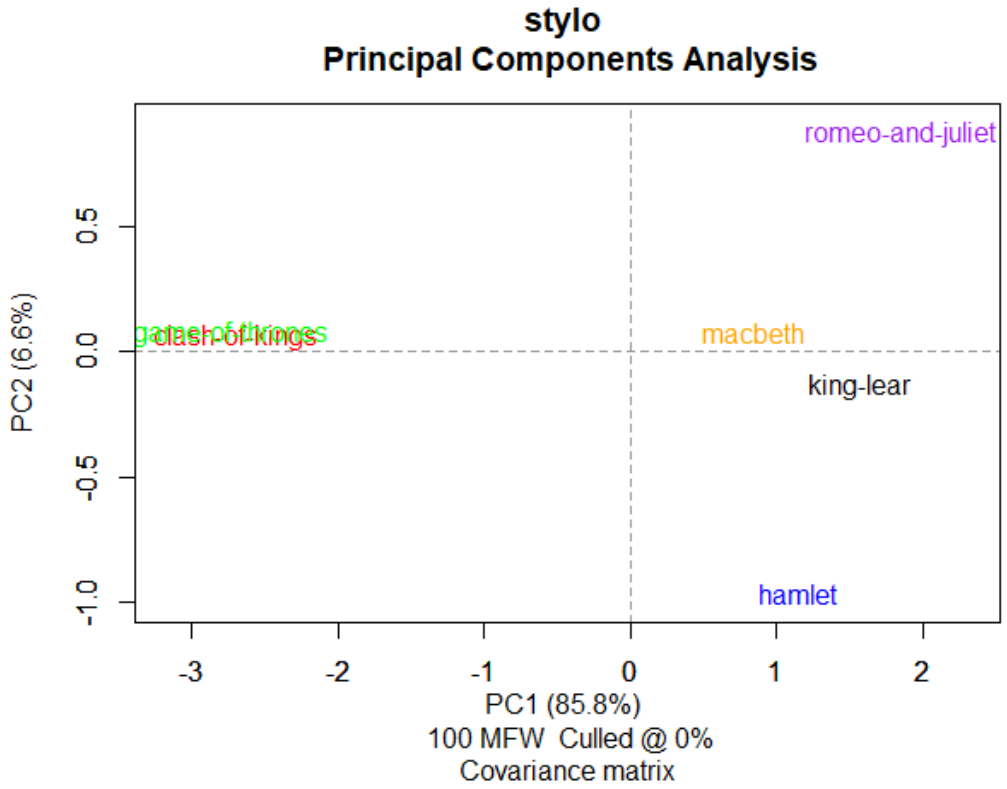
Zakładka sampling pozwala na konfigurację podziału analizy tekstu na mniejsze fragmenty w przypadku, gdyby tekst był za duży do analizy. W tym przypadku tak samo jak w przypadku features nie zmieniamy niczego, gdyż chcemy analizować tekst jako całość, a nie dzielić go na mniejsze części.

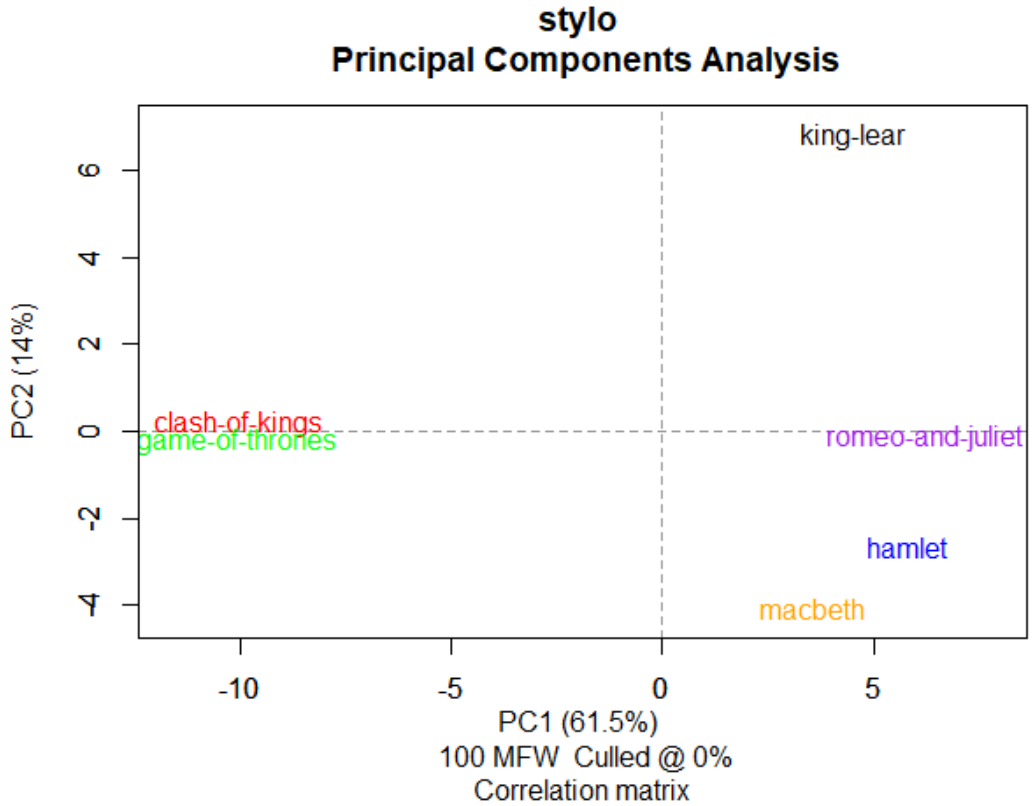


Cluster Analysis - analiza, która ma na celu wyznaczenie grup na podstawie pewnych cech. W naszym przypadku książki tego samego autora zostały połączone w tę samą grupę.



PCA - analiza głównych składowych, która ma na celu wyznaczenie nowej przestrzeni obserwacji, w której najwięcej zmienności wyjaśniają początkowe czynniki. PCA może być oparta o macierz kowariancji lub macierz korelacji. Widać w obu przypadkach wyraźny podział między dziełami stworzonymi przez różnych autorów i bez problemu moglibyśmy dokonać odseparowania oraz podzielenia utworów na grupy.


stylo
Principal Components Analysis
romeo-and-juliet
game-of-thrones
clash-of-kings
macbeth
king-lear
hamlet
PC2 (6.6%)
0.5
0.0
-0.5
-1.0
-3
-2
-1
0
1
2
PC1 (85.8%)
100 MFW Culled @ 0%
Covariance matrix


stylo
Principal Components Analysis
king-lear
clash-of-kings
game-of-thrones
romeo-and-juliet
hamlet
macbeth
PC2 (14%)
6
4
2
0
-2
-4
-10
-5
0
5
PC1 (61.5%)
100 MFW Culled @ 0%
Correlation matrix

rolling.classify

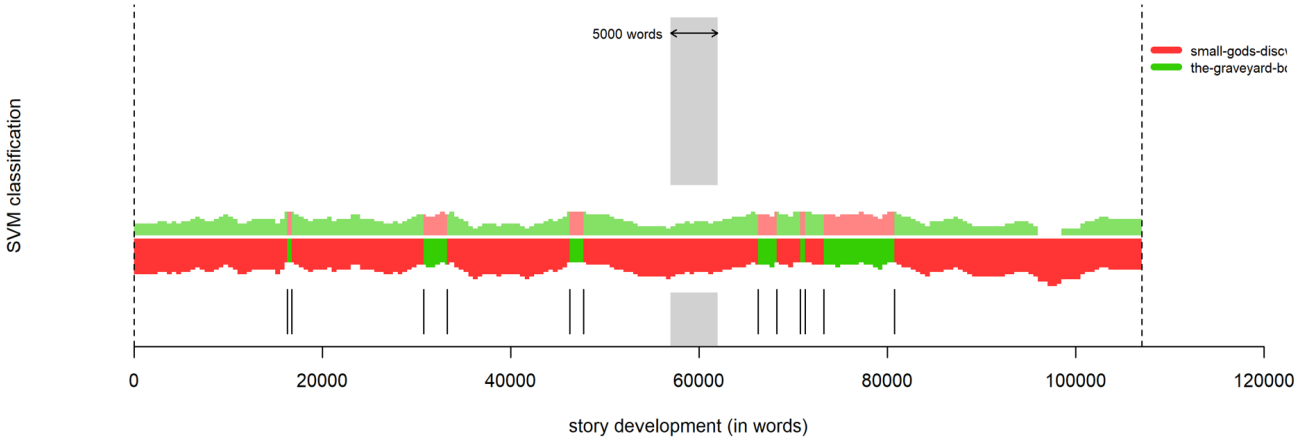
Funkcja dzieląca tekst na kolejne bloki o równej wielkości i przeprowadzająca nadzorowaną klasyfikację tych bloków względem zbioru uczącego. Funkcja ta może być wykorzystywana między innymi do tego, aby sprawdzać, które części tekstu mają inny styl co może oznaczać, że zostały napisane przez innego autora.

```
rolling.classify(
  write.png.file = TRUE,
  classification.method = "svm",
  mfw = 100,
  training.set.sampling = "normal.sampling",
  slice.size = 5000,
  slice.overlap = 4500
)
```

Do przetestowania zostały wykorzystane 3 książki:

- Good Omens (Neil Gaiman , Terry Pratchett)
- Small Gods (Terry Pratchett)
- The Graveyard Book (Neil Gaiman)

Good Omens zostało napisane przez dwóch autorów i chcemy dowiedzieć się, jak autorzy podzielili się w czasie tworzenia dzieła.



Wyraźnie widać, że utwór został stworzony przez 2 autorów i da się nawet określić, które części utworu zostały wykonane przez danego autora. Możliwe, że nie jest to idealne szczególnie, że autorzy, którzy piszą utwór w kilka osób starają się pisać w jednolitym stylu, niemniej jednak mimo to algorytm był w stanie odróżnić 2 autorów ze względu na ich styl.

Podsumowanie i wnioski

Z pomocą biblioteki “stylo” jesteśmy w stanie całkiem dobrze oraz łatwo określić podobieństwo między innymi utworami, a nawet fragmentami utworów. W oparciu o tą bibliotekę byłaby możliwość stworzenia prostego systemu antyplagiatowego, problemem byłoby tylko uzyskanie zbioru danych.