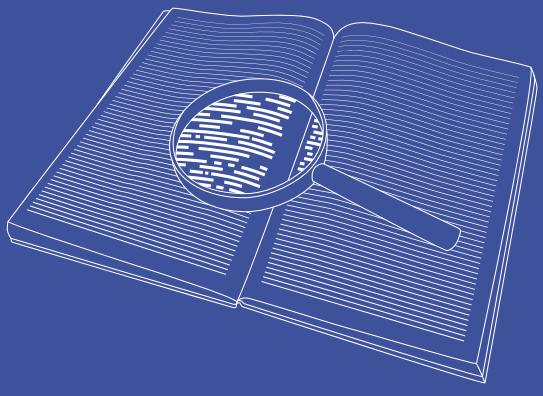
OŚRODEK PRZETWARZANIA INFORMACJI
PAŃSTWOWY INSTYTUT BADAWCZY

JSA

Jednolity System Antyplagiatowy



redaktor naukowy
Marek Kozłowski

Systemy informatyczne
wspierające naukę i szkolnictwo wyższe

JSA:
Jednolity System Antyplagiatowy

Redakcja naukowa:
dr Marek Kozłowski

OŚRODEK PRZETWARZANIA INFORMACJI
PAŃSTWOWY INSTYTUT BADAWCZY

Systemy informatyczne wspierające naukę i szkolnictwo wyższe
JSA: Jednolity System Antyplagiatowy

Redakcja naukowa:
dr Marek Kozłowski

Redakcja techniczna:
Krystyna Fetera

Autorzy:
Ewa Dulińska: rozdział 1
Daniel Karaś: rozdział 2, 3
Maciej Kowalski: rozdział 2, 3
dr Marek Kozłowski: rozdział 1
Dawid Lubowiecki: rozdział 3
Marek Nurzyński: rozdział 2
Łukasz Podlodowski: rozdział 2, 3
Paweł Szczepanowski: rozdział 3
Martyna Śpiewak: rozdział 2, 3
Jakub Uszyński: rozdział 2, 3
Małgorzata Wartacz: rozdział 1

Recenzent:
prof. dr hab. inż. Henryk Rybiński

Wydawca:
Ośrodek Przetwarzania Informacji – Państwowy Instytut Badawczy
al. Niepodległości 188b
00-608 Warszawa
e-mail: opi@opi.org.pl
www.opi.org.pl

© Copyright by Ośrodek Przetwarzania Informacji – Państwowy Instytut Badawczy
Warszawa 2021
Wszelkie prawa zastrzeżone

ISBN 978-83-63060-21-3

Projekt okładki i opracowanie graficzne:
Karina Maszewska

Skład i łamanie:
dr Jakub Kierzkowski

Druk:
Drukarnia DSS Szczepan Szymański
ul. Wolska 108, 05-119 Wola Aleksandra

Szanowni Państwo,

rzetelność i oryginalność to cechy, którymi zawsze powinny charakteryzować się prace dyplomowe i doktorskie. Niezależnie, czy ich autorem jest student, który ubiega się o tytuł licencjata, czy też doktorant. Niezwykle ważne jest, aby dbać o najwyższe standardy. W tym celu właśnie Ośrodek Przetwarzania Informacji – Państwowy Instytut Badawczy (OPI PIB) zbudował Jednolity System Antyplagiatowy (JSA).

Nowoczesne narzędzie powstało w odpowiedzi na oczekiwania polskiej kadry akademickiej i uporządkowało rynek antyplagiatowy w Polsce. Wcześniej istniało wiele programów, opartych na różnych algorytmach i referencyjnych bazach danych. Dzięki oddaniu JSA do dyspozycji promotorów w 2019 roku, wszystkie prace inżynierskie, licencjackie, magisterskie i doktorskie są analizowane za pomocą tego samego narzędzia. System jest całkowicie bezpłatny. Korzystać z niego mogą promotorzy prac dyplomowych i doktorskich zatrudnieni we wszystkich krajowych uczelniach i instytutach, które mają prawo nadawania tytułów zawodowych i stopni naukowych. Dzięki wprowadzeniu JSA, zapewniliśmy większą niż w latach wcześniejszych porównywalność wyników weryfikacji oryginalności prac dyplomowych i doktorskich pomiędzy wszystkimi ośrodkami z całej Polski.

Wprowadzenie Jednolitego Systemu Antyplagiatowego okazało się dużym sukcesem. Jeszcze w 2016 roku zaledwie 30-40 proc. prac dyplomowych było analizowanych pod kątem plagiatu. Od 2019 roku takim obowiązkiem objęte są wszystkie prace. Warto także dodać, że ponad 2 miliardy dokumentów zawierają bazy referencyjne, z którymi JSA porównuje badaną pracę, a system pracuje na aż prawie 80 TB danych. Dodatkowo to nowoczesne narzędzie jest zintegrowane z Ogólnopolskim Repozytorium Pisemnych Prac Dyplomowych (ORPPD), które zawiera ponad 3 miliony obronionych od 2009 roku prac.

System stale jest rozwijany przez ekspertów OPI PIB, którzy pracują nad jego rozbudową i modernizacją. Poszerzono liczbę baz referencyjnych i zapewniono większe umiędzynarodowienie tego użytecznego narzędzia. Zdarza się, że plagiat jest zaczerpnięty z innych zasobów internetu niż tylko z polskich źródeł, czy też polega on na dosłownym przetłumaczeniu prac lub stron internetowych z innego języka. Dzięki wprowadzanym przez OPI PIB zmianom, system może takie teksty również wskazać promotorowi. JSA automatycznie tłumaczy tekst i porównuje z źródłami obcojęzycznymi, cały proces jest łatwy i nie wymaga stosowania dodatkowego narzędzia.

Od 2018 roku prowadzone są liczne szkolenia, kursy oraz spotkania mające na celu ułatwienie pracownikom akademickim pracy w systemie. Przygotowano także materiały edukacyjne dostępne na kanale YouTube OPI PIB oraz kurs w portalu NAVOICA. Wszystkich zainteresowanych zachęcam do zamawiania bezpłatnych szkoleń, które

z pewnością pomogą lepiej zrozumieć działanie systemu. Eksperci OPI PIB są do Państwa dyspozycji.

Kończąc chciałbym przypomnieć, że Jednolity System Antyplagiatowy sprawdza podobieństwo prac dyplomowych i doktorskich do tekstów i opracowań, które powstały już wcześniej. System jednak nie przesądza, czy dana praca jest plagiatem. JSA służy jedynie jako narzędzie wspierające pracę promotora i ostateczna decyzja zawsze należy do niego.

Wojciech Murdzek

Sekretarz Stanu,
Pełnomocnik Rządu do spraw reformy
funkcjonowania instytutów badawczych

Ministerstwo Edukacji i Nauki

SPIS TREŚCI

Wstęp 9

Rozdział 1. O Systemie JSA 11

Ewa Dulińska, Małgorzata Wartacz, dr Marek Kozłowski

1.1. Podstawa prawna JSA ... 13

1.2. Historia JSA ... 15

1.2.1. Etap budowy systemu i wdrożenia ... 15

1.2.2. Etap rozwoju i utrzymania systemu ... 17

1.3. Konkurs JSA ... 23

1.3.1. Zadanie 1: Wykrywanie nadużyć między dokumentami ... 24

1.3.2. Zadanie 2: Wykrywanie podejrzanych dokumentów ... 25

1.3.3. Zadanie 3: Wykrycie potencjalnego plagiaryzmu w ramach pojedynczego dokumentu ... 27

1.3.4. Zadanie 4: Wykrywanie podejrzanych stron internetowych ... 28

1.4. Geneza powstania oraz zakres funkcjonalny systemu ... 29

1.4.1. Geneza powstania JSA ... 29

1.4.2. Zakres funkcjonalny JSA ... 30

1.4.3. Interfejs graficzny użytkownika ... 31

1.4.4. Integracja z JSA poprzez interfejsy maszynowe (API) ... 33

1.4.5. Uprawnienia ... 35

1.4.6. Wynik badania – raport JSA ... 35

1.4.7. Administracja i ustawienia systemu ... 37

1.5. Szkolenia ... 38

1.6. Statystyki JSA ... 40

Rozdział 2. Metody Przetwarzania Danych 45

Daniel Karaś, Jakub Uszyński, Łukasz Podlodowski, Marek Nurzyński, Maciej Kowalski, Martyna Śpiewak

2.1. Indeksowanie danych – LSH Forest ... 46

2.1.1. LSH Forest – definicje ... 46

2.1.2. LSH Forest w JSA ... 47

2.1.3. LSH Forest – ewaluacja ... 48

2.1.4. LSH Forest – strona techniczna ... 49

2.2. Indeksowanie danych – Elasticsearch 50
2.2.1. Architektura klastra 50
2.2.2. Ładowanie danych 51
2.2.3. Wyszukiwanie kandydatów na dokumenty źródłowe w JSA 51
2.3. Detekcja podobnych pasaży między dwoma dokumentami 53
2.3.1. Opis problemu 53
2.3.2. Specyfikacja wejścia i wyjścia algorytmu 53
2.3.3. Zarys działania algorytmu 55
2.3.4. Ewaluacja algorytmu 57
2.3.5. Adaptacja algorytmu w ramach systemu JSA 59
2.4. Stylometria 61
2.4.1. Założenia 61
2.4.2. Funkcja zmiany stylu 62
2.4.3. Detekcja fragmentów podejrzanych 63
2.5. Chatbot JSA 65
2.5.1. Geneza zbudowania chatbota w JSA 65
2.5.2. Model chatbota 67

Rozdział 3. Architektura i Technologie **71**
Daniel Karaś, Dawid Lubowiecki, Jakub Uszyński, Łukasz Podlodowski, Maciej Kowalski, Martyna Śpiewak, Paweł Szczepanowski

3.1. Architektura Systemu JSA 71
3.1.1. Model hybrydowy realizowany przez JSA 72
3.1.2. Cykl życia dokumentu 73
3.1.3. Cykl życia badania i próby 74
3.1.4. Pipeline 75
3.1.5. Listener 76
3.1.6. Broker 77
3.2. Technologie wspomagające 78
3.2.1. ActiveMQ 78
3.2.2. Wildfly 78
3.2.3. Load Balancing 78
3.2.4. Postgres 78
3.2.5. ELK Stack 79
3.2.6. Wkhtmltox 79
3.3. Integracja z systemami zewnętrznymi 80
3.3.1. POL-on 80
3.3.2. NEKST 81
3.3.3. Bing 82

3.3.4. OpenAccess ... 82
3.3.5. ORPPD ... 83
3.4. Aplikacja frontend ... 83
3.4.1. Komunikacja z backendem ... 84
3.4.2. Testy ... 85
3.4.3. Formularze ... 85
3.4.4. Wersje językowe ... 86
3.4.5. Ograniczenia przeglądarek ... 86
3.4.6. Dostępność dla osób z niepełnosprawnościami ... 87
3.5. Chatbot ... 88
3.5.1. Chatbot Broker ... 89
3.5.2. Serwis obliczeniowy ... 89

Zakończenie **91**

Spis ilustracji **93**

Spis tabel **94**

Spis fragmentów kodu **95**

Bibliografia **97**

WSTĘP

Jednolity System Antyplagiatowy (JSA) jest jedynym ustandaryzowanym i oficjalnym sposobem weryfikacji oryginalności prac dyplomowych i doktorskich w Polsce. Decyzję o stworzeniu JSA podjęło Ministerstwo Nauki i Szkolnictwa Wyższego, w celu uporządkowania rynku antyplagiatowego w Polsce.

JSA to jedyny bezpłatny system antyplagiatowy w Polsce o niespotykanej dotąd skali. Od 2019 roku pisemne prace inżynierskie, licencjackie, magisterskie i doktorskie dopuszczone do obrony muszą być przeanalizowane za pomocą JSA. W tym czasie zbadano w JSA już prawie 1 milion prac dyplomowych i doktorskich. Wynik analizy jest kluczowy dla procesu obrony i potwierdza weryfikację dokonaną przez promotora. JSA przeznaczony jest dla promotorów prac, a więc nie ma możliwości korzystania z systemu przez studentów.

Jednolity System Antyplagiatowy został zrealizowany jako projekt pozakonkursowy w ramach Programu Operacyjnego Wiedza Edukacja Rozwój realizowanego przez Narodowe Centrum Badań i Rozwoju.

U podstaw decyzji o wdrożeniu Jednolitego Systemu Antyplagiatowego (JSA) stoją następujące założenia:

1. Z systemu na zasadach wolnego dostępu (nieodpłatnie) korzystać mogą uczelnie publiczne i niepubliczne, a także inne jednostki dyplomujące jak instytuty prowadzące prace doktorskie.
2. Koszt wdrożenia i utrzymania jednolitego systemu oznacza duże oszczędności dla budżetu państwa – wg szacunków liczonych w latach 2015-16 rocznie miał on kosztować kilka mln zł zamiast kilkunastu-kilkudziesięciu mln zł, co miałoby miejsce gdyby w całej strukturze szkolnictwa wyższego istniało kilkanaście różnych podsystemów zakupionych od różnych dostawców, które wymagałyby płatnych aktualizacji i indywidualnych zakupów infrastrukturalnych.
3. Wprowadzenie Jednolitego Systemu Antyplagiatowego było rekomendowane przez Komisję Europejską.
4. Kluczową zaletą proponowanego rozwiązania jest zakładana jednolitość i kompletność danych, a także to, że uczelnie zamiast organizować podsystemy na własną rękę, utrzymywać je i aktualizować, będą dostawały nieodpłatnie gotowe rozwiązanie. Wypracowany system musi obejmować wszystkie obszary kształcenia na poziomie wyższym.
5. Efektywność ekonomiczna, organizacyjna i informatyczna – system antyplagiatowy musi współpracować z ogólnopolskim repozytorium prac dyplomowych (ORPPD). W przypadku braku JSA i funkcjonowania wielu różnych systemów każdy z nich

musiałby być osobno zintegrowany z ORPPD, co znacznie zwiększyłoby koszty. Funkcjonujące w latach 2015-2016 systemy antyplagiatowe były niewystarczające i wymagały znacznej rozbudowy. Korzystniejsze z perspektywy ekonomicznej, organizacyjnej i informatycznej było stworzenie centralnego systemu, zastępującego dotychczas istniejące systemy na uczelniach wyższych i jednocześnie uzupełniającego ich funkcje.

Jak powstał JSA?

Jednolity System Antyplagiatowy powstał na podstawie analizy potrzeb użytkowników. W tym celu odbyła się seria spotkań na uczelniach w całym kraju, w ramach których zebrano oczekiwania wobec nowego systemu antyplagiatowego. Równolegle, na każdej uczelni wyznaczono przedstawicieli ds. systemu antyplagiatowego, z którymi zespół OPI PIB, odpowiedzialny za zbudowanie systemu, utrzymywał kontakt w formie mailingów. W maju 2018 roku, udostępniono pierwszą wersję oprogramowania demonostracyjnego. Promotorzy, którzy chcieli mieć wpływ na finalny kształt programu do wykrywania plagiatów testowali jego wersję alfa a potem beta. Siedem miesięcy później oficjalnie uruchomiono system i przekazano go do użytku wszystkim uczelniom w kraju. Pojawiające się kolejno i zgłaszane do OPI PIB potrzeby użytkowników były analizowane i wdrażane w oprogramowaniu. Przeprowadzono także 3 tury badań użyteczności po to, aby system był przyjazny dla użytkowników. Na podstawie raportu z tych badań wprowadzono wiele udogodnień i zmian w systemie.

Niniejsza publikacja ma na celu przedstawienie głównych cech systemu i jego zastosowań w formie opisu jego historii rozwoju, metod w nim użytych czy architektury i technologii wykorzystanych przy jego budowie, oraz opisanie doświadczeń i ich wyników w obszarze wdrażania tak rozległego przedsięwzięcia. Szczególna uwaga poświęcona została zwłaszcza technicznym aspektom w obszarze praktyki wdrażania systemów informatycznych. Budowa systemu stanowiła wyzwanie w zakresie znalezienia rozwiązań do bardzo szerokiej i zróżnicowanej gamy problemów. Każdy z tych problemów stanowi ciekawy materiał do dalszych analiz i wymiany doświadczeń na gruncie naukowym.

Publikacja podzielona została na następujące rozdziały:

1. Rozdział 1 – ma na celu przedstawienie jak największej liczby faktów dotyczących systemu JSA, aby dać czytelnikowi jak największą wiedzę na temat jego skali i znaczenia.
2. Rozdział 2 – skupia się na metodach przetwarzania danych wykorzystanych w JSA w procesie wieloaspektowej detekcji plagiatów.
3. Rozdział 3 – skupia się na architekturze i technologiach wykorzystanych przy budowie systemu.

Publikacja ma na celu nie tylko przedstawienie kluczowych faktów dotyczących systemu, ale przede wszystkim opis empirycznych doświadczeń i zastosowanych rozwiązań, które mogą stanowić wartość dla innych osób w realizacji podobnych przedsięwzięć lub szukających zestawu ciekawych rozwiązań w tym metod przetwarzania danych, czy architektury i technologii stosowanej przy tego rodzaju wyzwaniach.

ROZDZIAŁ 1

O SYSTEMIE JSA

Ewa Dulińska
Małgorzata Wartacz
dr Marek Kozłowski

Poniższy rozdział ma na celu przedstawienie jak największej liczby faktów dotyczących systemu JSA, by dać czytelnikowi jak najobszerniejszą wiedzę na temat jego skali i znaczenia w opisywanej dziedzinie. Stanowi to też formę wprowadzenia w bardziej dogłębne zagadnienia opisujące doświadczenia z pracy nad systemem i przeprowadzone eksperymenty badawcze, przedstawione w kolejnych rozdziałach.

Celem opisu nie jest wyłącznie zaprezentowanie danych i aspektów prawnych, ale próba przedstawienia i określenia społecznego wpływu systemu na wiele dziedzin, w których można zaobserwować jego oddziaływanie.

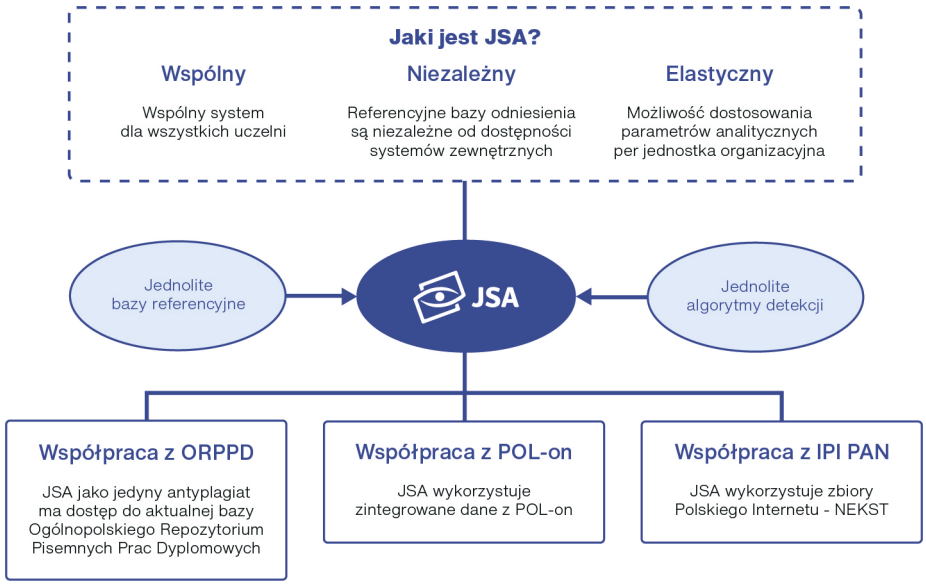
Jednolity System Antyplagiatowy jako jedyny na rynku systemów antyplagiatowych ma pełen dostęp do stale aktualizowanej bazy pisemnych prac dyplomowych jaką jest ORPPD, a dane wprowadzane wraz z badaną pracą są wspierane przez system POL-on. System korzysta z wyszukiwarki Bing ale także z korpusu polskich stron internetowych NEKST, nie jest więc ograniczony tylko do wyników z wyszukiwarki internetowej. JSA jest także systemem nieodpłatnym i wspólnym dla wszystkich uczelni w kraju, gwarantując tym samym jednolite metody badawcze, które działają na jednorodnym zestawie danych odniesienia. Jedną z zalet systemu jest zapewnienie jednakowych standardów badania antyplagiatowego wszystkim instytucjom nadającym tytuły zawodowe oraz stopnie i tytuły naukowe.

W skład podstawowych referencyjnych źródeł danych systemu JSA wchodzą:

1. NEKST – korpus polskojęzycznych zasobów internetowych liczący ponad 2 mld stron, wykorzystywany w semantycznej wyszukiwarce internetowej autorstwa IPI PAN;
2. Bing – wyszukiwarka internetowa stworzona przez amerykańską firmę Microsoft, pozwalająca na dostęp do międzynarodowych zasobów internetowych;
3. ORPPD – Ogólnopolskie Repozytorium Pisemnych Prac Dyplomowych, liczące ponad 3 mln prac dyplomowych;
4. Wikipedie – 10 najpopularniejszych w Unii Europejskiej wersji językowych Wikipedii;

5. OpenAccess – publikacje naukowe tzw. Otwartego Dostępu liczące ponad 3 miliony dokumentów;
6. Bazy aktów prawnych i orzeczeń;
7. Baza postępowań awansowych.

Korzystanie z systemu zapewnia standaryzację badania antyplagiatowego w odniesieniu do jednakowych baz referencyjnych i według tych samych algorytmów, co jednocześnie zapewnia równe traktowanie wszystkich dyplomantów.



Ilustracja 1.1. JSA – ujęcie ogólne

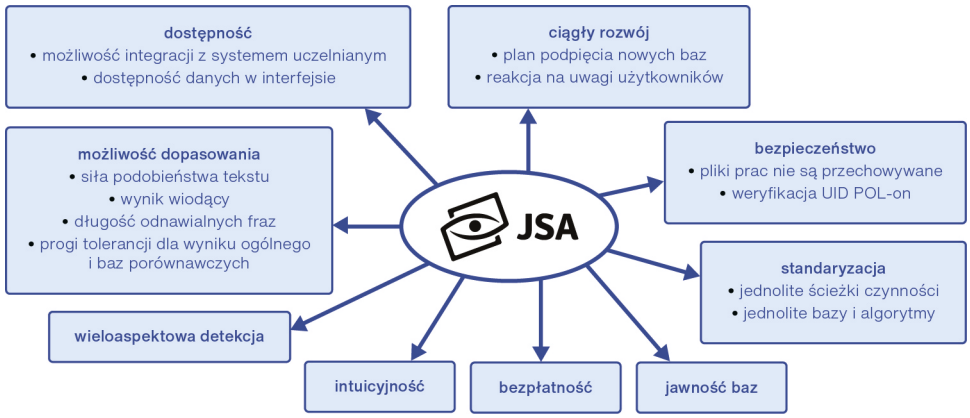
Wśród zalet systemu można wymienić jego ciągły rozwój. Zespół analityków i deweloperów pracuje niezmiennie nad doskonaleniem i rozwijaniem funkcjonalności. Ostatnio rozszerzyliśmy system JSA o nowe dane pochodzące z wyszukiwarki Bing, czy bazy dokumentów wykorzystywanych w postępowaniach awansowych. W kolejnych kwartałach rozwoju systemu dalej będziemy dążyć do podłączenia kolejnych nowych baz referencyjnych. Wychodząc naprzeciw potrzebom uczelni dostarczyliśmy możliwość dopasowania ustawień parametrów na poziomie całej uczelni lub pojedynczych jednostek. W zależności od typu prac bronionych na uczelniach można określić progi tolerancji wyników oraz wrażliwość systemu na wyszukiwane podobieństwa.

Dzięki ciągłej weryfikacji doświadczeń użytkowników i dostosowaniu budowy interfejsu system jest bardzo intuicyjny i łatwy w obsłudze. Aktualnie trwa również dostosowanie systemu do potrzeb osób niepełnosprawnych.

W celu wsparcia uczelni, które posiadają wewnętrzne systemy obsługi administracyjnej, JSA udostępnia możliwość zintegrowania się systemów i wspólnej obsługi procesu

badania antyplagiatowego. Prace przesłane przez system uczelniany są jednakowo widoczne w interfejsie użytkownika, dzięki czemu w razie potrzeby każdy promotor może pracować na wyniku zgodnie ze swoimi upodobaniami.

System JSA jest wyjątkowo bezpieczny. Uprawnienia do dostępu weryfikowane są poprzez numery UID z systemu POL-on, a pliki wgrane w celu badania nie są przez system przechowywane.



Ilustracja 1.2. JSA – główne cechy systemu

1.1. Podstawa prawna JSA

Idea powstania ujednoliconego, na skalę polskiego szkolnictwa wyższego, systemu pozwalającego na skuteczną walkę z plagiatem nie jest nowa. Pierwsze poważne kroki poczyniono w 2016 roku, kiedy to 5 kwietnia do Sejmu wpłynął poselski projekt ustawy o zmianie ustawy „Prawo o szkolnictwie wyższym" oraz niektórych innych ustaw. Pierwsze przepisy powołujące do życia JSA zapisane zostały już w treści ustawy z dnia 20 maja 2016 r. o zmianie ustawy – Prawo o szkolnictwie wyższym oraz ustawy o zmianie ustawy – Prawo o szkolnictwie wyższym oraz niektórych innych ustaw ([25]), dokładniej w artykule 1 zgłoszono zmiany, które przytaczamy poniżej:

```
1) w art. 167a ust. 4 otrzymuje brzmienie:
  „4. Uczelnia jest obowiązana do sprawdzania pisemnych prac dyplomowych
przed egzaminem dyplomowym z wykorzystaniem Jednolitego Systemu
Antyplagiatowego, o którym mowa w art. 167c ust. 1.";

2) po art. 167b dodaje się art. 167c w brzmieniu:
  „Art. 167c. 1. Minister właściwy do spraw szkolnictwa wyższego
prowadzi Jednolity System Antyplagiatowy współpracujący z ogólnopolskim
repozytorium pisemnych prac dyplomowych oraz zapewnia uczelniom
nieodpłatne korzystanie z tego systemu.
  2. Opracowanie systemu, o którym mowa w ust. 1, i administrowanie tym
systemem minister właściwy do spraw szkolnictwa wyższego może zlecić
```

```
instytutowi badawczemu, nadzorowanemu przez ministra właściwego do spraw
nauki, którego przedmiot działania jest ściśle związany ze świadczeniem
usług w zakresie systemów informacyjnych, zapewniając środki finansowe
na ten cel.".
```

Ustawa powołująca do życia Jednolity System Antyplagiatowy przeszła sprawnie przez cały proces legislacyjny i po niespełna 3 miesiącach została podpisana przez Prezydenta RP 23 czerwca 2016 roku. Uchwalenie tego aktu prawnego pozwoliło na wprowadzenie obowiązku sprawdzania przez wszystkie uczelnie wyższe pisemnych prac dyplomowych za pomocą ujednoliconego systemu antyplagiatowego. Prace nad systemem mogły się oficjalnie rozpocząć.

W związku z podjęciem przez Sejm prac nad nowelizacją ustawy o szkolnictwie wyższym 5 kwietnia 2018 roku rozpoczęto prace nad rządowym projektem ustawy Prawo o szkolnictwie wyższym i nauce. Poza poprawkami dotyczącymi konsolidacji niektórych podstawowych regulacji i połączeniu przepisów normujących nie tylko szkolnictwo wyższe, ale także finansowanie nauki, stopnie i tytuły naukowe czy pomoc materialną dla studentów, przedmiotem zmian były także przepisy dotyczące Jednolitego Systemu Antyplagiatowego. Tak zwana ustawa 2.0 została uchwalona 20 lipca 2018 roku ([26]).

Treść artykułów dotyczących JSA w ramach ustawy 2.0:

```
Art. 188 ust. 4 Jeżeli rozprawa doktorska jest pracą pisemną, podmiot
doktoryzujący sprawdza ją przed obroną z wykorzystaniem Jednolitego
Systemu Antyplagiatowego, o którym mowa w art. 351 ust. 1.
```

```
Art. 351. 1. Minister prowadzi Jednolity System Antyplagiatowy.
2. Jednolity System Antyplagiatowy zapewnia wsparcie w zakresie
przeciwdziałania naruszeniom przepisów o prawie autorskim i prawach
pokrewnych.
3. Jednolity System Antyplagiatowy wykorzystuje dane zawarte
w repozytorium pisemnych prac dyplomowych oraz bazie dokumentów
w postępowaniach awansowych.
4. Minister zapewnia nieodpłatne korzystanie z Jednolitego Systemu
Antyplagiatowego uczelniom, instytutom PAN, instytutom badawczym,
instytutom międzynarodowym, PKA oraz RDN.
```

1.2. Historia JSA

System JSA był rozwijany dwuetapowo: zbudowanie i wdrożenie w ramach projektu POWER, a następnie w ramach utrzymania aplikacji, po jej uruchomieniu, w postaci rozbudowy i dostosowania zakresu funkcjonalności do oczekiwań użytkowników, zmian prawa, czy też zmian zależnych od rozwoju systemów zintegrowanych z JSA.

1.2.1. Etap budowy systemu i wdrożenia

Ilustracja 1.3. Harmonogram prac

Pierwszy etap powstania Jednolitego Systemu Antyplagiatowego został zrealizowany jako projekt w ramach Programu Operacyjnego Wiedza Edukacja Rozwój (POWER) realizowanego przez Narodowe Centrum Badań i Rozwoju. Jego zakres obejmował zaprojektowanie i budowę systemu od podstaw aż do momentu wdrożenia na środowisko produkcyjne. Z uwagi na względy organizacyjne i ekonomiczne, na tym etapie, jedynym wykonawcą był Ośrodek Przetwarzania Informacji – Państwowy Instytut Badawczy, który zrealizował prace od 2017 roku do czerwca 2019 roku w trybie pozakonkursowym. Zlecenie dla OPI PIB utworzenia i utrzymania systemu antyplagiatowego zostało zapisane w ustawie i powierzone przez ministra właściwego Ministerstwa Nauki i Szkolnictwa Wyższego.

Projekt, oprócz prac programistycznych, obejmował szereg szkoleń mających na celu zapoznanie z jego funkcjonalnościami kadry uczelnianej odpowiedzialnej za wprowadzanie prac na uczelniach oraz ich weryfikację. OPI PIB, zgodnie z obowiązkiem nakładanym na niego przez ustawę, tworzy, utrzymuje i rozwija Jednolity System Antyplagiatowy.

Rozpoczęcie prac miało miejsce w 2016 roku. Prace w projekcie można podzielić w następujący sposób:

1. Zbieranie wymagań funkcjonalnych od użytkowników końcowych podczas spotkań w różnych uczelniach;

2. Analiza aktualnego stanu cyfryzacji uczelni w odniesieniu do funkcjonujących programów antyplagiatowych. Na tej podstawie opracowano i przygotowano warunki konkursowe, które pozwoliły wybrać optymalne rozwiązanie antyplagiatowe (więcej o konkursie znajduje się w podrozdziale 1.3);
3. Wybór w drodze konkursu optymalnych rozwiązań, które docelowo zostały w różny sposób wykorzystane w ramach rozwijanej centralnej platformy antyplagiatowej. Zapewniły efektywne i skalowalne działanie systemu na dużych zbiorach prac dyplomowych oraz stron internetowych;
4. Właściwa budowa Jednolitego Systemu Antyplagiatowego i sukcesywny jego rozwój. Budowa na tym etapie obejmowała powstanie silnika detekcji, interfejsów użytkownika i końcówek integracyjnych dla systemów zewnętrznych;
5. Od wczesnych etapów prac jako integralną część projektu traktowano szkolenia przedstawicieli uczelni, których zadaniem było przekazywanie wiedzy dotyczącej procedur weryfikacji prac, charakterystyka systemu, jego założeń, możliwości funkcjonalnych i technicznych (więcej na temat szkoleń można przeczytać w podrozdziale 1.5);
6. Jako dodatkowe wsparcie kadry naukowej i administracyjnej stworzono też dedykowany system obsługi zgłoszeń. Głównym celem tego systemu było wspieranie użytkowników korzystających z JSA w trakcie realizacji projektu, jak również po jego zakończeniu.

W ramach testów zewnętrznych w dniu 21 maja 2018 roku udostępniona została pierwsza wersja systemu JSA DEMO1. System był wówczas dostępny dla niewielkiej grupy użytkowników z różnych instytucji, którzy zgłosili się jako ochotnicy do testów systemu w wersji beta.

Przeprowadzone testy, poza pozytywnym przyjęciem systemu, wykazały także elementy, które funkcjonalnie wymagały poprawy, dopracowania lub niewielkiej zmiany. Informacje zwrotne uczestników pilotażu dotyczyły zarówno sfery graficznej (*user experience*) jak i technicznej. Użytkownicy zgłosili, a zespół programistyczny zrealizował, takie postulaty jak: dodanie opcji wydruku raportu, zwiększenie efektywności logiki wyboru znalezionych przez system pasaży, które się na siebie nakładały, zniesienie limitu trzech prób badania dla jednej pracy, możliwość podglądu i stałego dostępu do wszystkich wykonanych prób badania czy też uzupełnienie raportów o pełną analizę tekstu. Uwzględnianie sugestii naszych użytkowników było dla nas zawsze priorytetem, a te uwagi pozostają dla nas cennym źródłem wymagań.

Z początkiem lipca 2018 roku oddano użytkownikom do zapoznania się pierwszy dokument opisujący usługi integracyjne API. Dzięki tak wczesnemu udostępnieniu dokumentu mogliśmy otrzymać informację zwrotną o powodzeniu w integracji z systemami uczelnianymi.

Październik 2018 roku obfitował w aktualności. Kolejna wersja testowa systemu, tym razem udostępniona dla wszystkich użytkowników, ujrzała światło dzienne z początkiem miesiąca. Środowisko testowe DEMO1 zostało zastąpione nowym środowiskiem DEMO2 jsa-demo.opi.org.pl zostało otwarte dla administratorów, których zadaniem było utworzenie kont innym użytkownikom w ramach swoich uczelni. Ze względu na uwie-

rzytelnianie wysokiego poziomu uprawnień, administratorem instytucji mógł zostać tylko taki użytkownik, który posiadał uprawnienia administracyjne w systemie POL-on.

Niespełna dwa tygodnie po odsłonie JSA dla szerokiego grona użytkowników przyjęto cykl cotygodniowych wdrożeń nowych wersji systemu, które sukcesywnie naprawiały wykryte błędy oraz wprowadzały dodatkowe funkcjonalności. Taka terminowość w wypuszczaniu nowych wersji systemu została utrzymana do końca 2019 roku, czyli pierwszego roku działania systemu w trybie produkcyjnym. Pod koniec października 2018 roku wdrożono kolejną wersję na środowisko testowe, która uwzględniała zmianę metody importu użytkowników oraz wprowadzenie możliwości logowania przez Moduł Centralnego Logowania (MCL, centralny sposób logowania do systemów wytwarzanych przez OPI PIB) dla administratorów POL-onu, a także wprowadzono wersjonowanie aktualizacji systemu.

Jeszcze przed odsłoną wersji produkcyjnej, w grudniu 2018 roku dodano w systemie, na prośbę użytkowników, możliwość ustawienia parametrów wyniku badania na poziomie instytucji. Użytkownicy zgłaszali konieczność zróżnicowanego traktowania prac o charakterystyce humanistycznej od technicznej czy matematycznej. Wdrożenie rozróżnialnych i podlegających zmianie parametrów pozwoliło na zaspokojenie tej potrzeby.

1.2.2. Etap rozwoju i utrzymania systemu

W dniu 2 stycznia 2019 roku uruchomiono środowisko produkcyjne pod adresem jsa.opi.org.pl.

Wszystkie badania prac przeprowadzone na środowisku testowym do dnia wdrożenia wersji produkcyjnej JSA zostały na prośbę użytkowników przeniesione wraz z ich kontami do wersji produkcyjnej. W zespole projektowym podjęto decyzję, że oprócz środowiska produkcyjnego systemu, środowisko testów zewnętrznych (DEMO2) pozostanie dostępne dla użytkowników. Od tego też dnia oba środowiska stały się niezależne, co znaczy, że o ile kolejne wersje systemu wdrażane były zarówno na środowisko testowe jak i produkcyjne, to jednak konta użytkowników i ich akcje są niezależne od siebie.

W ciągu kolejnych dwóch tygodni, wychodząc naprzeciw oczekiwaniom użytkowników, umożliwiono administratorom JSA pochodzącym z POL-onu nadawanie swoim kontom innych uprawnień.

Kluczową zmianą było także dodanie parametru siły podobieństwa tekstu. Siła podobieństwa tekstu była pierwszym sterowalnym parametrem udostępnionym użytkownikom. Dawała ona możliwość dostosowywania wrażliwości detekcji na parafrazy, których dopuszczalność i zasadność w pracach dyplomowych w różnych dziedzinach nauki jest zróżnicowana. Opcja ta była dostępna dla Administratorów systemu, a jej ustawienia wpływały na to, w jakim stopniu system sprawdzał podobieństwo semantyczne tekstów (fragmenty, w których mogły być użyte modyfikacje takie jak zmieniony szyk zdań, parafraza czy synonimy). Zmiana jego wartości nie odzwierciedlała jednoznacznie zwracanej

liczby oznaczonych fragmentów. Parametr ten jednak nie funkcjonuje w systemie JSA od wdrożenia wersji v1.2.2 w dniu 19 grudnia 2019 roku, i został zastąpiony przez nowy parametr „Poziom podobieństwa".

Poziom podobieństwa to funkcjonalność, która służy do dokładnego określenia przez jednostkę czy i w jakim stopniu system ma wskazywać fragmenty będące sparafrazowanym odpowiednikiem znalezionego źródła. Wpływa on więc na wrażliwość algorytmu detekcji na zbieżność tekstów. Należy pamiętać, że jest to jedynie jeden z wielu w algorytmie detekcji i nie wpływa na niego bezpośrednio. System określa wskazania fragmentów w trakcie badania, zaś zmiany parametru wpływają na to, czy jednostka chce zobaczyć je wszystkie. Dokładniejszy opis działania można znaleźć w podrozdziale 2.3.

W związku z licznymi sugestiami użytkowników będącymi promotorami, z początkiem lutego 2019 roku wprowadzono możliwość przeliczania wyniku badania na raporcie, co pozwoliło promotorom wykluczać z wyniku badania takie fragmenty, które uznali za zasadnie użyte treści.

Kolejne tygodnie przyniosły dalsze poprawki i udoskonalenia odpowiadające na zgłaszane przez użytkowników potrzeby. Były to m.in.:

- dodanie widoczności badań przesłanych przez API w interfejsie użytkownika dla promotora;
- dodanie opcji wybrania raportu szczegółowego bez analizy tekstu;
- poprawa obsługi badań z plikami większymi niż limit 15 MB;
- usunięcie limitu trzech prób badania pracy.

Długo wyczekiwaną zmianą było udostępnienie pełnej treści źródła, do którego podobieństwo fragmentu odnalazł system. Wgląd w pełną treść źródła został przewidziany ze względów bezpieczeństwa tylko dla użytkowników z uprawnieniami promotorskimi. Dostęp do treści pozwolił promotorom w pełni zweryfikować rzeczywiste znaczenie i stopień podobieństwa tekstów.

Wiele istotnych zmian przyniósł dla systemu maj 2019 roku. Przede wszystkim zmieniono kolejność przeszukiwania baz referencyjnych. Wcześniej dane z prac sprawdzanych w JSA były porównywane do danych z następujących repozytoriów we wskazanej kolejności:

1. Baza aktów prawnych;
2. Polski Korpus Internetu oraz elementy światowego internetu (Wikipedia dla zasobów anglojęzycznych, hiszpańskich, niemieckich, rosyjskich, włoskich);
3. ORPPD tj. Ogólnopolskie Repozytorium Pisemnych Prac Dyplomowych.

Zmiany zniosły wyżej wymienioną kolejność na rzecz porównywania pod względem jakości i stopnia podobieństwa do każdej z baz. Uzupełniono również listę wspieranych języków obcych o kolejne najbardziej popularne w ORPPD (angielski, francuski, hiszpański, niemiecki, polski, rosyjski, ukraiński, włoski, inny).

W maju 2019 roku przeprowadzono również kolejną turę badań użyteczności systemu, w tym 15 wywiadów wraz z testami systemu. Respondenci pochodzili z następujących czterech grup:

- Administratorzy systemów na uczelniach wyższych – 3 osoby;
- Promotorzy prac dyplomowych – 4 osoby;
- Recenzenci prac dyplomowych – 4 osoby;
- Pracownicy dziekanatu, obsługujący system JSA – 4 osoby.

Respondenci w obrębie grupy pochodzili z różnych uczelni.

Jednolity System Antyplagiatowy został przez respondentów oceniony pozytywnie. Promotorzy, czyli grupa, która ma najwięcej zadań i obowiązków do wykonania w systemie, miała częściowo problem z wykrywaniem manipulacji w pracy czy działaniem na tekście pracy. Osoby, które nie uczestniczyły w szkoleniu z obsługi systemu, nie potrafiły korzystać z części systemu. Te problemy wynikały z braku przeszkolenia oraz braku wcześniejszego kontaktu z systemem. Dla porównania użytkownicy, którzy mieli doświadczenia z innymi systemami, podkreślali, że system JSA jest lepszy od konkurencji, zarówno na płaszczyźnie wykrywania plagiatów, jak i w kwestii użyteczności czy funkcjonalności. Dodatkowo respondenci doceniali fakt, że system jest darmowy.

Na podstawie badań można powiedzieć, że od czasu wprowadzenia JSA uczelnie wyższe dzielą się na te, które zrezygnowały już z systemu antyplagiatowego, z którego dotychczas korzystały i na te, które korzystają z obydwu systemów. Zaprzestanie korzystania z innych systemów antyplagiatowych wynika z potrzeby wprowadzenia jednolitego sposobu badania prac względem innych uczelni jak i szukania oszczędności na uczelniach, a JSA jest systemem bezpłatnym. Dotychczasowe rozwiązania uczelniane służyły jednak nie tylko do weryfikacji prac licencjackich i magisterskich (do czego obecnie wykorzystywany jest JSA), sprawdzane były również prace zaliczeniowe, podyplomowe oraz doktorskie. Uczelnie rozumieją możliwości jakie daje im Jednolity System Antyplagiatowy na dwa sposoby. Część z nich nie weryfikuje pod kątem plagiatu niektórych typów prac, które sprawdzane były do tej pory, ponieważ nie mają takiej możliwości systemowej. Dotyczy to np. prac podyplomowych lub zaliczeniowych. Dla drugiej części uczelni niedostosowanie systemu do tego typu prac nie stanowi problemu. Co ciekawe w niektórych przypadkach rozpoczęcie korzystania z JSA spowodowało, że teraz badanych jest więcej prac niż wcześniej, ponieważ nie ma limitów na liczbę wgranych prac.

Różnica w pracy w systemie JSA w stosunku do dotychczasowych komercyjnych rozwiązań jest taka, że przed wprowadzaniem na uczelnie systemu JSA promotorzy dostawali do wglądu raporty tylko i wyłącznie wtedy, gdy wynik badania był negatywny. Respondentom, którzy używali wcześniej innych systemów antyplagiatowych, system JSA wydaje się bardziej intuicyjny i czytelniejszy. Dodatkowo doceniane są algorytmy wykrywające plagiaty, a respondenci wierzą, że jest on w stanie wykryć więcej plagiatów niż inne systemy.

29 maja wdrożono nową wersję, w ramach której dodano dostęp (tj. logowanie) dla instytucji uprawnionej do nadawania stopnia doktora. Umożliwiło to także wpuszczenie jednostek naukowych na środowisko produkcyjne. W związku z nowym typem badanych prac wprowadzono szereg zmian pod tym kątem, także w zakresie integracji REST API.

Dnia 31 maja 2019 roku, dokładnie o godzinie 13:00 przekroczyliśmy próg 100 tysięcy przebadanych prac w JSA. Ta magiczna bariera została przebita zaskakująco szybko, przechodząc nasze najśmielsze oczekiwania. Od momentu uruchomienia środowiska produkcyjnego do końca maja uzyskaliśmy takie statystyki:

- Maksymalna liczba badań w ciągu doby: 3.712;
- Przeciętny czas badania: 3,93 min;
- Średnia liczba badań przypadająca na promotora: 4,85;
- Liczba zgłoszonych prac inżynierskich: 35.032;
- Liczba zgłoszonych prac magisterskich: 33.526;
- Liczba zgłoszonych prac licencjackich: 29.142;
- Liczba wszystkich uczelni, które rozpoczęły korzystanie z JSA: 341;
- Liczba użytkowników: 37.580;
- Liczba zgłoszeń helpdesk: 2.821.

Z dniem 30 czerwca 2019 roku projekt „Przygotowanie Jednolitego Systemu Antyplagiatowego i jego wdrożenie oraz obsługa" został oficjalnie zakończony i tym samym wszedł w fazę utrzymania oraz dalszego rozwoju. Główne zadanie polegające na powstaniu i oddaniu do użytkowania systemu JSA zostało wykonane z powodzeniem. Dalsze prace zespołu przeszły w ograniczoną fazę utrzymania, co oznacza, że nowe funkcjonalności od zakończenia projektu są realizowane w ramach zlecenia zmian czy aktualizacji cyklicznych.

UTRZYMANIE SYSTEMU

Po sesji letniej, w drugiej połowie lipca 2019 roku, system JSA otrzymał kolejne znaczące ulepszenie funkcjonalne. Przygotowano i wdrożono rozpoznawanie grafik wektorowych .svg w plikach .docx i .odt i pomijanie ich przy wyodrębnianiu tekstu z badanych plików. Była to bardzo znacząca zmiana, wyczekiwania szczególnie przez pracowników uczelni technicznych.

Od września 2019 roku rozpoczęły się intensywne przygotowania JSA zmierzające do aktualizacji funkcjonalności w związku z powstającą nową wersją systemu POL-on 2.0. W wyniku trwającej wówczas migracji danych do nowej wersji POL-onu, dotychczasowe logowanie Administratorów do JSA przez opcję „LOGOWANIE POL-ON" zostało na stałe wyłączone, co nastąpiło równolegle z wyłączeniem tej usługi w systemie POL-on.

W tym samym czasie wprowadzono optymalizację listy badań poprzez wprowadzenie filtrów domyślnych. Wdrożenie to miało na celu usprawnienie pracy na dużych jednostkach prowadzących liczne nadania stopni i tytułów. Była to forma wczesnej archiwizacji

badań. W ten sposób osoby posiadające dostęp do dużej liczby badań na swojej liście mogą swobodnie i sprawnie pracować na tych najbardziej aktualnych.

W październiku zdecydowaliśmy się usystematyzować proces wdrażania zmian dla integracji API JSA. W związku z tym zastosowano dwutygodniowy okres karencji zmian. Każde wdrożenie związanie z integracją systemów miało od tej pory przez dwa tygodnie być testowane na wersji systemu demo. Taka zmiana pozwoliła użytkownikom systemów zewnętrznych na odpowiednie przygotowanie się do zbliżających się wdrożeń. Pierwszą w taki sposób wprowadzoną aktualizacją było zwiększenie limitu dla przesyłanego pliku do badania do wielkości 20 MB, jak również dostosowanie listy jednostek i kierunków do nowej struktury POL-onu 2.0.

Ostatnie miesiące 2019 roku były bardzo intensywne w pracach nad JSA. Listopad rozpoczął się ogłoszeniem testów funkcjonalności podłączania wewnętrznych zasobów instytucji do systemu JSA jako dodatkowej bazy referencyjnej wykorzystywanej w badaniu antyplagiatowym. Baza ta ma służyć potrzebom tej instytucji, która ją przygotuje i zintegruje z JSA, a dla systemu stanowi dodatkowe źródło w badaniu antyplagiatowym. Źródło podobieństwa, jeżeli zostanie wyszukane przez JSA, jest prezentowane w taki sposób i z takim zakresem danych jakie będą zwrócone przez bazę instytucji. Jeżeli źródło zostanie przygotowane w taki sposób, że będzie posiadało dane np. typ pracy, dane autora i tytuł, to takie informacje wskazane zostaną także w metadanych źródła (np. Typ: praca semestralna, Autor: Luiza Szukalska, Tytuł: Nowoczesne środki komunikacji i ich wpływ na relacje społeczne). Zakres danych, które należy wskazać przy źródle, nie jest określany przez JSA i oznacza to, że – poza obowiązkowymi polami (tj. ID dokumentu, tytuł, treść) – każda instytucja może przygotować bazę w indywidualny sposób.

Pod koniec roku 2019 rozpoczęto etapowe oddawanie dla użytkowników kolejnych pogłębionych statystyk aktywności. Z końcem roku udostępniono anglojęzyczny interfejs użytkownika. Tłumaczenie ekranów systemowych przewidziane zostało głównie dla użytkowników z rolą Promotor. Dodano również bazę dokumentów OpenAccess jako uzupełnienie baz referencyjnych. Zaplanowano również trzyetapowe ulepszenie funkcjonalne systemu. Pierwsze zmiany dotyczyły źródeł przeszukiwanych przez system w poszukiwaniu podobieństw. Do baz referencyjnych dołączone zostały obszerne zasoby dokumentów OpenAccess, które poszerzyły źródła internetu. Oprócz tego, po okresie karencji, udostępniono możliwość podłączenia przez każdą instytucję jej indywidualnej bazy referencyjnej. Drugi etap prac miał na celu zmianę priorytetyzacji znalezionych fragmentów według stopnia ich podobieństwa do oryginału. Oznacza to, że w pierwszej kolejności system zaczął wskazywać fragmenty najbardziej podobne między sobą oraz takie, które pochodzą z możliwie jak najmniejszej liczby źródeł.

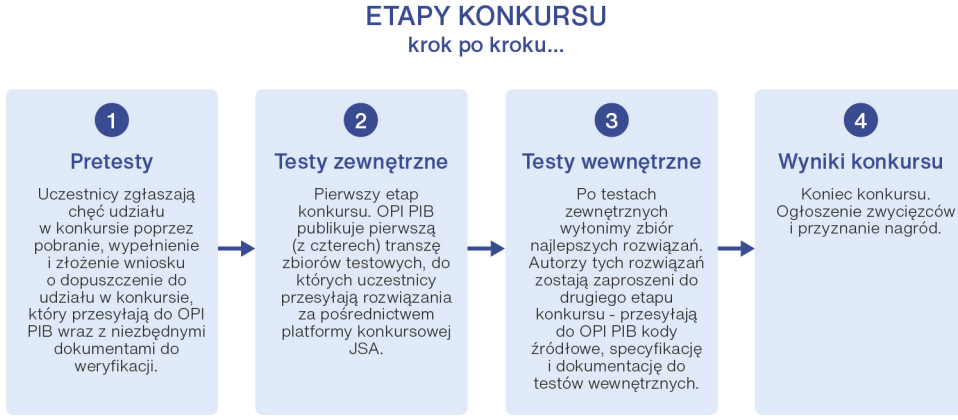
W ostatniej fazie wprowadzono funkcjonalność wskazywania możliwych poprawnych cytowań wykorzystanych w sprawdzanej pracy. Dzięki niej ułatwione zostało wykluczanie przez promotora pojedynczych oraz grupowych cytatów z wyniku. To właśnie w tym czasie nastąpiła zmiana sterowalnego parametru „siły podobieństwa tekstu" na parametr „poziom podobieństwa", o którym wcześniej była mowa w tym rozdziale.

Rok 2020 upłynął pod znakiem dostosowywania systemu JSA do POL-onu 2.0. Poza mniejszymi zmianami obejmującymi m.in. wprowadzenie sekcji wniosków uzupełnianej przez interfejs użytkownika, wprowadzonej do elektronicznej wersji raportu z badania. Zgodnie z harmonogramem integracji POL-onu 2.0 z JSA, nową wersję wgrano na środowisko testowe w dniu 29 kwietnia, zaś na środowisko produkcyjne trafiła ona 14 maja. Zaktualizowanie integracji było niezbędne aby te dwa systemy mogły wciąż bez przeszkód funkcjonować na tych samych danych. Nowa architektura POL-onu 2.0 fundamentalnie zmieniła identyfikację obiektów w JSA. Zmiany dotyczyły w równej mierze interfejsu użytkownika jak i integracji REST API. Podstawową zmianą były nowe identyfikatory obiektów, na których oparte zostały dane w POL-onie 2.0. Dotyczyło to instytucji, jednostek, filii, kierunków (obecnie uruchomień), pracowników oraz studentów. W przypadku API JSA oznaczało to zmianę również źródła pobierania przez użytkowników wszystkich identyfikatorów. Integracja z POL-onem 2.0 wpłynęła także na zmianę sposobu rozpoznawania użytkowników (nowe identyfikatory) czy administrowanie ustawieniami parametrów oraz uprawnieniami użytkowników (rozróżnienie jednostek technicznych), oraz na pobieraną listę kierunków, generowanie raportów, czy listy studentów i doktorantów. Niemniej jednak integracja z nowszą wersją systemu POL-on 2.0. powiodła się, dzięki czemu JSA pozostało spójne z Zintegrowanym Systemem Informacji o Szkolnictwie Wyższym i Nauce POL-on.

W pierwszym kwartale 2021 udało się poszerzyć bazę referencyjną JSA o usługi dostępne w ramach Azure Cloud. Dzięki temu użytkownicy systemu zyskali porównywanie tekstów nie tylko z polską częścią internetu ale ogólnodostępnymi źródłami światowego internetu. W kolejnym wdrożeniu nowej wersji systemu udało się podłączyć tłumaczenie maszynowe tekstów wysyłanych do badania.

1.3. Konkurs JSA

Jednym z ważnych etapów budowy systemu był konkurs na algorytmy i kody źródłowe różnych rozwiązań z dziedziny walki z plagiaryzmem. Składał się on z 4 etapów i trwał do 30 września 2017 roku. W celu optymalnego zarządzania całym wydarzeniem, przepływem rozwiązań oraz ich ewaluacji powstała dedykowana platforma internetowa dostępna pod adresem konkurs-jsa.opi.org.pl.



Ilustracja 1.4. Konkurs JSA – etapy

Konkurs JSA podzielony był na cztery etapy:

1. Pretesty (od 25 kwietnia 2017 r. do końca maja 2017 r.); w tym czasie OPI PIB opublikowało:
   a. platformę konkursową umożliwiającą proces rejestracji,
   b. próbki zbiorów danych do czterech zadań,
   c. formaty rozwiązań, które przyjmuje platforma konkursowa,
   d. opisy miar ewaluacji do poszczególnych zadań,
   e. mechanizm ewaluacji wyników w ramach każdego zadania.

   Na tym etapie nastąpiło zgłaszanie uczestników Konkursu. Uczestnikiem mogła być osoba fizyczna, konsorcjum lub firma, mająca siedzibę na terenie Polski, która złożyła wniosek o dopuszczenie do udziału w konkursie. Wypełniony i podpisany wniosek należało przesłać do OPI PIB.

   Organizator Konkursu przesłał na adres mailowy podany we wniosku o dopuszczenie do udziału w Konkursie interaktywny formularz Wniosku, umożliwiający Uczestnikowi wykonanie Rejestracji i utworzenie Konta w ramach drugiej fazy zgłoszenia udziału w Konkursie.

   Po otrzymaniu zgłoszenia specjaliści z OPI PIB dokonali jego weryfikacji. Po pozytywnej weryfikacji nastąpiła aktywacja konta uczestnika na platformie konkursowej. Wówczas użytkownik uzyskał dostęp do podstron z opisami zadań, próbkami danych oraz specyfikacją formatów.

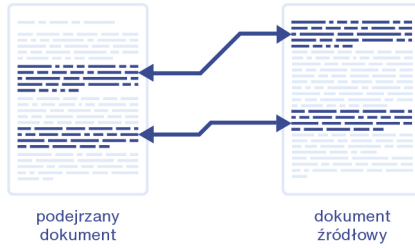
2. Testy zewnętrzne (od 1 czerwca 2017 r. do 31 sierpnia 2017 r.) składały się z:
   a. opublikowania przez OPI PIB pierwszej transzy (z czterech) zbiorów testowych, do których uczestnicy konkursu powinni przesyłać rozwiązanie za pośrednictwem platformy konkursowej JSA;
   b. przesłania przez uczestników za pośrednictwem platformy konkursowej JSA rozwiązań do opublikowanych transz zbiorów testowych. Zawsze brano pod uwagę najbardziej aktualne rozwiązanie dla danej transzy, czyli: jeżeli aktualnie opublikowana była np. transza 2 to uczestnik był zobowiązany zamieścić rozwiązanie dla tej transzy, natomiast jeżeli uczestnik nie wgrał do platformy rozwiązań do tej transzy, jego wynik nie był publikowany oraz jego udział nie był brany pod uwagę w końcowej ocenie;
   c. wymagany był udział w każdej fazie ze względu na osiągnięcie minimalnego założonego przez Organizatorów progu jakościowego. Ocenie podlegały wszystkie fazy w ramach każdego Zadania Etapu I Konkursu. W każdym Zadaniu Etapu I Konkursu wygrywały tylko trzy najwyższe wyniki uzyskane w ostatniej fazie.
3. Testy wewnętrzne (okres po zakończeniu testów zewnętrznych). Po zakończeniu testów zewnętrznych został wytypowany zbiór najlepszych rozwiązań – zgodnie z zapisami regulaminu konkursu. Następnie, w ten sposób wytypowani uczestnicy konkursu przesyłali do OPI PIB rozwiązania w postaci kodów źródłowych, specyfikacji oraz dokumentacji technicznej. Przesłane materiały umożliwiły operatorom konkursu uruchomienie rozwiązań w wewnętrznym środowisku OPI PIB w ramach dedykowanej w tym celu infrastruktury informatycznej. Operatorzy przetestowali przesłane rozwiązania i dokonali weryfikacji:
   a. zgodności wyników uzyskanych na platformie konkursowej oraz wyników uzyskanych w trakcie testów,
   b. skalowalności przesłanych rozwiązań, poprzez manipulowanie parametrami maszyny wirtualnej, takimi jak: ilość danych, wielkość pamięci operacyjnej, liczba dostępnych rdzeni.
4. Ogłoszenie zwycięzców konkursu.

W ramach Konkursu były przygotowane cztery zadania, o których więcej niżej.

1.3.1. Zadanie 1: Wykrywanie nadużyć między dokumentami

Na podstawie dostarczonych dwóch dokumentów – źródłowego i podejrzanego, należało wykryć wszystkie zapożyczenia dokumentu źródłowego w podejrzanym. Możliwe jest istnienie czterech różnych zapożyczeń, w różnych wariantach:

1. kopiuj-wklej,
2. kopiuj-wklej + zmiana szyku,
3. kopiuj-wklej + parafraza,
4. kopiuj-wklej + parafraza + zmiana szyku.



Ilustracja 1.5. Konkurs JSA – zadanie nr 1

Danymi wejściowymi był zbiór par dokumentów: podejrzany o plagiat oraz źródło dla plagiatu. Para dokumentów zostanie zapisana w formacie tekstowym: suspiciousXXXX.txt, sourceXXXX.txt, gdzie XXXX oznacza numer pary, pierwszy to dokument, w którym popełniono plagiat, drugi to dokument źródłowy, z którego nastąpiły zapożyczenia.

Dla zadanej pary plików uczestnik konkursu wygenerował plik XML w następującym formacie:

Kod 1.1. Przykład pliku xml z wynikami działania zgłoszonego rozwiązania

```
<?xml version="1.0" encoding="UTF-8"?>
    <alignment document="suspiciousXXXX.txt" source="sourceXXXX.txt">
    <passage documentFrom="1" documentTo="4" sourceFrom="7" sourceTo="10"/>
    <passage documentFrom="2" documentTo="8" sourceFrom="2" sourceTo="8"/>
</alignment>
```

Tag xml o nazwie *passage* oznacza znaleziony plagiat. Znaczenie jego atrybutów:

1. documentFrom – indeks początkowy znalezionego fragmentu w dokumencie podejrzanym,
2. documentTo – indeks końcowy znalezionego fragmentu w dokumencie podejrzanym,
3. sourceFrom – indeks początkowy znalezionego fragmentu w dokumencie źródłowym,
4. sourceTo – indeks końcowy znalezionego fragmentu w dokumencie źródłowym.

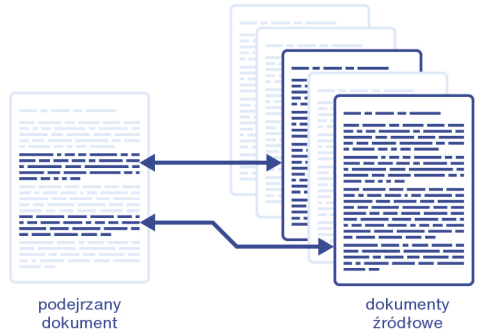
Zawartość była umieszczona w pliku o nazwie suspiciousXXXX-sourceXXXX.xml. Dla całego zadania był to zbiór plików XML, który następnie spakowano do archiwum ZIP.

Do mierzenia poprawności przesłanego rozwiązania używane były następujące miary: precyzja [20], kompletność [20], granularność [17], plagdet (główna miara) [17].

1.3.2. Zadanie 2: Wykrywanie podejrzanych dokumentów

Na podstawie dostarczonego dokumentu należało wytypować z grupy dokumentów źródłowych te, w stosunku do których dany dokument jest podejrzany o zapożyczenia.

Należy przyjąć, że zbiór dokumentów źródłowych (zwany dalej korpusem referencyjnym) jest na tyle duży, że przeszukiwanie wszystkich dokumentów będzie czasowo nieopłacalne, dlatego nie należy korzystać z rozwiązań typu *brute-force* opartych na rozwiązaniu zadania pierwszego. Docelowo uczestnicy konkursu w ramach zadania 2 dostarczali dwa rozwiązania: (a) rozwiązanie budujące model efektywnej reprezentacji korpusu referencyjnego dla celów wydajnego wyszukiwania informacji oraz (b) rozwiązanie pracujące na tym modelu w celu detekcji potencjalnych podejrzanych dokumentów. Oba rozwiązania (a i b) musiały być wydajne, w szczególności skalowalne, np. wykorzystujące obliczenia rozproszone lub architekturę wielordzeniową. Poziom skalowalności rozwiązań został zweryfikowany w ramach testów wewnętrznych.



Ilustracja 1.6. Konkurs JSA – zadanie nr 2

Danymi wejściowymi były dwa zbiory dokumentów: dokumenty podejrzane (o nazwie formatu suspiciousXXXX.txt), które zostały utworzone na podstawie dokumentów źródłowych, oraz dokumenty źródłowe, wśród których będą typowane źródła plagiatów.

Dla każdego podejrzanego dokumentu uczestnik konkursu wygenerował plik XML w następującym formacie:

Kod 1.2. Przykład pliku xml z wynikami działania zgłoszonego rozwiązania

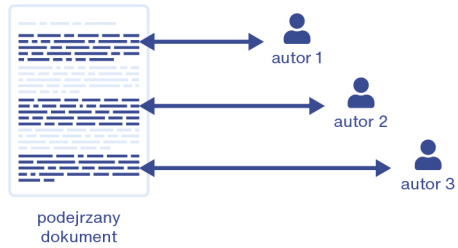
```
<?xml version="1.0" encoding="UTF-8"?>
<external document="suspiciousXXXX.txt">
    <potential source="sourceXXXX.txt" />
    <potential source="sourceXXXX.txt" />
    <potential source="sourceXXXX.txt" />
    <potential source="sourceXXXX.txt" />
</external>
```

Tag xml o nazwie *potential* oznacza znalezione źródło plagiatu. Dla każdego podejrzanego dokumentu użytkownik utworzył plik XML o nazwie suspiciousXXXX.xml oraz formacie opisanym powyżej. Rozwiązaniem zadania był zbiór plików XML, który następnie spakowano do archiwum ZIP.

Do mierzenia poprawności przesłanego rozwiązania używana była miara F1 [20].

1.3.3. Zadanie 3: Wykrycie potencjalnego plagiaryzmu w ramach pojedynczego dokumentu

Na podstawie danego dokumentu należało stwierdzić, które części były pisane przez innego autora lub autorów (przy założeniu, że praca ma głównego autora). Celem tego zadania było profilowanie (wyznaczanie cech stylu autora na podstawie tekstu oraz wyznaczenie, które bloki tekstu mają inny profil niż profil głównego autora).



Ilustracja 1.7. Konkurs JSA – zadanie nr 3

Danymi wejściowymi był zbiór podejrzanych dokumentów (każdy element tego zbioru był tworzony z innych dokumentów źródłowych). Nazwa każdego z dokumentów miała format: suspiciousXXXX.txt, gdzie XXXX oznacza numer lub identyfikator dokumentu.

Dla każdego podejrzanego dokumentu uczestnik konkursu wygenerował plik XML w następującym formacie:

Kod 1.3. Przykład pliku xml z wynikami działania zgłoszonego rozwiązania

```
<?xml version="1.0" encoding="UTF-8"?>
<authorship document="suspiciousXXXX.txt">
    <block from="123" to="128" />
    <block from="234" to="238" />
    <block from="345" to="349" />
</authorship>
```

Tag xml o nazwie *block* oznacza blok tekstu, którego autorem jest jedna osoba. Znaczenie jego atrybutów:

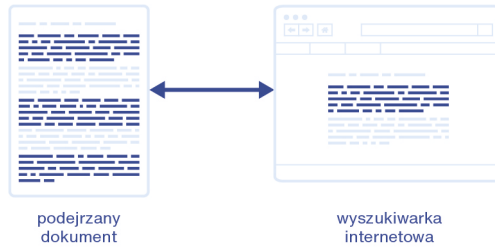
1. from – indeks początkowy znalezionego bloku w dokumencie podejrzanym,
2. to – indeks końcowy znalezionego bloku w dokumencie podejrzanym.

Znalezione bloki powinny określać, które fragmenty zostały napisane przez autora innego niż główny autor danego dokumentu (pasaże podejrzane o plagiat). Jako rozwiązanie zadania uczestnik konkursu musiał wygenerować zbiór plików XML w tym formacie, z których nazwa każdego miała format suspiciousXXXX.xml, który następnie spakował do archiwum ZIP.

Do mierzenia poprawności przesłanego rozwiązania używana była miara plagdet [17].

1.3.4. Zadanie 4: Wykrywanie podejrzanych stron internetowych

Na podstawie zadanego dokumentu należało wytypować z korpusu stron internetowych takie strony, z których dany dokument jest podejrzany o zapożyczenia. Zakładamy, że korpus stron internetowych (zwany dalej referencyjnym obrazem internetu) jest na tyle duży, że przeszukiwanie wszystkich stron będzie czasowo nieopłacalne, dlatego nie należy korzystać z rozwiązań typu brute-force. Docelowo uczestnicy konkursu musieli dostarczyć dwa rozwiązania w ramach realizacji zadania 4: (a) rozwiązanie budujące model efektywnej reprezentacji obrazu internetu dla celów wydajnego wyszukiwania informacji oraz (b) rozwiązanie pracujące na tym modelu w celu detekcji potencjalnie podejrzanych stron. Oba rozwiązania (a i b) musiały być wydajne, w szczególności skalowalne, np. wykorzystujące obliczenia rozproszone lub architekturę wielordzeniową. Poziom skalowalności rozwiązań został zweryfikowany w ramach testów wewnętrznych.



Ilustracja 1.8. Konkurs JSA – zadanie nr 4

Dane wejściowe dostarczone zostały w dwóch częściach: (1) zbioru podejrzanych dokumentów oraz (2) źródeł internetowych w postaci korpusu surowych dokumentów HTML.

Dla każdego podejrzanego dokumentu uczestnik konkursu wygenerował plik XML w następującym formacie:

Kod 1.4. Przykład pliku xml z wynikami działania zgłoszonego rozwiązania

```
<?xml version="1.0" encoding="UTF-8"?>
<web document="suspiciousXXXX.txt">
    <potential source="documentXXXX.html" />
    <potential source="documentXXXX.html" />
    <potential source="documentXXXX.html" />
    <potential source="documentXXXX.html" />
</web>
```

Tag xml o nazwie *potential* oznacza źródło, które posłużyło do tworzenia podejrzanego dokumentu html.

Dla każdego podejrzanego dokumentu uczestnik konkursu utworzył plik XML o nazwie suspiciousXXXX.xml oraz formacie opisanym powyżej. Rozwiązaniem zadania był zbiór plików XML, który następnie spakowano do archiwum ZIP.

Do mierzenia poprawności przesłanego rozwiązania używana była miara F1 [20].

1.4. Geneza powstania oraz zakres funkcjonalny systemu

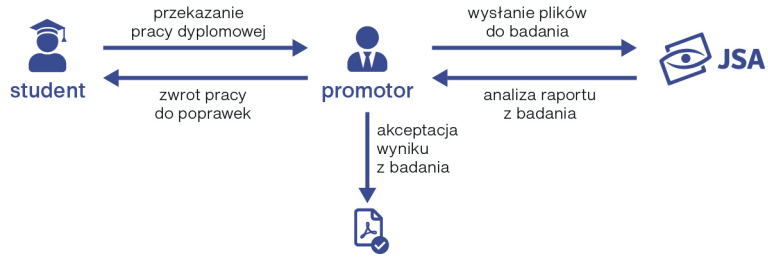
1.4.1. Geneza powstania JSA

Jeszcze w 2017 roku w polskim szkolnictwie wyższym korzystanie z systemów antyplagiatowych było wewnętrznie zróżnicowane. Z badań wynika, że część uczelni w Polsce stanowiąca niemal połowę w ogóle nie posiadała dostępu do usług antyplagiatowych, a reszta zmuszona była kupować licencje z systemów komercyjnych np. plagiat.pl, Genuino, OSA czy zagraniczne Turnitin czy Urkund. Istniała także praktyka samodzielnego przygotowywania takich programów przez uczelnie. Korzystanie z różnych systemów skutkowało rozbieżnością w procesie oceny antyplagiatowej prac. Każdy system działał na różnych bazach referencyjnych, co skutkowało bardzo różnymi wynikami badań, a w efekcie było powodem nierówności w traktowaniu dyplomantów. Większość systemów antyplagiatowych była dostępna dla studentów co nie miało przełożenia na lepszą jakość pisanych tekstów tylko iteracyjnej nauki „jak obejść system", aby wynik badania był korzystny. Sytuacja ta była bardzo silnie odczuwalna w środowisku szkolnictwa wyższego, gdzie każda uczelnia działała według innej procedury przygotowania prac dyplomowych przed obroną.

W związku z powyższym, pojawiła się potrzeba stworzenia spójnego programu antyplagiatowego, który ujednolici proces sprawdzania prac dyplomowych na wszystkich uczelniach w kraju. Jeden wspólny system, który każdy badany tekst będzie analizował według tego samego algorytmu wykorzystując za każdym razem te same bazy referencyjne, a przy tym zapewni w pełni skalowalne działanie systemu na dużych zbiorach bazodanowych. Jednolity System Antyplagiatowy (JSA) wydaje się być doskonałą odpowiedzią na powyższe problemy.

JSA stał się podstawowym narzędziem kadry akademickiej wykorzystywanym na uczelniach w celu ochrony praw autorskich tekstów, z których czerpią studenci piszący prace dyplomowe. Jest to innowacyjne podejście w dziedzinie detekcji plagiaryzmu w kraju. Dzięki zapewnieniu przez MNiSW bezpłatnego dostępu dla każdego podmiotu naukowego projekt ten ma zasadnicze znaczenie dla podniesienia jakości nauki i szkolnictwa wyższego, których rezultatem ma być autorska praca naukowa. Znosi bariery nie tylko finansowe, ale także organizacyjne i wyrównuje poziom szkolnictwa wyższego oraz wymagania stawiane pracom dyplomowym. Wskazywany jest również jako ważny czynnik podnoszący jakość prac badawczych i rozpraw naukowych podejmowanych i tworzonych w kraju.

Zgodnie z obowiązującymi przepisami prawa każda pisemna praca dyplomowa, przed zatwierdzeniem jej do obrony, powinna zostać zbadana przez system JSA pod kątem naruszeń prawa autorskiego. W tym celu student, zanim przystąpi do egzaminu obrony pracy dyplomowej, powinien przekazać promotorowi pracy odpowiednio przygotowane pliki. Po przeanalizowaniu przez system wskazanych plików generowany jest dla promotora raport z badania. Na podstawie tego dokumentu student dopuszczany jest lub nie do egzaminu obrony pracy. Podobnie jest z pracą doktorską, która także musi być przebadana przez JSA przed egzaminem obrony i zakończeniem procesu nadania stopnia doktorskiego. Tak wygląda proces od strony teorii prawa natomiast rzeczywistość wydaje się bardziej skomplikowana gdy tylko zacznie się analizować możliwości systemu.



Ilustracja 1.9. Podstawowy scenariusz wykorzystania JSA

1.4.2. Zakres funkcjonalny JSA

Funkcjonalności operacyjne systemu:

1. Składanie zleceń do badania prac pod kątem plagiatu.
2. Możliwość dodania kolejnych prób badania do zlecenia.
3. Przeglądanie listy zleceń i statusów badania.
4. Modyfikacja danych zlecenia (dane o pracy badanej: tytuł pracy, autor, promotor itp.).
5. Prezentacja wyników badania, czyli raportu:
   a. Raport podstawowy – dane statystyczne oraz źródła zapożyczeń.
   b. Raport szczegółowy – raport podstawowy rozszerzony o szczegóły zapożyczeń i nadużyć w tekście pracy badanej.
6. Przesyłanie linku z dostępem do strony z wynikiem badania antyplagiatowego.
7. Przeliczanie wyników poprzez wykluczanie fragmentów, wykazanych w raporcie jako zapożyczenia.
8. Raportowanie sprawozdawcze (raporty zbiorcze, okresowe itp.).

Funkcjonalności administracyjne w systemie:

1. Konfiguracja wybranych parametrów dla uczelni – konfiguracja Ustawień.
2. Nadawanie uprawnień użytkownikom w ramach podgrup (jednostek) uczelni.
3. Zarządzanie użytkownikami – dodawanie, usuwanie, edycja.

Opcja dostępu do JSA:

1. Interfejs użytkownika oraz zarządzanie uprawnieniami i kontami użytkowników.
2. Integracja z systemem uczelnianym po API (w technologii REST).

1.4.3. Interfejs graficzny użytkownika

Interfejs graficzny systemu jest zaprojektowany tak, aby użytkownik w szybki i łatwy sposób mógł wykonać badanie tekstu lub przeanalizować raport dostępny w systemie.

Po zalogowaniu do systemu wyświetlana jest główna strona użytkownika. W zależności od posiadanej roli jest to lista użytkowników (dla administratorów systemu) lub lista badań (dla wszystkich pozostałych ról).

W widoku listy badań można podejrzeć szczegóły metryki każdego z wykonanych badań. Domyślnie, lista posortowana jest od ostatnio wykonanego do najstarszego badania. W zależności od bieżącego statusu badania oraz od uprawnień użytkownika może on przejść z tego widoku do konkretnego raportu z badania, wydruku lub udostępnienia badania. Filtry i wyszukiwarka ułatwiają odnalezienie konkretnego rekordu w liście.

W ramach jednego badania możliwe jest wykonanie wielu prób. Oznacza to, że nowa próba jest kolejnym podejściem do badania tych samych plików pracy. Najczęściej dzieje się tak, kiedy student wraca do opiekuna pracy z poprawionym tekstem i należy zbadać go ponownie. Każda próba widoczna jest w szczegółach badania co umożliwia szybkie porównania wyników każdej z nich.

Promotor lub odpowiedni pracownik administracyjny uczelni z rolą Biuro, może założyć nowe badanie w systemie, czyli wypełnia formularz i wysyła pliki pracy dyplomowej do analizy. Do badania można dodać więcej niż jeden plik, jednak każdy z nich musi mieścić się w limicie wielkości 20 MB (maksymalnie 60 MB łącznie) oraz każdy z nich musi mieć akceptowane rozszerzenie tj. .txt, .pdf, .doc, .docx, .odt, .rtf. Należy podkreślić, że żadne pliki, które są przesyłane do JSA, nie są zapisywane w bazie referencyjnej systemu. Jedynie wyodrębniony jest z nich sam tekst i w takiej formie analizowany jest dalej przez system.

Po każdym etapie analizy tekstu przez system zmienia się status badania widoczny na liście. Tabela 1.1 zawiera ich szczegółowe opisy.

Tabela 1.1. Tabela statusów badania

Status	Opis
Zarejestrowane	Status wskazuje na badanie, w którym została zarejestrowana tylko metryka badania, bez pliku(ów) pracy dyplomowej
W kolejce	Status informuje, że badanie wraz z plikiem(ami) pracy dyplomowej zostało wysłane do analizowania pod kątem plagiatu i znajduje się w kolejce oczekującej do badania
Badane	Status informuje, że proces analizy tekstu pracy dyplomowej jest w trakcie realizacji
Wykonane	Status informuje, że analizowanie pracy dyplomowej pod kątem plagiatu zostało zakończone
Zaakceptowane	Status wskazuje, że raport z badania pracy dyplomowej został zaakceptowany przez promotora
Błąd	Status informuje, że wystąpił błąd podczas badania pracy, zaś to badanie zostanie automatycznie powtórzone po upływie 15 minut

W formularzu badania znajdują się pola, które należy uzupełnić w celu późniejszej identyfikacji raportu. Dla usprawnienia pracy w systemie, w niektórych polach formularza zastosowano gotowe listy wyboru jednostek, studentów oraz pracowników uczelni. Dane pochodzą ze zintegrowanej bazy systemu POL-on i są na bieżąco aktualizowane. Wszystkie wprowadzone dane utworzą finalnie metrykę raportu.

Po zakończeniu badania system wysyła informacje do promotora oraz innych powiązanych z badaniem użytkowników. Następnie można je udostępnić innym użytkownikom, ale tylko tym, którzy posiadają rolę Promotor i posiadają konta w kontekście tej samej instytucji.

Raport powinien zostać dokładnie przeanalizowany. Jest to dokument, który dostarcza wielowymiarowej informacji o zbadanym tekście. W raporcie z badania system nie tylko wskazuje zbieżność tekstów ale także analizuje czy w pracy użyto słów pochodzących z innego języka, lub wprowadzono znaki z innego alfabetu, a także jest w stanie wskazać fragmenty pracy, które zostały napisane innym stylem. Dlatego analiza raportu przez promotora powinna być bardzo dokładna i przeprowadzona starannie, zgodnie z wytycznymi twórców JSA. Na jego podstawie promotor podejmuje decyzję o dopuszczeniu lub niedopuszczeniu pracy do obrony.

Wyodrębniony tekst z badanego pliku porównywany jest do tekstów zgromadzonych w repozytoriach:

1. Baza aktów prawnych i orzeczeń sądowych,
2. internet,
3. ORPPD tj. Ogólnopolskie Repozytorium Pisemnych Prac Dyplomowych,
4. baza publikacji naukowych dostępnych w otwartym dostępie (ang. *Open Access*),
5. Baza Postępowań Awansowych,
6. prywatna baza instytucji (opcjonalna).

Niezależnie od języka w jakim napisana jest praca, badanie jest wykonywane zawsze względem pełnego i dostępnego zestawu baz referencyjnych. Jeżeli w jednej z nich znajdzie się tekst podobny, spełniający zadane kryteria (np. poziom podobieństwa, długość frazy), to ten tekst zostanie zaznaczony w raporcie jako wymagający głębszej analizy promotora. Jeżeli podczas weryfikacji opiekun pracy uzna, że wskazany przez system fragment nie jest istotnie podobny to może go wykluczyć z wyniku i przeliczyć ponownie. Wtedy w raporcie widoczny będzie wynik pierwotny wskazany przez system oraz wynik po przeliczeniu.

Każdy tekst badany jest w systemie pod kątem:

1. Podobieństwa – system szuka różnych typów zgodności tekstu. Mogą to być:
   a. klony prac dyplomowych,
   b. zapożyczenia fragmentów z dokumentów w bazach referencyjnych,
   c. fragmenty podobne semantycznie do tekstów z baz referencyjnych (niezależnie od szyku słów);
2. Manipulacji – system szuka nadużyć typu:
   a. dzielenie wyrazów mikrospacjami,
   b. różnego typu łączenie wyrazów,
   c. zamianę czcionek w wyrazach,
   d. ukryte znaki specjalne,
3. Zmiany stylu pisania pracy.

1.4.4. Integracja z JSA poprzez interfejsy maszynowe (API)

Jednym z kluczowych wymagań przekazywanych przez uczelnie podczas projektowania systemu była konieczność posiadania przez system interfejsu umożliwiającego podłączenie się innych systemów uczelnianych, które obsługują wiele procesów odbywających się wewnątrz jednostki. Dla takich podmiotów JSA udostępnia swoje usługi za pośrednictwem dedykowanego API, które w znakomitej większości pokrywa większość funkcjonalności dostępnych w interfejsie graficznym systemu. Wykorzystując tą opcję dostępu do JSA podmiot oszczędza czas na wdrożeniu nowego oprogramowania oraz na szkoleniu pracowników. Użytkownicy systemu uczelnianego dostają nowe możliwości w znanym wcześniej interfejsie systemu uczelnianego.

Za pomocą API JSA można:

1. utworzyć nowe badanie,
2. utworzyć nową próbę badania już istniejącego,
3. edytować metrykę badania,
4. pobrać raport w wersji ogólnej oraz szczegółowej,
5. akceptować raport z badania.

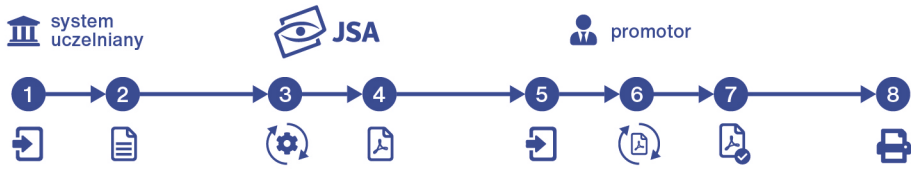
Jedną z niewielu różnic w korzystaniu z JSA za pomocą interfejsu i poprzez API jest dostęp do tekstu źródła podobieństwa. Wynika to z ograniczeń prawnych jakie nakłada

Ustawodawca na dostęp do danych zawartych w bazie Ogólnopolskiego Repozytorium Pisemnych Prac Dyplomowych. Użytkownicy, którzy używają JSA za pomocą API poprzez system uczelniany nie są uwierzytelniani według modelu uprawnień JSA, a tym samym nie ma kontroli nad tym komu udostępniane byłyby treści zawarte w ORPPD. Dlatego zaleca się, mimo korzystania z API, założenie kont użytkownikom, którzy potencjalnie mogą potrzebować dostępu do wglądu w teksty wykrytych źródeł podobieństw. Ponadto, w interfejsie JSA opiekun pracy po zapoznaniu się z raportem może wykluczyć z wyniku poszczególne fragmenty wskazane jako podobne. To jest funkcjonalność, która bezpośrednio może wpłynąć na ostateczny wynik w raporcie ale jest dostępna tylko dla promotora pracy w interfejsie użytkownika systemu.

Po przesłaniu przez API pracy do badania można znaleźć ją w interfejsie użytkownika na liście badań w zakładce Badania. Ze względu na to, że JSA nie ma dostępu do danych użytkowników systemów zewnętrznych, dane w metryce uzupełniane są standardowym oznaczeniem „użytkownikAPI". Obsługa badań zgłoszonych przez API jest identyczna jak w przypadku badań zarejestrowanych bezpośrednio w systemie. Dostępne są wszystkie przyciski funkcyjne (zgodnie z uprawnieniami użytkownika) tj. „Historia zmian" – pokazująca akcje przeprowadzane przez użytkowników w ramach jednego badania, „Udostępnij" – pozwalające na wysłanie badania do wglądu innemu użytkownikowi, „Edytuj" – pozwalające na edycję metryki badania, „Nowa próba" – zlecenie wykonania ponownego sprawdzenia pracy, „Pobierz" – umożliwiający wygenerowanie raportu ogólnego lub szczegółowego, oraz „Raport" – przenoszący do podglądu wyników z badania pracy.

Przykładowy schemat wykorzystania API JSA poprzez system uczelniany:

1. Student loguje się do systemu uczelnianego.
2. Zakłada badanie tzn. wypełnia metrykę pracy i dodaje plik (bądź pliki pracy). Plik pracy zostaje wysyłany do systemu JSA za pomocą systemu uczelnianego.
3. System sprawdza pracę pod kątem zapożyczeń, manipulacji na tekście i obecności w pracy obcego stylu.
4. System JSA generuje raport w systemie i przesyła go w postaci pdf (raport ogólny i szczegółowy) do systemu uczelnianego.
5. Promotor loguje się do systemu uczelnianego i ma dostęp do raportu pdf.
6. Promotor analizuje wynik raportu (Analiza statystyczna i wykres).
7. Promotor akceptuje raport w systemie uczelnianym. Raport pdf może być udostępniany innym osobom poprzez system uczelniany np. recenzentowi lub studentowi.
8. Promotor pobiera i drukuje raport pdf, na którym uzasadnia dopuszczenie lub niedopuszczenie studenta do obrony.



Ilustracja 1.10. Przykładowy proces wykorzystania API JSA

1.4.5. Uprawnienia

Moduł uprawnień w JSA ściśle określa dostęp do obiektów w systemie. Konta użytkowników nadawane są przez administratora każdej jednostki naukowej na podstawie jego uprawnień w systemie POL-on. Konto użytkownika wymagane jest do korzystania z interfejsu JSA. W przypadku podmiotów korzystających wyłącznie z API systemu, takie konta nie są wymagane ale są pomocne przy interpretacji raportu z badania. Dostęp do JSA zalecany jest dla wszystkich promotorów i recenzentów ale także dla osób wspierających promotorów np. pracowników dziekanatu, którzy stanowią zaplecze techniczne opiekunów prac dyplomowych. Jedna z ról, o nazwie Podgląd, dedykowana jest dla osób pełniących funkcję rzecznika dyscyplinarnego w uczelni. W zależności od poziomu uprawnień użytkownik dostaje dostęp do poszczególnych funkcji systemu.

Jak pokazują statystyki, po ponad dwóch latach działania systemu, najpopularniejszą rolą nadawaną użytkownikom jest Promotor, dzięki niej m.in. promotorzy mogą dodawać nowe prace do badania, analizować raport z badania, wykluczać poszczególne wskazania systemu i przeliczać wynik, udostępniać badanie i raport innym promotorom lub recenzentom oraz drukować raport. Drugą największą grupę użytkowników stanowi rola Biuro, która umożliwia zgłoszenie tekstu do badania, przypisanie opiekuna pracy i recenzenta oraz wydrukowanie ogólnego raportu z badania. W systemie wyróżnia się także role nadzorcze oraz administratorskie.

Poniżej przedstawiona została statystyka ról i uprawnień na dzień 15.11.2020 roku:

Użytkownik	Ogółem	Biuro	Promotor	Podgląd	Ustawienia	Statystyka	Integracja	Administracja
JSA	46506	2252	44361	1453	493	1303	334	673

Ilustracja 1.11. Statystyki uprawnień w JSA na dzień 15.11.2020

1.4.6. Wynik badania – raport JSA

Raport prezentuje wynik z badania antyplagiatowego, który dostępny jest poprzez stronę www na liście badań dla uprawnionych użytkowników. Dodatkowo, jeśli plik pracy został przesłany do systemu przy pomocy API to raport przesyłany jest w odpowiedzi

na żądanie systemu zewnętrznego. Raport w każdej chwili może zostać wydrukowany w odpowiedniej formie – skróconej lub rozszerzonej.

Wynik przedstawiony jest w wartościach procentowych wskaźnika procentowego rozmiaru podobieństwa (PRP), który obrazuje poziom podobieństwa tekstu badanej pracy do innych dokumentów źródłowych znajdujących się w bazach porównawczych. Jest to wartość, stanowiąca pewien punkt odniesienia, natomiast nie definiuje wyniku procesu obrony studenta. Analiza raportu przez promotora pracy powinna uwzględniać dane zawarte we wszystkich siedmiu sekcjach dokumentu.

Oto najważniejsze informacje jakie znajdują się w raporcie:

1. Sekcja Akcje Raportu – grupuje najważniejsze akcje dostępne w raporcie m.in. wyświetlenie tekstu pracy, akceptację raportu, udostępnienie lub pobranie wersji pdf.
2. Sekcja Parametry Raportu – dostarcza ogólnych informacji jak numer próby, wartość ustawień z jakimi zbadany był tekst.
3. Sekcja Metryka badania – są to informacje zaczerpnięte z metryki badania.
4. Sekcja Tekst pracy – wyświetlany jest w niej badany tekst wraz z pokolorowanymi fragmentami, które system wskazał jako zbieżne z innym dokumentem znajdującym się w bazie referencyjnej. W tej części raportu można porównać tekst badany do tekstu źródłowego. Można także wykluczyć z wyniku dany fragment.
5. Sekcja Analiza tekstu – dostarcza informacji statystycznych z zakresu analizy tekstu tj., ile wykryto znaków spoza języka pracy, ile znaleziono wyrazów nierozpoznanych, ile usunięto grafik podczas wyodrębniania tekstu z pliku, liczba fragmentów innego stylu oraz rozkład długości wyrazów.
6. Sekcja Wyniki ogólne – jest to podsumowanie wyniku PRP w zależności od długości wyróżnionych fragmentów. Każda wartość PRP oznaczona jest dodatkowo informacją czy mieści się w zadanych progach tolerancji przyjętych na uczelni.
7. Sekcja Referencyjne bazy porównawcze – wyświetla wyniki pogrupowane dla każdej bazy referencyjnej oddzielnie.
8. Sekcja Wnioski – sekcja, w której promotor pracy może zapisać komentarz odnoszący się do raportu oraz swoją decyzję.

Raport w wersji pdf jest odwzorowaniem raportu widocznego w interfejsie systemu. Jedyną różnicą jest to, że na wydruku nie jest prezentowane źródło podobieństwa tekstu. Raport można wydrukować lub pobrać w wersji ogólnej lub szczegółowej. Ta pierwsza dołączana jest najczęściej do dokumentów studenta, ponieważ zawiera tylko podstawowe informacje z raportu. Szczegółowa wersja raportu zawiera cały wyodrębniony tekst ze wszystkich plików badania. Raport zawiera zaznaczenia fragmentów tekstu, które zostały wskazane przez system jako podobne. Ze względu na długość tego dokumentu najczęściej jest on pobierany w formie elektronicznej.

Podczas analizy raportu promotor nie musi zgadzać się ze wskazaniami systemu i może zrezygnować z ich wliczania do wyniku. Funkcjonalność ta nazywa się wykluczaniem i tylko promotor pracy ma do niej uprawnienia. Po odznaczeniu fragmentu pojawi się okno, w którym należy podać powód wykluczenia np. informację o zapożyczeniu zasadnym. Zapisanie komentarza do wykluczenia skutkuje wykreśleniem podobieństwa

oraz usunięciem kolorowego podkreślania w tekście pracy. Wykluczenia można wskazywać pojedynczo lub grupowo według wskazanego źródła. Wynik zostanie przeliczony ale oryginalna wartość także zostanie zapisana. Może się tak zdarzyć, że system oznaczy także takie fragmenty, które są zacytowane prawidłowo przez autora pracy. Wtedy takie teksty oznaczone są jako „możliwe prawidłowe cytowania" i można je wszystkie na raz wykluczyć z wyniku. Przeliczenie wyniku najczęściej powoduje obniżenie wyniku PRP w raporcie dlatego ta funkcja jest dostępna tylko dla opiekuna pracy. Recenzent pracy ma dostęp do szczegółowego raportu jednak nie ma możliwości wpłynąć na wynik badania.

Mimo, że JSA jest systemem obowiązującym z mocy prawa to na rynku dostępna jest cała gama komercyjnych systemów antyplagiatowych. Niezalecane jest porównywanie raportów między systemami, gdyż każdy system antyplagiatowy generuje raporty w unikatowej dla siebie formie. Wyniki raportów zależne są przede wszystkim od algorytmu wyszukiwania podobieństw, który nigdy nie jest taki sam dla dwóch systemów. Ponadto wynik badania jest bezpośrednio związany ze źródłami referencyjnymi, które są przeszukiwane przez algorytm detekcji. Jeżeli dwa porównywane systemy pracują na innych bazach źródeł to raporty nigdy nie będą takie same, a nawet mogą być rozbieżne. Jeżeli źródło podobieństwa znajduje się w bazie A podłączonej do jednego systemu, a jednocześnie nie znajduje się w bazie B wykorzystanej w innym systemie, to nie ma możliwości aby wyniki końcowe były takie same. Ponadto należy pamiętać o najważniejszej zasadzie jaka przyświecała twórcom JSA – aby analiza Promotora pracy skupiała się na istotnych fragmentach tekstu, czyli na tekstach długich, powyżej trzech zdań, a nie pojedynczych słowach jak to przedstawiane było w innych systemach.

1.4.7. Administracja i ustawienia systemu

Administrator jest pierwszym użytkownikiem w kontekście uczelni. Dostaje dostęp do JSA na podstawie uprawnień administratorskich w systemie POL-on. Jego zakres uprawnień pozwala konfigurować system w ramach całej uczelni lub poszczególnych jednostek. Tylko administrator może tworzyć konta dla innych użytkowników systemu i zarządzać ich uprawnieniami w kontekście uczelni oraz w podległych jej jednostkach.

Administrator konfiguruje również połączenie systemu uczelnianego z JSA za pośrednictwem API jeśli uczelnia wymaga takiej integracji pomiędzy systemami. Dodatkowo, administrator może wskazać lokalne zasoby danych uczelni w postaci dedykowanych usług o zadanym kontrakcie, które mają być użyte w trakcie badania prac dyplomowych lub doktorskich. Oprócz tych ogólnych ustawień, administrator może ustawiać także:

1. Wartości progów tolerancji PRP, dzięki którym użytkownicy JSA ostrzegani są o wysokim lub podwyższonym wyniku badania.
2. Długość fraz w jakich będzie przedstawiony wynik PRP w raporcie badania.
3. Wartość parametru poziomu podobieństwa z jakim będzie badany każdy tekst w kontekście uczelni. Funkcjonalność ta wpływa na wrażliwość algorytmu detekcji na

zbieżność tekstów. Służy do dokładnego określenia przez jednostkę, czy i w jakim stopniu system ma wskazywać fragmenty będące sparafrazowanym odpowiednikiem znalezionego źródła. Dlatego też skala, na jakiej można zmieniać wartość, jest czterostopniowa, a poziom można ustawić na: niski (wartość rekomendowana), średni, wysoki oraz bardzo wysoki.

4. Wprowadzenie tekstów, które nie będą brane pod uwagę podczas każdego badania tekstu w kontekście uczelni lub jednostki. Jest to szczególnie przydatne dla stałych tekstów powtarzających się w większości prac dyplomowych np. strona tytułowa, podziękowania lub oświadczenia.

Domyślnie ustawione wartości parametrów można zmienić wedle preferencji per instytucja lub per jednostka, co oznacza, że każda jednostka może mieć indywidualnie dostosowaną konfigurację ustawień. Dzięki temu, badania realizowane w ramach jednej jednostki uczelni mogą mieć inne ustawienia czułości na podobieństwo tekstu. Jest to bardzo istotne w przypadku niektórych specyficznych kierunków studiów, na których prace dyplomowe oparte są na powtarzających się frazach tekstu. Są to zamieszczone w pracy badania w postaci wypełnionych tych samych formularzy lub tabel. Przykładem takich kierunków są inżynieryjne lub ekonomiczne. Przy niskim parametrze podobieństwa jest bardzo prawdopodobne, że system wskaże takie fragmenty jako nadużycie i wynik PRP będzie wysoki. Dlatego w JSA przewidziano możliwość wyodrębnienia ustawień dla konkretnych jednostek uczelni.

1.5. Szkolenia

Początkowe założenia projektu stanowiły, że z każdej uczelni przeszkolonych zostanie kilka osób, a następnie rozpropagują one wiedzę o systemie wśród grona zainteresowanych pracowników uczelni. Dlatego Ośrodek Przetwarzania Informacji – Państwowy Instytut Badawczy wraz z Ministerstwem Nauki i Szkolnictwa Wyższego zorganizował cykl konferencji, na których zaprezentowano system jeszcze zanim nastąpiło ostateczne wdrożenie. Przeszkolono blisko 1000 osób, a spotkania odbyły się w największych ośrodkach akademickich w kraju, tj. Warszawa, Wrocław, Kraków, Gdańsk i Lublin. Zainteresowanie tymi spotkaniami przerosło oczekiwania organizatorów i udowodniło, że przyjęta formuła rozpowszechniania się wiedzy na uczelniach nie sprawdziła się. Większość osób zainteresowanych nowym systemem pragnęła otrzymać informacje od twórców JSA a nie od pośredników, którzy sami mogli posiadać niewystarczający zakres wiedzy o systemie.

Na podstawie opinii uczestników pierwszej tury konferencji postanowiono o rozpoczęciu cyklu szkoleń dedykowanych uczelniom. Szkolenia podzielone były tematycznie tak, aby lepiej dopasować przekazywane informacje do słuchaczy. Każda jednostka naukowa mogła zwrócić się do OPI PIB z prośbą o zorganizowanie bezpłatnego szkolenia swoich pracowników w dogodnym terminie. W większych miastach szkolenia na różnych uczelniach były grupowane w pojedyncze wydarzenia. W trakcie szkolenia poruszane były nie tylko tematy teoretyczne dotyczące pracy z systemem ale także praktyczne ćwiczenia pomagające w interpretowaniu raportu z badania antyplagiatowego.

Dedykowane szkolenia rozpoczęły się wraz z udostępnieniem systemu na środowisku produkcyjnym. W roku 2019 odbyło się 101 szkoleń, podczas których przeszkolono 9.970 osób.

W sumie w całym kraju, przeszkolonych i wdrożonych zostało 11.000 pracowników uczelni. Zespół szkoleń Ośrodka Przetwarzania Informacji – Państwowego Instytutu Badawczego w sposób ciągły prowadzi bezpłatne szkolenia z JSA. Szkolenia są okazją do wymiany opinii między promotorami, pracownikami administracyjnymi uczelni oraz pracownikami OPI PIB. Dla twórców systemu jest to konstruktywna informacja zwrotna z rynku. Wiele z wniosków zgłaszanych przez promotorów zostało już wdrożonych w JSA w postaci dodatkowych funkcjonalności. Po szkoleniach spłynęło do OPI PIB bardzo wiele pochlebnych opinii na temat JSA oraz na temat szkoleń i materiałów szkoleniowych. Użytkownicy często wskazywali i chwalili proaktywną postawę OPI PIB przy organizacji wielokanałowej pomocy.

W ramach działań edukacyjnych użytkownicy mogą korzystać z przygotowanych dodatkowych materiałów szkoleniowych, m.in. z serii sześciu filmów instruktażowych dostępnych na kanale OPI PIB na platformie YouTube oraz samouczków. Oglądalność wszystkich części od grudnia 2018 roku wyniosła ponad 40 tys.

Dla chętnych udostępnione zostało szkolenie e-learningowe na platformie Navoica.pl. W szkoleniu uczestniczyło około 200 osób. Ponad 80 osób zakończyło szkolenie certyfikatem. Mailing z kompletem materiałów dla wszystkich użytkowników JSA otrzymało prawie 40 tys. osób.

Od początku funkcjonowania JSA działa wsparcie użytkownika, gdzie operatorzy helpdesku odpowiadają na zgłoszenia użytkowników i rozwiązują zgłaszane problemy. Średni czas odpowiedzi z helpdesku to 30 minut, a maksymalny czas reakcji to 1 godzina. Na stronie bazy wiedzy JSA użytkownicy mają do wglądu niemal 150 publikacji, które zawierają treści ułatwiające korzystanie z systemu. Użytkownicy znajdą tam ponad 90 artykułów, które zostały przygotowane w oparciu o zagadnienia poruszane w trakcie szkoleń oraz pytania zadawane poprzez system zgłoszeń helpdesk. Powstała również baza najczęściej zadawanych pytań licząca około 100 odpowiedzi w postaci wyjaśnień kluczowych zagadnień.

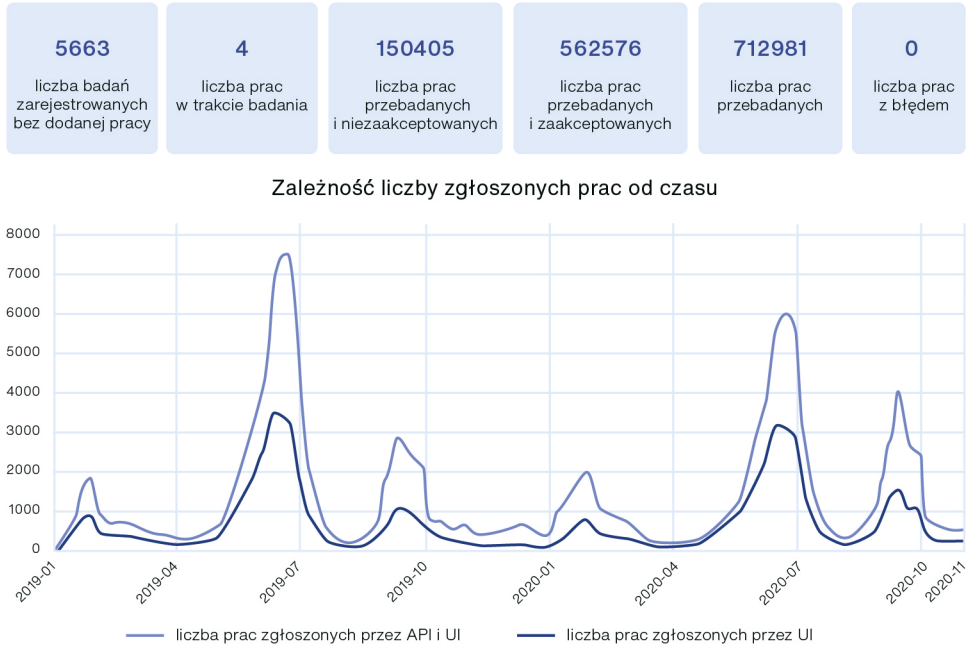
Statystyki dotyczące bazy wiedzy JSA na dzień 10.11.2020 roku:

- liczba użytkowników wyświetlających stronę: 174.000
- liczba odsłon strony: 263.000
- liczba użytkowników wyświetlających stronę tygodniowo: 6100
- liczba użytkowników wyświetlających stronę od uruchomienia witryny: 173.000
- liczba odsłon strony od uruchomienia witryny: 262.000
- liczba wyświetleń stron w dni powszednie: od 960 do 1615
- średni czas wyświetlania strony: 3 min 29 sek.
- wyświetlanie strony na komputerach i laptopach: 76 procent użytkowników
- wyświetlanie strony na urządzeniach mobilnych: 24 procent użytkowników.

1.6. Statystyki JSA

W ramach prac nad systemem JSA powstał analityczny intranetowy dashboard napisany z użyciem języka R i pakietu Shiny. Rozwiązanie to działa na serwerze R (Shiny Server), który korzysta z danych zbieranych w JSA ale aby nie obciążać bazy produkcyjnej korzysta z jej repliki w Zapasowym Centrum Danych. Jest to zalecone rozwiązanie w przypadku gdy silniki analityczne wykonują wiele zapytań, które mogą spowolnić bazę produkcyjną obciążaną zapisami i aktualizacjami.

Dodatkowo analiza gromadzonych danych w JSA jest ważnym elementem w administrowaniu systemem. Z jednej strony pozwala na monitorowanie zajętości naszych zasobów, poprzez np. śledzenie liczby zgłoszonych prac do przebadania, z drugiej strony pozwala na analizę statystyczną zgromadzonych danych. Dlatego aplikacja prezentuje istotne dane na temat m.in. liczby przebadanych prac dyplomowych, liczby zakładanych kont oraz przyznawanych ról, rozkładzie wartości PRP i detekcji prac o podwyższonej wartości PRP.



Ilustracja 1.12. JSA – aktywność użytkowników systemu w czasie, dane na dzień 10.11.2020

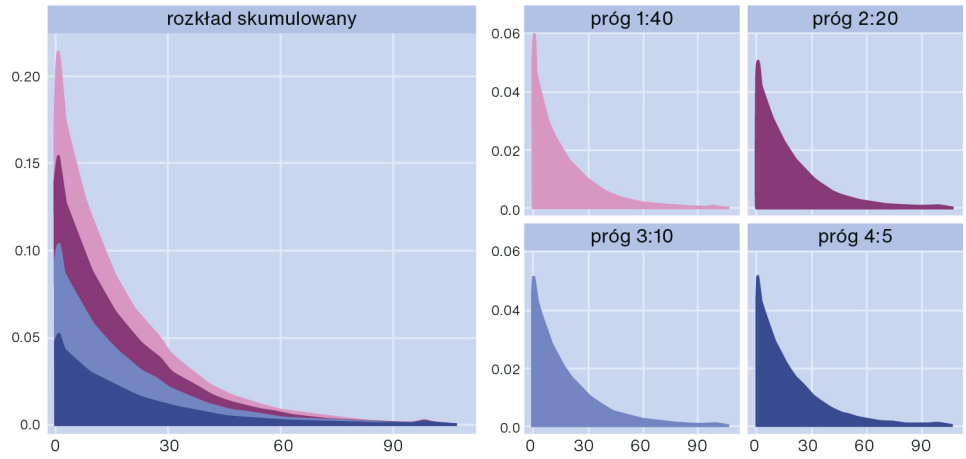
W pierwszej zakładce, zgodnie z jej nazwą, przedstawiamy podsumowanie szeregu różnych statystyk dla zgłoszonych badań i prób. Przede wszystkim prezentujemy podsumowanie liczby zgłoszonych badań względem ich statusu, gdzie w JSA wyróżniamy następujące statusy:

- nowe badanie bez zgłoszonej pracy
- nowe badanie przesłane do analizy
- badanie zostało wykonane, ale jeszcze nie zostało zaakceptowane
- badanie zostało wykonane i zaakceptowane
- wystąpił błąd w czasie wykonywania badania.

Dodatkowo prezentujemy całkowitą liczbę zgłoszonych prac w ciągu każdego dnia z wyróżnioną liczbą prac, które zostały zgłoszone przy użyciu aplikacji JSA lub poprzez API REST (patrz Ilustracja 1.12). Dla wybranego przedziału czasowego wyznaczamy statystyki informujące o: czasie badania pracy, średniej liczbie prac przypadających na badanie, średniej liczbie dokumentów przypadających na pracę, średniej liczbie badań przypadających na promotora oraz pokazujemy rozkład liczby badań względem typu – praca licencjacka, inżynierska, magisterska lub doktorska.

W JSA do wskazania poziomu podobieństwa pracy badanej do innych dokumentów znajdujących się w bazach porównawczych (ORPPD, internet, akty prawne, uczelniane bazy referencyjne) jest PRP – Procentowy Rozmiar Podobieństwa i temu wskaźnikowi poświęciliśmy kolejną zakładkę. Każda jednostka oraz instytucja ma przypisany zbiór wskaźników, które definiują sposób wyznaczania wartości PRP, tj. liczbę wyrazów w frazie, od której dany fragment uznajemy jako zapożyczony. JSA umożliwia ustawienie czterech takich progów zastrzegając wybranie jednego wiodącego, dla którego będzie wyznaczona główna wartość PRP. W związku z tym, zgłaszając pracę do JSA, wskazujemy, w ramach której jednostki dana praca ma być przebadana, czyli wskazujemy ustawienia do sposobu wyznaczania wartości PRP.

W aplikacji Shiny przedstawiamy rozkład oraz wartości decyli wiodącej wartość PRP dla przebadanych prac we wskazanym przedziale czasowym. Wykresy te pozwalają na porównanie rozkładu PRP w różnych przedziałach czasowych. Jest to np. konieczne w momencie aktualizacji algorytmu wskazującego pasaże podejrzane o plagiat, czy planowana zmiana nie wpływa znacząco na wyznaczane wartości PRP, tzn. zawyża lub zaniża je w porównaniu do wcześniejszych przeliczeń. Innym zastosowaniem tego rodzaju wykresów jest monitorowanie, czy wartości PRP nie maleją wraz z czasem. Taka sytuacja mogłaby sygnalizować o pewnych lukach systemu, np. autorzy prac „nauczyli się" działania systemu i potrafią „oszukiwać" algorytm detekcji, w związku z tym wartości wyznaczanych PRP będą coraz niższe. Dodatkowo prezentujemy rozkład wartości PRP dla zaakceptowanych badań z uwzględnieniem wybranych wartości progów wyników od ilu wyrazów fraza jest uwzględniana przy wyznaczaniu PRP (patrz Ilustracja 1.13). Domyślnie są to wartości 40, 20, 10, 5, gdzie wartością wiodącą jest 10.



Ilustracja 1.13. JSA – rozkład wartości PRP w podziale na progi i skumulowany, dane na dzień 10.11.2020

Użytkownik	Ogółem	Biuro	Promotor	Podgląd	Ustawienia	Statystyka	Integracja	Administracja	Administracja POL-on
JSA	46466	2248	44325	1441	492	1303	334	674	0
POL-ON	715	255	93	191	176	185	163	336	697

Ilustracja 1.14. JSA – informacje o użytkownikach posiadających konto w JSA i ich historii zakładania na dzień 10.11.2020

Kolejnymi informacjami ujętymi w aplikacji Shiny są dane na temat użytkowników. Większość użytkowników JSA to osoby posiadające lokalne konto w JSA, jednakże istnieje również możliwość korzystania z JSA przy użyciu konta MCL, gdy logująca się osoba posiada uprawnienia INST_ADM lub INST_NAUK_ADM w POL-onie. W związku z powyższym, na Ilustracji 1.14 dokonaliśmy podziału użytkowników na użytkowników JSA i tak zwanych administratorów POL-onu. W każdej grupie przedstawiliśmy rozkład ról nadawanych w JSA. Jak widzimy większość użytkowników to promotorzy. Warto

nadmienić, że realna liczba promotorów korzystających z JSA jest znacznie większa – wiele uczelni korzysta z JSA poprzez integrację API, w takich przypadkach wielu promotorów nie posiada lokalnego konta w JSA, a korzysta jedynie z gotowych raportów otrzymywanych poprzez API.

Na Ilustracji 1.14 została zaprezentowana również informacja o liczbie założonych kont w ujęciu czasowym. Informacje te były istotne w początkowej fazie działania JSA, kiedy uczelnie masowo zakładały nowe konta swoim pracownikom, obecnie dzienny przyrost nowych użytkowników JSA jest niewielki.

ROZDZIAŁ 2

METODY PRZETWARZANIA DANYCH

Daniel Karaś
Jakub Uszyński
Łukasz Podlodowski
Marek Nurzyński
Maciej Kowalski
Martyna Śpiewak

Poniższy rozdział skupia się na analizie metod przetwarzania danych wykorzystanych przy budowie JSA. W tym rozdziale zostały opisane istotne elementy silnika przetwarzania danych.

Przetwarzanie danych to przekształcanie treści i postaci danych wejściowych metodą wykonywania systematycznych operacji w celu uzyskania wyników w określonej postaci. Wyróżnia się przetwarzanie numeryczne (stosunkowo skomplikowane algorytmy i niewielkie ilości danych) oraz systemowe (stosunkowo proste algorytmy i duże ilości danych) [3]. W przypadku JSA mamy do czynienia z oboma rodzajami przetwarzania danych.

Jednolity System Antyplagiatowy nie jest typowym systemem opartym na paradygmacie CRUD (ang. *Create*, *Read*, *Update*, *Delete*) wzbogaconym o przepływy, ma duży komponent obliczeniowy, którego celem jest potokowe badanie pracy dyplomowej pod kątem wystąpienia nadużyć plagiatu. W tym rozdziale zagłębiamy się w dziedziny uczenia maszynowego, przetwarzania języka naturalnego, czy algorytmiki i specjalizowanych struktur danych.

2.1. Indeksowanie danych – LSH Forest

Obok klasycznego przeszukiwania pełnotekstowego z użyciem Elasticsearch, w JSA funkcjonuje alternatywa pod postacią indeksu LSH, który swoje działanie luźno opiera na ideach zawartych w publikacji autorstwa Bawa i in. [1]. Autorzy opisują w wyczerpujący sposób zarówno zasadę działania algorytmu, któremu nadali miano LSH Forest oraz jego zalet względem podobnych rozwiązań. Dlatego też, czytelnik, który zainteresowany jest bardziej formalną stroną opisywanego rozwiązania powinien znaleźć wszelkie interesujące go niuanse w wymienionym źródle. Podrozdział ten natomiast podejmuje próbę określenia roli oraz charakterystyki indeksu LSH w bardziej nieformalny sposób.

2.1.1. LSH Forest – definicje

Aby móc naszkicować pewien intuicyjny obraz całości, należy rozpocząć od podstawowego budulca całego zagadnienia, czyli samego LSH. LSH, czyli *locality-sensitive hashing* rozwiązuje problem przeszukiwania pewnej dużej przestrzeni poszukiwań ze zdefiniowaną metryką charakterystyczną dla danego problemu – w szczególności dla obiektów będących duplikatami bądź różniącymi się w bardzo małym stopniu. Dokonuje tego poprzez użycie pewnej rodziny funkcji hashujących $\mathcal{H}$, gdzie każda funkcja w $\mathcal{H}$ spełnia bardzo konkretny warunek – jeśli wyznaczymy wartości funkcji h dla zadanej pary obiektów oddalonych maksymalnie od siebie o pewną zadaną odległość r, to z dużym prawdopodobieństwem, wartości hashów dla obu obiektów będą takie same. Konsekwencja tego warunku jest prosta – oznacza to, że można pogrupować dostatecznie zbliżone do siebie obiekty. Oznacza to, że chcąc znaleźć w przestrzeni zaindeksowanych obiektów, obiekt podobny do rozpatrywanego w danej chwili – wystarczy dla niego policzyć wartość funkcji hashującej, a następnie odnaleźć grupę obiektów z tą samą wartością funkcji hashującej. W praktyce do identyfikowania grup obiektów stosuje się nie tyle wartość pojedynczej funkcji hashującej, co raczej pewną wypadkową „sygnaturę" powstałą na skutek złączenia ze sobą n wartości różnych funkcji hashujących. Jak wykazują autorzy [1], występuje pewna ciekawa, acz dość intuicyjna zależność – im dłuższa sygnatura (czyli im więcej wartości różnych funkcji hashujących wykorzystuje), tym mniejsza szansa na błędne zgrupowanie ze sobą obiektów. Należy jednocześnie uważać, gdyż przesadne wydłużenie sygnatury może mieć odwrotny efekt od zamierzonego i nawet blisko spokrewnione obiekty mogą trafić do zupełnie innych grup. Dlatego też w praktyce często spotyka się rozwiązanie polegające na tym, że operację wydłużania sygnatury zastępuje się poprzez tworzenie kilku istniejących obok siebie indeksów LSH, gdzie każdy posiada odrębny zbiór krótszych sygnatur. W szczególności, indeksy te mogą różnić się od siebie długościami sygnatur, co w pewnym sensie może symulować „szersze" lub „węższe" przeczesywanie przestrzeni w poszukiwaniu najbliższych sąsiadów.

LSH Forest natomiast idzie z tą ideą o krok dalej i traktuje każdy z tych indeksów nie tyle jak prostą strukturę słownikową typu sygnatura→wartość/zbiór wartości, a raczej drzewo prefiksowe, w którym poszukiwanie najbliższych sąsiadów polega na trawersowaniu najpierw do liścia, który najbliżej odpowiada poszukiwanej sygnaturze, a na-

stępnie pięciu się w górę w miarę konieczności uzyskania kolejnych, nieco dalszych sąsiadów. „Pięcie się w górę" w tym wypadku oznacza „wymazywanie" części sygnatury poczynając od jej końca tak, by niejako złagodzić ograniczenie narzucone na wyszukiwanie. Rozwiązanie to ma pewną fundamentalną wadę, o której autorzy [1] nie mówią wprost – implicite zakłada, że składowe sygnatury są ułożone „od lewej do prawej" zgodnie z jakąś bliżej nieokreśloną wagą, co nie jest prawdą, gdyż funkcje z rodziny $\mathcal{H}$ wybierane są losowo i nie da im się przypisać żadnej wagi. Rozważmy to na przykładzie sygnatury złożonej z 3 funkcji hashujących, z czego każda zwraca binarny wynik – 0 lub 1. W indeksie posiadamy obiekty o sygnaturach 000 i 111. Staramy się znaleźć najbliższych sąsiadów dla nowego obiektu, z sygnaturą 011. Jak widać, pierwszy bit dzieli całą przestrzeń poszukiwań na dwie części, z czego pierwsza zostanie rozpatrzona część z sygnaturami 0xx, mimo że gałąź 1xx posiada znacznie lepszego kandydata (000 przecina poszukiwaną sygnaturę jedynie na jednej pozycji – skrajnie lewej, natomiast 111 już na dwóch, co przekłada się na większe prawdopodobieństwo, że obiekty są do siebie podobne). Ten nieco „zabawkowy" przykład może nie wydawać się dużym problemem – w końcu ostatecznie procedura przeszukiwania może wrócić do korzenia, a potem rozpatrzeć pod-drzewo 1xx, ale należy pamiętać, że w realnych zastosowaniach rozmiar danych jest dużo większy i czasowy oraz pamięciowy koszt takiej trawersacji przestaje być zaniedbywalny.

Drzewa te z kolei spajane są w całokształt – las (stąd nazwa LSH Forest), gdzie do odpowiedzialności lasu należy konkatenacja wyników zwracanych z poszczególnych drzew i zatrzymanie procesu poszukiwania w momencie gdy osiągnie się zadowalającą liczbę wyników (lub zupełnie wyczerpano całą przestrzeń poszukiwań). Dodatkowo ma on tę ukrytą zaletę, że poprzez wykorzystanie kilku drzew, minimalizuje on problem omówiony w poprzednim paragrafie i w praktyce można otrzymać zadowalające wyniki bez konieczności przechodzenia przez całą przestrzeń poszukiwań.

2.1.2. LSH Forest w JSA

Komponent LSH Forest zaimplementowany w JSA nie realizuje literalnie algorytmu opisanego w pracy [1] – nadal wykorzystuje struktury drzewa i lasu, ale z uwagi na koszt pamięciowy i czasowy, upraszcza procedurę przechodzenia przez drzewo. Po wyznaczeniu sygnatury danego dokumentu, nie dochodzi do trawersowania góra/dół po węzłach drzewa odpowiadających malejącej „od prawej strony" sygnaturze – zamiast tego, wyznaczany jest prefiks z poszukiwanej sygnatury o zadanej długości, a następnie każde drzewo zwraca węzły o sygnaturze rozpoczynającej się od zadanego prefiksu, wraz z oceną podobieństwa kandydatów względem poszukiwanego obiektu. W kolejnym kroku, las zbiera wyniki z drzew, usuwa duplikaty, sortuje kandydatów względem malejącej wartości oceny i zwraca podzbiór M najlepiej dopasowanych kandydatów.

Należy w tym miejscu zastanowić się nad tym, czym w rozumieniu LSH Forest jest „obiekt" w kontekście JSA. Jeśli założymy, iż „obiekt" w JSA stanowi cały tekst pracy źródłowej to LSH Forest dostając na swoim wejściu tekst pracy podejrzanej o plagiat jest w stanie zwrócić jedynie takie prace źródłowe, które są praktycznie klonem bada-

nej pracy. Takie zjawiska są jednak niezmiernie rzadkie i znacznie częściej dochodzi do plagiatu polegającego na „sklonowanych", może nieznacznie przetworzonych fragmentach będących częścią większej pracy. To z kolei wymusza podział dokumentów źródłowych na mniejsze, kilku-kilkunasto zdaniowe części z użyciem przesuwnego okna (*sliding window*), które poddane są w pierwszej kolejności przetwarzaniu (normalizacja białych znaków, usuwanie znaków interpunkcyjnych, usuwanie znaków końca linii itp.) a następnie dla każdego fragmentu generowana jest oddzielna sygnatura na potrzeby indeksacji. Analogiczny proces odbywa się dla badanej pracy, ale generowane sygnatury służą do wyszukiwania w drzewach.

W związku z wyżej przyjętym modelem, pojawia się kilka ciekawych wyzwań. W pierwszej kolejności – wybór tokenizatora odpowiedzialnego za wyodrębnianie z tekstu zdań, na bazie których w następnej kolejności zostaną utworzone większe fragmenty gotowe do indeksacji. Wybór ten może wydawać się mało znaczący, jednak dane doświadczalne pokazują, że ma on bardzo duże znaczenie dla skuteczności działania algorytmu. Model posiada też kilka parametrów, którymi można sterować – jak chociażby liczba drzew, długość indeksowanych fragmentów tekstu, rozmiar okna przesuwnego, długość sygnatury czy liczba zwracanych kandydatów, które zostały dopasowane drogą eksperymentalną poprzez ewaluację na licznych zbiorach testowych.

2.1.3. LSH Forest – ewaluacja

Jako miary jakości proponowanego rozwiązania, przyjęto precyzję (*precision*) i czułość (*recall*) będące analogami $prec_{mac}$ (*macro precision*) oraz rec_{mac} (*macro recall*) [17] zdefiniowanymi jako:

$$prec_{mac}(S,R) = \frac{1}{|R|}\sum_{r\in R}\frac{|\bigcup_{s\in S}(s\cap r)|}{|r|},$$

$$rec_{mac}(S,R) = \frac{1}{|S|}\sum_{s\in S}\frac{|\bigcup_{r\in R}(s\cap r)|}{|s|},$$

gdzie S to zbiór przypadków plagiaryzmu, a R – zbiór detekcji plagiaryzmu, gdzie każdy przypadek zawiera w sobie zbiór identyfikatorów dokumentów. Część wspólna ($S \cap R$) przypadków oznacza prawdziwie pozytywne (*true positive*) wskazania algorytmu.

Algorytm w fazie testowej był ewaluowany względem zbioru utworzonego na bazie Ogólnopolskiego Repozytorium Pisemnych Prac Dyplomowych. Korpus prac źródłowych obejmował blisko 7000 dokumentów w języku polskim i blisko 2000 prac w języku angielskim oraz prawie 9000 przypadków plagiaryzmu utworzonych syntetycznie poprzez „mieszanie" ze sobą dokumentów źródłowych oraz operację zmiany szyku w obrębie zdania.

Pierwszy z zamieszczonych niżej przebiegów wykorzystywał do tokenizacji „naiwne" rozwiązanie oparte na prostych wyrażeniach regularnych, natomiast drugi – bibliotekę Stanford CoreNLP. Wyniki potwierdzają wcześniej opisaną zależność pomiędzy jakością tokenizatora a skutecznością algorytmu i prezentują się następująco:

Tabela 2.1. Wyniki ewaluacji algorytmu w zależności od obranej metodyki tokenizacji

Metoda	Precyzja	Czułość
Regexp	0,1340	0,3070
CoreNLP	0,4881	0,9540

Widać, iż bardziej rozwinięty mechanizm tokenizacji z CoreNLP przekłada się na wyższe wskaźniki zarówno precyzji, jak i kompletności.

Ponadto, rozwój implementacji LSH Forest odbywał się równolegle z konkursem JSA, co pozwoliło na porównywanie algorytmu z rozwiązaniami zaproponowanymi przez różnych uczestników konkursu.

Tabela 2.2. Wyniki uzyskane w zadaniu 2, fazie 4 konkursu JSA

Precyzja	Czułość	F1
0,8079	0,8568	0,8316

2.1.4. LSH Forest – strona techniczna

Nieoceniona w procesie implementacji okazała się biblioteka MapDB, która dostarcza bardzo stabilną, szybką i zrównoleglоną implementację drzewa prefiksowego z możliwością korzystania z niego jako struktury on-disk, co było o tyle kluczowe, że rozmiar indeksowanych danych nie pozwalał na umieszczenie całej zawartości indeksu w pamięci operacyjnej. Indeks LSH Forest nie jest jednak definiowany jedynie przez zapisaną zawartość wchodzących w jego skład drzew. Koniecznym było utworzenie dodatkowego deskryptora, który informuje o konkretnej konfiguracji użytej do indeksowania tekstu, co pozwala wymusić symetrię konfiguracji pomiędzy procesem indeksacji i odpytywania indeksu – w przeciwnym razie, mogłoby dojść do sytuacji, gdy proces indeksujący korzysta przykładowo z fragmentów 10-zdaniowych, natomiast klient odpytujący indeks dzieli tekst na fragmenty 50-zdaniowe – wtedy poszukiwanie nie dałoby pożądanych rezultatów, nawet dla klonów.

Obecnie indeks LSH Forest w JSA wykorzystywany jest jedynie do indeksacji treści prac z Ogólnopolskiego Repozytorium Pisemnych Prac Dyplomowych, natomiast baza kodowa stworzona jest tak, by bez większego problemu móc zaindeksować dowolne inne źródło. Nie wymaga to żadnych zmian w strukturze samego LSH Forest, a jedynie dostarczenie dwóch klas – jednej odpowiedzialnej za dostarczanie treści i drugiej, która przekształci rzeczone treści do postaci metadanych przechowywanych docelowo w samym indeksie. Ponadto implementacja wspiera zarówno wariant dyskowy (dla dużych zbiorów danych) jak i taki, który może być załadowany w pełni do pamięci operacyjnej.

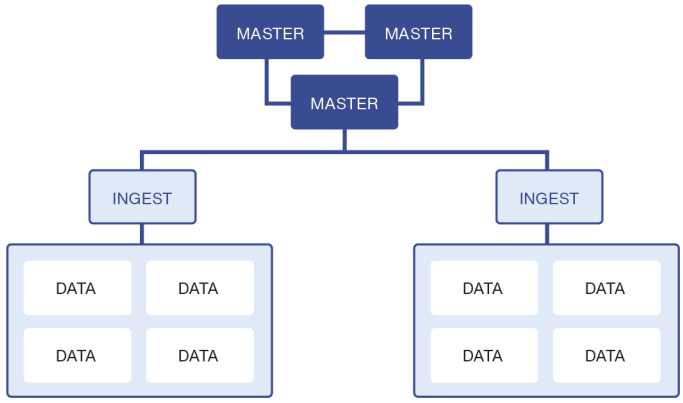
2.2. Indeksowanie danych – Elasticsearch

2.2.1. Architektura klastra

Elasticsearch[1] to usługa pełnotekstowego wyszukiwania oparta na bibliotece Lucene, która jest skalowalna i udostępnia wszystkie swoje funkcje za pośrednictwem formatu JSON i dedykowanych zestawów API. Może służyć do wyszukiwania wszelkiego rodzaju dokumentów (w tym tekstowych). Zapewnia skalowalne wyszukiwanie w czasie zbliżonym do rzeczywistego. Posiada wbudowany mechanizm zarządzania węzłami (rozproszenie klastra), co oznacza, że indeksy można podzielić na fragmenty, a każdy fragment może mieć zero lub więcej replik. Każdy węzeł obsługuje jeden lub więcej fragmentów i działa jako koordynator w celu delegowania operacji do odpowiedniego fragmentu. Równoważenie i trasowanie odbywa się automatycznie. Powiązane dane są często przechowywane w tym samym indeksie, który składa się z co najmniej jednego fragmentu podstawowego i zera lub większej liczby fragmentów replik. Rozproszona architektura Elasticsearch zapewnia istnienie trzech rodzajów węzłów (master, data, ingest):

- ingest – używany, aby wstępnie przetworzyć dokumenty przed faktycznym ich zaindeksowaniem;
- data – używany, aby przechowywać i przeglądać dane; każdy z węzłów posiada domyślnie włączoną opcję „data", co jest spowodowane dużym rozmiarem zbioru danych;
- master – służy do zarządzania całym klastrem, w klastrze może być kilka węzłów typu „master".

W trakcie prac nad systemem uzyskano najbardziej wydajną postać architektury klastra, która składa się z trzech węzłów typu „master", siedmiu węzłów typu „ingest" oraz trzydziestu węzłów typu „data" (patrz Ilustracja 2.1)



Ilustracja 2.1. Architektura klastra Elasticsearch użytego na potrzebę JSA

[1] elastic.co/elasticsearch (dostęp: 18.11.2020)

Architekturę tak zaprojektowaną można dowolnie rozszerzać dodając kolejne węzły typu „ingest” oraz „data”. Maszyny jakie obsługują węzły powinny dysponować dużą liczbą rdzeni procesora, oraz wysokowydajnymi dyskami SSD. Dzięki zastosowaniu takich wielu maszyn, możliwe jest przechowanie i szybkie przeszukiwanie indeksów o rozmiarze globalnym liczonym w dziesiątkach TB.

2.2.2. Ładowanie danych

Danymi, które są ładowane do klastra Elasticsearcha są dokumenty tekstowe (np. teksty prac dyplomowych, czy zrzuty stron internetowych, a właściwie teksty wydobyte z tych stron). Każdy oryginalny tekst pracy/strony jest podzielony na pięciozdaniowe kawałki z użyciem okna przesuwnego, które następnie są uzupełniane o dodatkowe metadane, takie jak identyfikator dokumentu źródłowego (document_id), URL skąd został pobrany ten dokument, czas modyfikacji (pobrania z sieci danego dokumentu – modification_timestamp), oraz który jest to fragment (chunk_id).

Kod 2.1. Przykład dokumentów w rozumieniu Elasticsearcha

```
"hits" : [
  {
    "_index" : "internet",
    "_type" : "document",
    "_id" : "4f167772f3ca97788c20a3228c7a3fa9e18425adac5a98eae8a1301a67c9bc29",
    "_score" : 1.0,
    "_source" : {
      "modification_timestamp" : "17-05-2018, 05:05:23",
      "document_id" :
           "5481e541377a7d6d7d8e45f37711c40a795c05c5510ccefa26159540b580cb55",
      "type" : "text/html",
      "index_id" :
           "4f167772f3ca97788c20a3228c7a3fa9e18425adac5a98eae8a1301a67c9bc29",
      "url" : "http://jsa-strona-internetowa.pl?asc=tak",
      "content" : " jeżeli dodałeś ofertę i nie pojawiła się - odśwież stronę
           sprzedaż wynajem; mieszkania domy działki lokale i obiekty numer
           oferty rejon miejscowość/dzielnica typ budynku pow.  całkowita
           od:do: liczba pokoi od:do: cena za m2 [pln] od:do: zapraszamy do
           zapoznania się z naszą ofertą  zapraszamy do współpracy
           nieruchomości rejon  miejsc",
      "chunk_id" : "5"
    }
  }
]
```

2.2.3. Wyszukiwanie kandydatów na dokumenty źródłowe w JSA

Wyszukiwanie dokumentów, wśród których istnieje podejrzenie, że to z nich zaczerpnięto tekst do badanego dokumentu odbywa się za pomocą istniejących mechanizmów silnika Elasticsearch i metody MoreLikeThis [8, 10].

Stosowana przez nas metoda działa w oparciu o wyszukiwanie najlepszych dopasowań dokumentów źródłowych do dokumentu, który jest użyty w zapytaniu. Odbywa się to przy użyciu natywnej formuły punktacji wyników wyszukiwania Lucene[2] oraz z zastosowaniem ważenia słów miarą TF-IDF (*Term Frequency – Inverse Document Frequency*)[18]. Jest to jedna ze statystycznych metod obliczania istotności słowa/kolekcji słów w oparciu o liczbę ich wystąpień w samym dokumencie jak i całym korpusie. TF-IDF informuje o częstości wystąpienia słowa w ramach zdania/akapitu/dokumentu uwzględniając jednocześnie odpowiednie wyważenie jego znaczenia w kontekście pełnej kolekcji dokumentów.

Dla zadanego tekstu z dokumentu źródłowego zostaje zidentyfikowany zestaw słów kluczowych, które najlepiej opisują dokument (jest to realizowane przy pomocy miary TF-IDF), które to słowa służą do znalezienia dokumentów podobnych w indeksie. Wykorzystując tę metodę w sposób bezpośredni do całej pracy dyplomowej nie można było efektywnie znajdować całych prac/stron internetowych, które są podobne – powodem jest duża ilość tekstu i rozmywanie się znaczenia miary TF-IDF, co wpływa na zaburzenie identyfikacji słów kluczowych. Rozwiązaniem tego problemu jest podzielenie badanej pracy na ograniczone rozmiarem fragmenty tekstu (okna długości 5 zdań). Cały indeks jest zbudowany z nowej klasy dokumentów będących fragmentami oryginalnych tekstów prac dyplomowych czy stron internetowych o długości 5 zdań, co wpływa na jego olbrzymi rozmiar liczony w dziesiątkach miliardów.

Cała procedura wygląda w uproszczeniu następująco:

- tekst badanej pracy jest dzielony na fragmenty o rozmiarze 5 zdań,
- dla każdego fragmentu szukane są w indeksie podobne dokumenty (w indeksie dokument to pięciozdaniowe okno oryginalnego tekstu dokumentu źródłowego dotyczącego pracy czy strony internetowej),
- każdy wyszukany dokument w indeksie zawiera informacje z jakiego oryginalnego dokumentu źródłowego pochodzi,
- oryginalne dokumenty źródłowe porządkowane są malejąco ze względu na liczbę ich wystąpień w procesie powyższego iteracyjnego wyszukiwania dla całego tekstu badanej pracy,
- do dalszego etapu porównywania przechodzi 100 najczęściej występujących oryginalnych dokumentów źródłowych.

Wyszukiwanie w taki sposób odbywa się na wszystkich dostępnych źródłach silnika Elasticsearch (ORPPD, NEKST, DOAJ, Wikipedie, akty prawne).

Użyte ustawienia opcji wyszukiwania, które osiągnęły najlepsze wyniki podczas ewaluacji jakościowej systemu to:

- maxQueryTerms: 100,
- minDocFreq: 5,

[2] lucene.apache.org (dostęp: 18.11.2020)

- minTermFreq: 2,
- minShouldMatch: 90%.

2.3. Detekcja podobnych pasaży między dwoma dokumentami

2.3.1. Opis problemu

Algorytm detekcji pasaży podobnych (ang. *text-alignment detection*) identyfikuje pary fragmentów podobnych między dwoma dokumentami, które nazywamy odpowiednio dokumentem podejrzanym (ang. *suspicious document*) i dokumentem źródłowym (ang. *source document*). W systemie JSA algorytm ten wywoływany jest w kroku następującym po wytypowaniu kolekcji dokumentów, które stanowią potencjalne źródło zapożyczeń. Na tym etapie system *nie wie* niczego o lokalizacji zapożyczeń między dokumentem podejrzanym a dokumentami źródłowymi. Celem omawianego tu algorytmu jest właśnie dokładne zlokalizowanie tych fragmentów dokumentów, które mogły być splagiatowane.

2.3.2. Specyfikacja wejścia i wyjścia algorytmu

CZYM JEST DOKUMENT?

W kontekście omawianego tu algorytmu, dokument jest utożsamiany ze strukturą danych typu String, tzn. dokument jest skończonym ciągiem o wartościach w skończonym zbiorze symboli. W dalszej części podrozdziału wyrażenie *dokument* będzie oznaczało właśnie tę strukturę danych. Ponadto dla ustalonego dokumentu X przyjmujemy następujące oznaczenia:

- $\mathrm{len}(X)$ oznacza długość dokumentu X (tzw. *długość znakowa*);
- $X[i]$ oznacza i-tą pozycję dokumentu X, gdzie $i = 0, \ldots, (\mathrm{len}(X) - 1)$;
- $X[i..j]$ oznacza dokument Y, gdzie $0 \leq i < j < \mathrm{len}(X)$ oraz $Y[k] = X[i + k]$ dla $0 \leq k < j - i$. Tak zdefiniowany dokument względem ustalonego dokumentu X będziemy w dalszej części nazywali *fragmentem dokumentu* X.

Do opisu algorytmu warto zdefiniować jeszcze następujące relacje dla dokumentów X, Y:

- $X \subseteq Y$ oznacza, że dokument X jest *fragmentem dokumentu* Y, tzn. $Y = X[i..j]$, dla pewnych i, j takich że $0 \leq i \leq j < \mathrm{len}(X)$;
- $\alpha = \beta$ oznacza, że dokumenty α i β są *równokształtne*, tzn. $\mathrm{len}(\alpha) = \mathrm{len}(\beta)$ oraz $\alpha[i] = \beta[i]$ dla $i = 0, \ldots, \mathrm{len}(\alpha)$;
- powiemy, że fragment $\alpha = X[i..j] \subseteq X$ *poprzedza* fragment $\alpha' = X[k..l] \subseteq X$, jeżeli $j \leq k$. W dalszej części dla tej relacji będziemy stosować zapis $\alpha < \alpha'$.

Na przykład, przyjmijmy, że dokumentem X jest zdanie *Ala ma kota*, wówczas: $\mathbf{len}(X) = 11$, $X[0] = A$, $X[7..10] = kota$, $kot \subseteq X$, $Ala < kot$.

Wprowadzamy też funkcję odległości między fragmentami tekstu oraz funkcję rzutowania dla pary:

- funkcja odległości oznaczana przez $\mathbf{dist}$ określona jest następująco: $\mathbf{dist}(\alpha, \beta) = k - j$, gdzie $\alpha, \beta = X[i..j], X[k..l] \subseteq X$ i $\alpha < \beta$, tzn. jest ona określona dla rozłącznych fragmentów tekstu.
- funkcja rzutowania dla pierwszej i drugiej współrzędnej pary oznaczana przez π_0, π_1 określona jest następująco: $\pi_0(x, y) = x$, $\pi_1(x, y) = y$.

Obraz zbioru X względem dowolnej ustalonej funkcji f zapisujemy jako $f[X]$, tj. $f[X] = \{f(x) : x \in X\}$.

SPECYFIKACJA WEJŚCIA I WYJŚCIA ALGORYTMU

Algorytm detekcji pasaży podobnych – w dalszej części podrozdziału – oznaczany symbolem $\mathcal{A}$ przyjmuje na wejściu parę dokumentów (X, Y) i stałą $c > 0$, a w wyniku obliczenia zwraca listę par fragmentów pary (X, Y) o długościach znakowych nie mniejszych niż c, gdzie fragmenty dokumentu X są rozłączne względem siebie, to jest zbiór $R = \{(\alpha, \beta) : \alpha \subseteq X \wedge \beta \subseteq Y\}$, taki że

$$\forall_{\alpha,\alpha' \in \pi_0[R]} (\alpha < \alpha' \vee \alpha' < \alpha). \tag{2.1}$$

Wynik obliczenia $\mathcal{A}$ na wejściu (X, Y, c) oznaczamy przez $\mathcal{A}_c(X, Y)$. Powyższa definicja dopuszcza niesymetryczność działania algorytmu względem wejściowej pary, co oznacza, że możliwe jest $\mathcal{A}_c(X, Y) \neq \mathcal{A}_c(Y, X)$. Ta charakterystyka opiera się na założeniu, że fragmenty z dokumentu źródłowego mogą być plagiatowane w wielu miejscach dokumentu badanego, natomiast splagiatowany fragment ma dokładnie jedno źródło, tzn. relacja między fragmentami splagiatowanymi dokumentu podejrzanego a dokumentu źródłowego jest typu wiele-jeden.

Fragmenty zwracane na wyjściu algorytmu $\mathcal{A}$ będziemy nazywać *detekcjami* a fragmenty, które stanowią faktyczny plagiat – *plagiatami*. Przyjmijmy, że dla dokumentów X, Y dany jest zbiór *plagiatów* $\mathcal{P}$ (np. wskazane przez skruszonego plagiatora miejsca zapożyczeń) i ich detekcje $\mathcal{D} = \mathcal{A}_c(X, Y)$. Te dwa zbiory wyznaczają tzw. macierz błędu, tzn. zbiór $\mathcal{P}$ wyznacza klasę rzeczywistą (tj. klasę prawdziwie pozytywną wraz z prawdziwie negatywną), a zbiór $\mathcal{D}$ klasę predykowaną (tj. klasę fałszywie pozytywną wraz z fałszywie negatywną):

- klasa *prawdziwie pozytywna* oznaczana przez tp to zbiór $\mathcal{P}$;
- klasa *prawdziwie negatywna* oznaczana przez tf to zbiór $(X, Y) \setminus \mathcal{P}$;
- klasa *fałszywie pozytywna* oznaczana przez fp to zbiór $\mathcal{D}$;
- klasa *fałszywie negatywna* oznaczana przez fn to zbiór $(X, Y) \setminus \mathcal{D} \setminus$ tn.

2.3.3. Zarys działania algorytmu

Kluczowym pojęciem, którego implementacja decyduje o wynikach działania algorytmu jest *podobieństwo tekstów*. W literaturze przedmiotu wyróżnia się podobieństwa *leksykalne* i *semantyczne*, pierwsze określające podobieństwo w oparciu o kształt porównywanych tekstów, a drugie w oparciu o podobieństwo znaczeń. Przegląd algorytmów obliczających podobieństwo dla obydwu tych typów dostępny jest np. w [9].

W implementacji algorytmu w systemie JSA zastosowano podejście hybrydowe o dwufazowym przebiegu. Takie podejście integruje dwa algorytmy, z których pierwszy należy do rodziny algorytmów implementujących podobieństwo leksykalne, a drugi do rodziny algorytmów implementujących podobieństwo semantyczne.

Celem takiego hybrydowego podejścia jest zapewnienie wysokiej granularności detekcji, w której fragmenty identyczne zostają wyodrębnione spośród pozostałych wyników[3]. Obydwa te algorytmy, a także algorytm integrujący ich działanie, spełniają opisaną wcześniej specyfikację wejścia/wyjścia, przyjmijmy, więc dla pierwszego i drugiego z nich następujący zapis: $\mathcal{A}_c^{lex}, \mathcal{A}_c^{sem}$. Z kolei algorytm, który integruje działanie $\mathcal{A}_c^{lex}, \mathcal{A}_c^{sem}$ oznaczymy przez $\mathcal{A}_c^{jsa}$. Przejdziemy teraz do zarysu działania obu tych algorytmów oraz algorytmu integrującego obydwa te podejścia.

ALGORYTM $\mathcal{A}_C^{LEX}$

W pierwszej fazie identyfikacji podobieństw używany jest algorytm wskazujący pary fragmentów identycznych pomiędzy dwoma dokumentami. Opiera się on z kolei na algorytmie wyszukiwania najdłuższych wspólnych podciągów pomiędzy parą łańcuchów znaków tzn. algorytmie, który na wejściu przyjmuje parę dokumentów (X, Y) i zwraca

$$(\alpha, \beta) = \max_{\text{len}(\alpha)} \{(\alpha, \beta) : \alpha \subseteq X \wedge \beta \subseteq Y \wedge \alpha = \beta\}. \tag{2.2}$$

Implementacja tego algorytmu została z kolei zrealizowana w oparciu o metodę wyszukiwania najdłuższych podstringów w posortowanej tablicy sufiksów, dalej $\mathcal{LCCS}$ ([14]). Konstrukcja $\mathcal{A}_c^{lex}$ opiera się na rekurencyjnym wywoływaniu $\mathcal{LCCS}$ w toku, którego do kolejnych wywołań przekazany zostaje iloczyn kartezjański fragmentów stanowiących dopełnienie fragmentów wskazanych w wyniku poprzedniego wywołania $\mathcal{LCCS}$. Rekurencyjne wywołanie zostaje przerwane a wynik $\mathcal{A}_c^{lex}$ zwrócony, gdy długość znakowa zwróconych wspólnych fragmentów osiągnie długość określoną parametrem c.

Pewną subtelnością w tej fazie jest to, że porównanie tekstów opiera się na zadanym alfabecie, jeżeli więc alfabet składa się z wielkich i małych liter języka polskiego,

[3] W terminologii przyjętej w systemie JSA fragmenty identyczne są określane jako *cytowania*. Do tej klasy zaliczają się faktyczne cytowania, jak i zapożyczenia typu kopiuj-wklej. Właściwe zaklasyfikowanie takiego fragmentu jest pozostawione w gestii promotora.

to wyrażenia *Ala ma kota* i *Alamakota* zostaną w tej fazie zwrócone jako wynik (tzn. algorytm w tym przypadku ignoruje znak spacji, porównując jedynie litery z alfabetu).

ALGORYTM $\mathcal{A}_C^{SEM}$

W działaniu algorytmu $\mathcal{A}_c^{sem}$ – wzorowanym na rozwiązaniu zaproponowanym w [21] – można wyodrębnić dwie główne fazy. Pierwsza faza odpowiada za podział dokumentów na zdania oraz zmapowanie tych zdań na ich reprezentację wektorową [11]. Mapowanie zdań na przestrzeń wektorową odbywa się poprzez przypisanie do słów w nich występujących wartości z przedziału $[0,1]$, która przybliża ich semantyczną istotność, wnioskowaną z ich rozkładu statystycznego w zdaniach. Najniższą wartość otrzymują słowa o najmniejszej lub największej częstotliwości występowania w zdaniach, np. nazwy własne (jako wyrażenia o małej częstotliwości występowania) lub spójniki (jako wyrażenia o dużej częstotliwości występowania). W drugiej fazie działania algorytmu elementarnymi danymi są pary zdań obydwu dokumentów wraz z przypisaną im reprezentacją wektorową, zbiór zdań dokumentu X oznaczamy w dalszej części przez S_X. Pierwszy krok drugiej fazy polega na selekcji par zdań z obydwu dokumentów, które spełniają warunek podobieństwa zdefiniowany jako odległość nieprzekraczająca progów $\theta_{\cos}, \theta_{\mathrm{dice}}$ dla miary cosinusowej i miary $\mathbf{dice}$, tj. dla dokumentów X, Y otrzymujemy:

$$\{(\alpha,\beta) \in \mathrm{S}_X \times \mathrm{S}_Y : \cos(\alpha,\beta) > \theta_{\cos} \wedge \mathbf{dice}(\alpha,\beta) > \theta_{\mathrm{dice}}\}, \tag{2.3}$$

gdzie

$$\cos(\alpha,\beta) = \frac{\alpha \cdot \beta}{\|\alpha\| \cdot \|\beta\|} = \frac{\sum_{i=0}^{n} \alpha_i \cdot \beta_i}{\sqrt{\sum_{i=0}^{n} \alpha_i^2} \cdot \sqrt{\sum_{i=0}^{n} \beta_i^2}},\ \alpha,\beta \in \mathbf{R}^{n+1}, \tag{2.4}$$

oraz

$$\mathbf{dice}(\alpha,\beta) = \frac{2|\delta(\alpha) \cap \delta(\beta)|}{|\delta(\alpha)| + |\delta(\beta)|}, \tag{2.5}$$

gdzie

- $|\cdot|$ to moc zbioru;
- $\delta(\cdot)$ to przyporządkowanie do danego fragmentu zbioru słów w nim występujących, np. $\delta(\textit{Ala ma kota}) = \{\textit{Ala}, \textit{ma}, \textit{kota}\}$.

W kolejnych krokach algorytm rozszerza początkowy zbiór poprzez klastrowanie tych par zdań z inicjalnego zbioru, których odległość w tekście między zdaniami nie przekracza wartości ustalonej parametrem θ_{dist}, tzn. klastry $C, C' = \{(\alpha,\beta)\}, \{(\alpha',\beta')\} \subseteq \mathrm{S}_X \times \mathrm{S}_Y$ zostaną połączone w nowy klaster ze względu na pierwszą (drugą) współrzędną, jeżeli zostanie spełniony następujący warunek:

$$\min\{\mathrm{dist}(\gamma,\gamma') : (\gamma,\gamma') \in \pi_i[C] \times \pi_i[C']\} < \theta_{\mathrm{dist}},\ i = 0,1. \tag{2.6}$$

Klastrowanie odbywa się naprzemiennie, tzn. raz według odległości pomiędzy zdaniami pierwszego dokumentu, a następnie według odległości pomiędzy zdaniami drugiego dokumentu. Tak utworzone klastry wyznaczają w obu tekstach fragmenty, które można

poddać walidacji. Walidacja wykonywana jest przez przypisanie do klastra C wartości określającej jakość podobieństwa między jego parami zdań w klastrze funkcją $q(\cdot)$:

$$q(C) = \cos\left(\sum_{\alpha \in \pi_0[C]} \alpha, \sum_{\beta \in \pi_1[C]} \beta\right). \tag{2.7}$$

Jeżeli wartość $q(C)$ jest wystarczająco wysoka, klaster zostaje dodany do wyniku algorytmu. W przeciwnym przypadku następuje rekurencyjne wywołanie procedury klastrowania z dekrementowaną wartością dla progu θ_{dist} i dodanie jej wyniku do wynikowego zbioru. Rekurencje wywołania zostają przerwane a zakumulowane klastry zwrócone, gdy wartość θ_{dist} osiągnie określone parametrem minimum. Ostatnia faza algorytmu odpowiada za wyselekcjonowanie spośród otrzymanych klastrów takiego ich podzbioru, który maksymalizuje ich jakość $q(\cdot)$ przy równoczesnym spełnieniu warunku rozłączności fragmentów pierwszego dokumentu.

ALGORYTM $\mathcal{A}_C^{JSA}$

Działanie algorytmów $\mathcal{A}_c^{lex}, \mathcal{A}_c^{sem}$ w systemie JSA jest zintegrowane w algorytmie $\mathcal{A}_c^{jsa}$, który otrzymując na wejściu parę dokumentów X, Y przekazuje je najpierw do wejścia algorytmu $\mathcal{A}_c^{lex}$. Wynik działania $\mathcal{A}_c^{lex}$ na parze X, Y, czyli pary fragmentów równokształtnych, zostają użyte do wygenerowania wejścia dla $\mathcal{A}_c^{sem}$. Dane wejściowe dla $\mathcal{A}_c^{sem}$ zostają wygenerowane poprzez utworzenie wszystkich par między fragmentami otrzymanymi przez usunięcie z pierwotnych dokumentów wejściowych X, Y fragmentów zwróconych jako wynik działania pierwszego algorytmu. Na przykład jeżeli algorytm fazy pierwszej, tj. $\mathcal{A}_c^{lex}$, przyjął na wejściu parę dokumentów $(\alpha_0\alpha_1\alpha_2\alpha_3, \beta_0\beta_1\beta_2\beta_3)$[4] i zwrócił pary fragmentów $(\alpha_0, \beta_0), (\alpha_2, \beta_2)$, to w fazie drugiej nastąpi czterokrotne wywołanie algorytmu $\mathcal{A}_c^{sem}$ na parach $(\alpha_1, \beta_1), (\alpha_1, \beta_3), (\alpha_3, \beta_1), (\alpha_3, \beta_1)$. Jako wynik działania $\mathcal{A}_c^{jsa}$ zwrócona zostanie suma wyników uzyskanych przez $\mathcal{A}_c^{lex}$ i $\mathcal{A}_c^{sem}$.

W dalszym ciągu tego podrozdziału przez $\mathcal{A}_c$ będziemy oznaczać opisany tu algorytm $\mathcal{A}_c^{jsa}$.

2.3.4. Ewaluacja algorytmu

TRUDNOŚCI DEFINICYJNE ZJAWISKA PLAGIARYZMU

Jedną z zasadniczych trudności napotykanych w projektowaniu algorytmu do detekcji plagiatu jest nieostrość zjawiska plagiaryzmu ([5, 6, 4, 16]). Określenie plagiatem

[4] Zapis $\alpha_0\alpha_1\alpha_2\alpha_3(\beta_0\beta_1\beta_2\beta_3)$ oznacza konkatenację dokumentów $\alpha_0, \alpha_1, \alpha_2, \alpha_3(\beta_0, \beta_1, \beta_2, \beta_3)$

tekstu, który został utworzony przez skopiowanie i wklejenie fragmentu innego tekstu, jest raczej bezsporne. Także zastąpienie w tym tekście jednego wyrazu jego synonimem nie zmieni jego oceny jako plagiatu. Jednak dostateczna liczba wprowadzonych zmian (parafraz, synonimizacji) spowoduje, że wyjściowy tekst będzie istotnie odmienny od tekstu źródłowego i określenie go mianem plagiatu będzie dużo bardziej wątpliwe. Gdzie leży ta granica liczby zmian w tekście, której przekroczenie powoduje zmianę klasyfikacji tekstu jako plagiatu jest właśnie tym problemem nieostrości.

Przy całej tej definicyjnej trudności można jednak wskazać warunki minimalne, które musi spełniać algorytm. Te warunki to pełna wykrywalność fragmentów identycznych (z pominięciem fragmentów krótszych niż wartość zadana parametrem c), tzn. dla badanych dokumentów X, Y wynik $\mathcal{A}_c(X,Y)$ musi zawierać:

$$\{(\alpha,\beta) : \alpha \subseteq X \wedge \beta \subseteq Y \wedge \mathrm{len}(\alpha) \geq c \wedge \alpha = \beta\}. \tag{2.8}$$

Wskazanie definicji szerszej zakresowo, która obejmowałaby także plagiaty poddawane wspomnianym zmianom jest z wyżej wspomnianych powodów dużo bardziej problematyczne. Zamiast tego w przypadku oceny poprawności działania algorytmu $\mathcal{A}_c(\cdot,\cdot)$ dla tej klasy fragmentów przyjęto ewaluację opartą na mierze *plagdet* ([17]) obliczaną względem wynikowego zbioru fragmentów wskazanych przez $\mathcal{A}_c(\cdot,\cdot)$ a zbiorem fragmentów zadanych zbiorem walidacyjnym.

ZBIÓR WALIDACYJNY

Zbiór walidacyjny został przygotowany w oparciu o 3 rodzaje przekształceń tekstu: identycznościowe (kopiuj-wklej), zamianę szyku wyrazów oraz synonimizację (zamianę wyrażeń w tekście na wyrażenia synonimiczne). Zbiór walidacyjny składał się z 1336 par dokumentów wraz z zapisanymi w plikach xml pozycjami splagiatowanych fragmentów (tp).

MIARA *PLAGDET*

Miara *plagdet* inkorporuje bardziej podstawowe miary *precision*, *recall*, *granularity*. Każda z nich jest obliczana w oparciu o zbiór detekcji i zbiór walidacyjny dla ustalonej pary dokumentów (X,Y). Miary *precision*, *recall* wyrażają odpowiednio stosunek poprawnych detekcji względem wszystkich detekcji ($\frac{\mathrm{tp}}{\mathrm{tp+fp}}$) i stosunek poprawnych detekcji względem wszystkich zapożyczeń ($\frac{\mathrm{tp}}{\mathrm{tp+fn}}$), obie miary przyjmują wartości z przedziału $[0,1]$, gdzie wartość 0 oznacza całkowite chybienie (fragment zwrócony przez algorytm jest rozłączny z faktycznym fragmentem splagiatowanym), a wartość 1 pełną poprawność detekcji (fragment zwrócony przez algorytm pokrywa się całkowicie z faktycznym fragmentem splagiatowanym). Miara *granularity* to stopień przykrycia splagiatowanego fragmentu przez detekcję z wartościami w przedziale $[1,n]$, gdzie n to liczba wynikowych detekcji algorytmu. Dla miary $granularity_{X,Y}$ Wartość 1 oznacza, że każda detekcja odpowiada dokładanie jednemu plagiatowi, a wartość n rozdrobnienie wyniku detekcji na

małe fragmenty. Formuła miary *plagdet* jest średnią miary *precision* i *recall*, z czynnikiem *granularity* penalizującym wysoką granularność detekcji:

$$\frac{precision_{X,Y} + recall_{X,Y}}{2log_2(1 + granularity_{X,Y})}, \tag{2.9}$$

przyjmując wartości w przedziale [0, 1]. Pełna formalna definicja metryk jest zamieszczona w artykule [17][5].

Tabela 2.3. Wyniki algorytmu $\mathcal{A}_c(\cdot,\cdot)$ na zbiorze walidacyjnym z parametrem $c = 200$ dla metryki *plagdet* (2.9) i jej składowych *precision*, *recall*, *granularity*

plagdet	0,902
precision	0,927
recall	0,939
granularity	1,047

2.3.5. Adaptacja algorytmu w ramach systemu JSA

SELEKCJA FRAGMENTÓW W WARSTWIE PREZENTACJI

W toku każdorazowego badania pracy w systemie JSA algorytm detekcji $\mathcal{A}_c$ zostaje wywołany tyle razy, ile zostało zwróconych potencjalnych źródeł zapożyczeń w poprzednim etapie działania systemu, tzn. jeżeli X jest badaną pracą a $\mathcal{Y}$ zbiorem dokumentów wytypowanych jako potencjalne źródła zapożyczenia przez wcześniej wywołany moduł systemu, to algorytm $\mathcal{A}_c$ zostanie wywołany na parach $\{X\} \times \mathcal{Y}$. W konsekwencji frontend musi zaprezentować użytkownikowi fragmenty tekstu należące do zbioru $\bigcup_{Y \in \mathcal{Y}} \mathcal{A}_c(X, Y)$. Ze względu na to, że w żaden sposób nie jest wykluczone, by w zbiorze tym występowały fragmenty nachodzące na siebie, pojawia się tu problem prezentacji tych danych w warstwie frontendowej. W takich przypadkach system realizuje przetwarzanie końcowe nachodzących na siebie fragmentów w oparciu o przypisaną im wartość liczbową, która przybliża stopień podobieństwa fragmentów. Tę przyporządkowaną wartość liczbową do danej pary fragmentów będziemy określać dalej jako *ranking*. Jest on zdefiniowany jako wprowadzona już miara dice, różniąc się tym, że oblicza się go na parze dowolnych fragmentów tekstu, a nie jak wcześniej na parze zdań. Jeżeli w wyniku zwróconym po przebadaniu pracy wystąpią fragmenty nachodzące na siebie w pracy badanej, to przetwarzanie końcowe będzie polegało na wskazaniu istotniejszego fragmentu (istotność jest rozumiana jako wyższa wartość liczbowa) i obcięciu mniej istotnego fragmentu do miejsca (od miejsca), w którym kończy się (zaczyna się) fragment z wyższą wartością rankingową. Ten sposób wyboru ma

[5] Ewaluator obliczający te metryki dostępny jest na konkursowej stronie systemu JSA, konkurs-jsa.opi.org.pl (dostęp: 18.11.2020)

zapewnić, aby użytkownik z dwóch nachodzących na siebie fragmentów otrzymał ten o wyższym podobieństwie.

PARAMETRYZACJA STOPNIA PODOBIEŃSTWA

Drugim zastosowaniem rankingowania fragmentów jest umożliwienie użytkownikowi filtrowania wyników badania ze względu na stopień podobieństwa. Służy temu czterostopniowa skala o dyskretnej liście wartości poziomu podobieństwa: *niski*, *średni*, *wysoki*, *bardzo wysoki*. Wybór wartości poziomu podobieństwa na *bardzo wysoki* zwraca pary fragmentów typu kopiuj-wklej, wybór niższych poziomów obniża restrykcyjność filtrów, aż do poziomu *niski* nie wpływającego na wynik badania. Taka zależność powinna skutkować dodatnią korelacją w wyborze poziomu podobieństwa a miary *precision*, *granularity*. Przy tym pierwsza zależność podnosi z kolei wartość miary *plagdet* (w przypadku, gdy pozostałe jej składowe są ustalone), a druga obniża (także przy tym zastrzeżeniu). Ujemną korelację powinniśmy zaobserwować z kolei między poziomem podobieństwa a miarą *recall*, co z kolei obniża wartość miary *plagdet* (przy wskazanym wcześniej zastrzeżeniu). Wyniki algorytmu $\mathcal{A}_c$ uzyskane na zbiorze walidacyjnym przy różnych wartościach poziomu podobieństwa potwierdzają tę prostą hipotezę.

Tabela 2.4. Wyniki algorytmu $\mathcal{A}_c$ ze względu na różne poziomy podobieństwa (niskie, średnie, wysokie, bardzo wysokie) dla metryk *plagdet*, *precision*, *recall*, *granularity*

	niskie	średnie	wysokie	bardzo wysokie
plagdet	0,902	0,895	0,821	0,443
precision	0,927	0,952	0,967	0,997
recall	0,939	0,900	0,749	0,109
granularity	1,047	1,047	1,040	1,375

2.4. Stylometria

Na ogół problem wykrywania plagiatów w pracach dyplomowych kojarzony jest z porównywaniem badanej pracy względem innych prac z dużego korpusu dokumentów, np. zbioru obronionych prac dyplomowych czy treści pochodzących z internetu. Badanie pracy dyplomowej pod względem zapożyczeń nie musi ograniczać się tylko do wspomnianego sposobu poszukiwań plagiatów. Innym podejściem wskazania fragmentów w pracy dyplomowej, które mogły zostać zapożyczone, jest użycie analizy stylometrycznej.

Stylometria polega na wykrywaniu fragmentów napisanych w pracy przez innego autora bądź autorów. W analizie stylometrycznej badaniu podlega tylko interesujący dokument, a badanie tekstu opiera się wyłącznie na zmianach stylistycznych i niespójnościach w obrębie tego dokumentu bez odniesienia do innych źródeł.

Metoda użyta w Jednolitym Systemie Antyplagiatowym opiera się na wyznaczaniu zmian stylu autora pracy w dokumencie za pomocą n-gramowych profili znakowych i zdefiniowaniu funkcji zmienności stylu w oparciu o miarę podobieństwa, która pozwala na wskazanie fragmentów niepasujących stylem do całej treści pracy i podejrzanych o plagiat.

2.4.1. Założenia

Głównym założeniem wyznaczenia fragmentów podejrzanych o plagiat przy użyciu analizy stylometrycznej jest brak porównania analizowanego tekstu z innymi dokumentami. Analiza odbywa się w oparciu tylko o analizę tekstu danej pracy.

W literaturze możemy znaleźć systematyczny podział cech pozwalający na ilościowe określenie stylu autora (por. [23]). Cechy te dzielimy na m.in.:

- leksykalne (np. długość słów lub zdań, wyrazy, n-gramy wyrazowe);
- znakowe (np. rodzaj znaku (litera, liczba, itd.), n-gramy znakowe);
- składniowe (np. części mowy, zdania, wyrażenia);
- semantyczne (np. synonimy, semantyczne zależności).

Cechy leksykalne i znakowe traktują tekst jako sekwencje słów lub znaków i jest to najprostszy i wydaje się naturalny sposób analizy tekstu. Dla ujęcia leksykalnego tekst jest reprezentowany jako sekwencja tokenów pogrupowanych w zdania, w których każdy element odpowiada słowu, cyfrze lub znakowi interpunkcyjnemu. W przypadku ujęcia znakowego, tekst jest postrzegany jako ciąg znaków, w ten sposób można zdefiniować różne miary na poziomie znaków, m.in. liczbę liter, liczbę cyfr, liczbę wielkich i małych liter, częstość liter, liczbę znaków interpunkcyjnych, itd. Istotną zaletą obu podejść jest możliwość zastosowania ich dla dowolnego języka i dla dowolnego zbioru dokumentów bez dodatkowych wymagań.

Bardziej złożoną metodą reprezentacji tekstu jest wykorzystanie informacji składniowych i semantycznych. Intuicyjnie zakładamy, że każdy autor ma swój unikalny styl pisania, między innymi, autorzy mają tendencję do nieświadomego stosowania podobnych wzorców składniowych. Dlatego takie informacje mogą być również przydatne w określaniu stylu autora, a nierzadko okazują się wiarygodniejsze niż cechy leksykalne i znakowe. Z drugiej strony ekstrakcja tego rodzaju cech wymaga dostępności narzędzi NLP, dzięki którym będziemy w stanie przeprowadzić analizę składniową i semantyczną tekstu. Fakt ten ponadto oznacza, że ekstrakcja tych cech jest procedurą zależną od języka.

W czasie budowy silnika do wykrywania plagiatów w Jednolitym Systemie Antyplagiatowym wykonaliśmy liczne testy celem wyznaczenia najlepszego podzbioru cech pozwalających na określenie stylu autora i detekcji fragmentów odstających od niego. Najlepsze rezultaty otrzymywaliśmy dla n-gramów znakowych oraz przy wyznaczaniu części mowy i te cechy wykorzystaliśmy jako ostateczną formę reprezentacji tekstu. Co ciekawe, dodanie informacji na temat wielkości liter, czy interpunkcji, które wydają się być istotnymi cechami stylometrycznymi, nie przynosiła nam istotnej poprawy wyników, z racji zmniejszenia złożoności obliczeniowej zrezygnowaliśmy z tych cech w dalszej analizie.

Kolejnym ważnym założeniem wykorzystanego przez nas algorytmu jest przyjęcie, że 70% pracy powinno być napisane przez jednego, „głównego" autora, jedynie pozostałe 30% może być (ale nie musi być) napisana przez inne osoby, tzn. te fragmenty uznajemy jako zapożyczone z innych źródeł. W przypadku prac dyplomowych takie założenie wydaje się być zasadne. Rzadko kiedy mamy do czynienia z plagiatami całych prac dyplomowych (chociaż takie przypadki również się zdarzają), częściej są to kilkuzdaniowe zapożyczenia. Ponadto, założenie o głównym autorze pracy jest niezbędne do zdefiniowania „głównego stylu pracy'' w celu identyfikacji fragmentów, które pod względem stylu różnią się od tego nadrzędnego stylu. Gdybyśmy pominęli to założenie, ciężko byłoby stwierdzić, co jest naszym punktem odniesienia.

W literaturze można również znaleźć przykłady badań na temat granic zmiany stylu (*style breach detection*, [12]), jednakże w przypadku prac dyplomowych zostajemy przy założeniu, że praca została w większej części napisana przez swojego autora, natomiast celem analizy stylometrycznej jest wykrycie potencjalnych fragmentów zapożyczonych.

2.4.2. Funkcja zmiany stylu

Główna idea algorytmu wykorzystanego do analizy stylometrycznej w Jednolitym Systemie Antyplagiatowym opiera się na podziale tekstu pracy na fragmenty i porównanie wyznaczonych fragmentów z tekstem całej pracy poprzez użycie miary podobieństwa, która rozstrzyga, czy rozważany fragment jest podobny stylem względem całej pracy, czy różni się od niej na tyle, że można uznać go za fragment podejrzany (por. [22]).

Niech dokument D będzie reprezentowany przez zbiór pewnej cechy stylometrycznej, zbiór ten zwany jest profilem tekstu. Dodatkowo dla tego dokumentu wyznaczamy „wektor znormalizowanych częstotliwości", tj. częstotliwość występowania rozważanej cechy w danym fragmencie tekstu znormalizowaną przez długość tego fragmentu. Następnie dokument D dzielimy na „ruchome okna" (tj. fragmenty: paragrafy, pasaże o ustalonej liczbie zdań, fragmenty o takiej samej długości znaków). Niech W będzie zbiorem wszystkich fragmentów w dokumencie D, które zostały otrzymane poprzez przesuwanie okna o długości l w kroku o długości s. Łatwo zauważyć, że jeśli $l > s$, wtedy kolejne fragmenty tekstu nachodzą na siebie.

Niech $p(A)$ oraz $p(B)$ określają, odpowiednio, profil fragmentu A oraz B ze zbioru W. Wówczas przez $m(A,B)$ definiujemy miarę podobieństwa pomiędzy fragmentem A i B (por. [22]), daną wzorem:

$$m(A,B) = \frac{\sum_{g \in p(A)} \left(\frac{f_A(g) - f_B(g)}{f_A(g) + f_B(g)} \right)^2}{|l(A)|}, \tag{2.10}$$

gdzie $f_A(g)$ oraz $f_B(g)$ to odpowiednio, częstotliwość (normalizowana przez długość fragmentu) występowania cechy g we fragmencie A i B oraz $|l(A)|$ oznacza długość fragmentu A.

Warto zaznaczyć, że miara $m(A,B)$ jest niesymetryczna – fragment B jest sprawdzany względem fragmentu A. Jeśli fragmenty A i B mają podobny styl, wówczas wartości $f_A(g)$ oraz $f_B(g)$ odpowiednich wartości cechy powinny być zbliżone, czyli miara m powinna przyjmować małe wartości.

Następnie wyznaczamy funkcję zmiany stylu, która porównuje liczebności wystąpień wartości cechy w całym dokumencie do liczebności w ustalonym fragmencie. Funkcję zmiany stylu (*style change function*) w dokumencie D definiujemy wzorem [22]:

$$sc(i,D) = m(w_i, D), \quad i = 1, \ldots, |W|, \tag{2.11}$$

gdzie w_i to i-ty fragment ze zbiorów wszystkich fragmentów w dokumencie D, które zostały wyznaczone przy użyciu metody „ruchomego okna". Jeśli wartość funkcji dla pewnego okna jest „zbyt duża", wówczas taki fragment traktujemy jako tekst napisany innym stylem od głównego stylu pracy, tj. w rozumieniu przez innego autora.

2.4.3. Detekcja fragmentów podejrzanych

W Jednolitym Systemie Antyplagiatowym każdą pracę dyplomową reprezentujemy jako:

- zbiór n-gramów (zbudowanych ze znaków, a nie słów),
- zbiór części mowy (*part of speech*).

Dokument jest podzielony na fragmenty o określonej minimalnej długości zbudowane na podstawie pełnych zdań, zapewniając tym samym aby te fragmenty nie różniły się

istotnie między sobą długością. Dodatkowo minimalna długość fragmentu została wyznaczona eksperymentalnie, tak aby fragmenty swoją długością przypominały paragrafy. Ponadto każdy dokument jest „oczyszczany'': usuwamy akcenty, interpunkcje, znaki specjalne, niepotrzebne odstępy i wszystkie litery zostają zamieniane na małe.

Dla zbioru n-gramów wyznaczyliśmy funkcję zmiany stylu sc_1 natomiast dla zbioru części mowy funkcję zmiany stylu sc_2. Ponadto dla sc_1 wyznaczyliśmy średnią M_{sc_1} oraz odchylenie standardowe S_{sc_1}, dla sc_2 tylko odchylenie standardowe S_{sc_2}.

Zauważyliśmy, że wyznaczone odchylenie standardowe dla funkcji zmiany stylu części mowy istotnie wpływa na wskazanie prac, w których występują fragmenty podejrzane o zmianę stylu w stosunku do głównego stylu pracy. Stąd też kryterium określające, czy w danej pracy występują fragmenty różniące się stylem od głównego stylu pracy dyplomowej określamy jako:

$$S_{sc_1} > a \text{ i } S_{sc_2} > b. \tag{2.12}$$

Jeśli odchylenie standardowe dla sc_1 i sc_2 jest większe niż określony próg (może być różny dla każdej cechy), wówczas w badanej pracy występują fragmenty różniące się stylem od stylu całej pracy. Natomiast fragmenty różniące się stylem od całej pracy wyznaczamy jedynie na podstawie statystyk funkcji zmiany stylu dla n-gramów. Uznajemy, że fragment i-ty ze zbiorów wszystkich fragmentów w dokumencie D, które zostały otrzymane przy użyciu metody „ruchomego okna" różni się stylem od stylu głównego autora, gdy

$$sc_1(i, D) > M_{sc_1} + c \cdot S_{sc_1}. \tag{2.13}$$

Zarówno progi a, b jak i c zostały wyznaczone symulacyjnie. Warto jedynie zaznaczyć, że parametr c określa czułość metody wykrywania plagiatu. Im wyższa wartość c, tym mniej fragmentów innego stylu zostanie wskazanych.

Warto pamiętać, że badanie stylometryczne pracy dyplomowej nie jest decydującą metodą orzekającą o potencjalnym plagiacie. Fragmenty wyznaczone za pomocą opisanego algorytmu nie są brane do wyznaczania procentowego rozmiaru podobieństwa (PRP), a są jedynie częścią analizy tekstu wspólnie z analizą znaków specjalnych lub spoza języka pracy i nierozpoznanych wyrazów. Wyznaczone fragmenty innego stylu, pomagają promotorowi pracy na określenie czy dany fragment może być podejrzany o zapożyczenie z innego źródła z racji, że różni się stylem względem całości tekstu.

2.5. Chatbot JSA

2.5.1. Geneza zbudowania chatbota w JSA

Coraz częściej spotykaną formą komunikacji z użytkownikiem systemów informatycznych jest chatbot. Inteligentny program wykorzystujący sztuczną inteligencję i uczenie maszynowe. Specjalnie zaprogramowany automatyczny konsultant wchodzi w interakcję z użytkownikiem i jest dla niego dostępny bez przerwy, 24 godziny na dobę, 7 dni w tygodniu. Funkcji jakie może spełniać chatbot istnieje wiele. Programy takie znajdują obecnie zastosowanie m.in. przy składaniu szybko zamówień, weryfikacji ich bieżących statusów czy też odpowiedzi na mniej lub bardziej skomplikowane pytania użytkowników. Technologia chatbotowa z każdym rokiem zyskuje na popularności, a duże firmy i systemy informatyczne coraz częściej podejmują decyzję o udostępnieniu wirtualnego konsultanta. Ze względów społecznych osoby introwertyczne, ceniące sobie prostą wymianę informacji, dużo chętniej wybierają rozmowę z chatbotem niż rozmowę z konsultantem. To rozwiązanie jest obecnie najpopularniejsze w firmach nastawionych na handel (szczególnie handel internetowy), bankowość czy też telekomunikację. Jednak coraz częściej chatboty wprowadzane są do systemów opieki medycznej oraz administracji publicznej. W wyniku wstępnych rozważań poddano analizie częstotliwość kontaktów użytkowników systemu z zespołem OPI-Helpdesk oraz ruch i akcje wykonywane na stronach Centrum Pomocy JSA. Ponad 90 procent użytkowników JSA to promotorzy, którzy organizują swoją pracę często w niestandardowych godzinach. Oznacza to, że w razie napotkania problemów lub pytań użytkownicy muszą oczekiwać na kontakt z operatorem OPI-Helpdesk, podczas gdy bot może zapewnić natychmiastową pomoc. Obserwując nowe trendy oraz potrzeby użytkowników Jednolitego Systemu Antyplagiatowego, zapadła decyzja o zaprojektowaniu i udostępnieniu chatbota JSA.

Nadrzędnym celem programu chatbot JSA jest zapewnienie profesjonalnej, szybkiej i zawsze dostępnej pomocy. Dodatkowym atutem projektu jest możliwość zapewnienia użytkownikom uproszczonej komunikacji, skróceniu czasu i optymalizacji poszukiwania wiedzy w Centrum Pomocy JSA. Implementacja chatbota wprowadzi również dodatkowe możliwości analizy zachowań użytkowników i rozpoznanie najbardziej problematycznych zagadnień. Chatbot JSA może zapewnić nie tylko profesjonalną i stałą obsługę, ale również ciągłą naukę i rozwój zarówno swojej wiedzy jak i samego systemu JSA.

Głównym celem projektu, jak już zostało wspomniane, było zaprogramowanie konsultanta zdolnego do odszukania w istniejących zasobach informacyjnych odpowiedzi na pytania użytkowników. Istotnym aspektem jest w tym przypadku prędkość reakcji, tzn. czas od otrzymania polecenia, poprzez przeszukanie zasobów, a następnie do sporządzenia i wyświetlenia odpowiedzi nie powinien być dłuższy niż kilka sekund. Założenie to chatbot JSA w zupełności spełnia.

Kolejnym wymaganiem biznesowym było to, aby chatbot JSA był dostępny w równym stopniu dla użytkowników zalogowanych do systemu JSA, jak i dla niezalogowanych. Bot ma zapewnić profesjonalną i równą obsługę zarówno nowym użytkownikom systemu, którzy dopiero rozpoczynają swoją pracę z badaniem prac dyplomowych, oso-

bom doświadczonym w analizie raportów, ale także studentom i doktorantom, których działanie systemu w równym stopniu interesuje i zastanawia. Mając na względzie bezpieczeństwo danych wszystkich rozmówców chatbota postanowiliśmy w jego pierwszej wersji ograniczyć zakres pozyskiwanych danych. Do archiwum przeprowadzonych rozmów trafiają więc tylko i wyłącznie treści odbytych konwersacji. Dzięki takiemu rozwiązaniu każdy z rozmówców pozostaje całkowicie anonimowy. Założenie to wpłynęło deterministycznie na szereg innych wymagań, dotyczących m.in. wyglądu bota, sposobu jego obsługi, zapisywania danych, czy samej archiwizacji przeprowadzonej rozmowy.

Użytkownik przebywając na stronie Jednolitego Systemu Antyplagiatowego może w każdej chwili rozpocząć rozmowę z chatbotem bez względu na to, czy jest zalogowany w systemie JSA czy nie. Dzięki temu, że chatbot jest dostępny także dla użytkowników, którzy nie posiadają konta w systemie, czat JSA może pomagać szerszej grupie odbiorców. Po wprowadzeniu pierwszej wiadomości do chatbota rozpoczyna się proces analizy tekstu za pomocą algorytmów i mechaniznów opisanych dokładniej w części 2.5.2. Chatbot przeszukuje swoją bazę wiedzy, ocenia najtrafniejsze odpowiedzi i znalezione treści, a następnie wybiera i wyświetla końcową odpowiedź. Baza wiedzy chatbota obejmuje zasoby zgromadzone na stronach Centrum Pomocy JSA (w zakresie zarówno Bazy Wiedzy jak i Często zadawanych pytań) oraz innych materiałów przygotowanych specjalnie w celu jego nauki.

Po przeanalizowaniu profilu użytkowników korzystających z usług oferowanych przez Jednolity System Antyplagiatowy można było stworzyć profil psychologiczny chatbota. Fundamentalnym założeniem stało się to, aby chatbot wykazywał wysoki poziom kultury osobistej wskazujący na pełen profesjonalizm zapewnianej przez niego obsługi oraz podkreślający jego związek ze środowiskiem akademickim. Praktyka jednak pokazuje, iż zasadnym jest uczłowieczenie technologii chatbotowej poprzez zaprogramowanie obsługi tak zwanego *smalltalk*. Smalltalk zapewnia użytkownikowi lepsze wrażenia z prowadzonej konwersacji poprzez obsługę grzecznościowej wymiany zdań, zdolność polemiki na niskim poziomie abstrakcji, czy też naturalne reakcje na zapytania dotyczące jego natury. Chatbot JSA posiada również wbudowany zasób wiedzy potocznej dotyczącej m.in. astrofizyki (oparty na rozprawie Stephena Hawkinga zatytułowanej „Krótka historia czasu") oraz innych ogólnych zasobów wiedzy. Dzięki odpowiednim zasobom wiedzy z dziedzin dodatkowych można pogłębić profil psychologiczny chatbota oraz zapewnić użytkownikowi większy komfort obcowania z programem.

Podstawowym założeniem technologii chatbotowej jest jej dostosowanie do wymagań użytkowników oraz ciągły rozwój. W celu zaspokojenia tej potrzeby przewidziano proces oceniania przez użytkowników odbywanych rozmów. Proces ten jest dwustopniowy, co ma zapewnić dokładniejsze dane i lepszą weryfikację pracy chatbota. Pierwszym etapem jest możliwość oceny każdej otrzymanej od chatbota odpowiedzi. Ocena ta jest binarna i możliwe jest wskazanie czy wypowiedź oceniamy negatywnie czy pozytywnie. Drugi etap to ocena przeprowadzonej konwersacji jako całości. W tym przypadku przewidziana jest skala pięciostopniowa 1-5, przy czym 1 oznacza bardzo negatywną ocenę rozmowy, zaś 5 oznacza bardzo pozytywną ocenę. W tym etapie przewidziano także

możliwość załączenia krótkiego komentarza użytkownika wskazującego na najważniejsze powody wysokości wystawianej oceny.

2.5.2. Model chatbota

Przygotowanie analitycznego modelu chatbota wiąże się z szeregiem problemów, które należy rozwiązać. Poszczególne aspekty modelu powinny być uważnie dobierane przez wzgląd na wymagania biznesowe, możliwości obliczeniowo-pamięciowe, a także kontekst w jakim program ma być wykorzystywany.

STRATEGIA GENEROWANIA ODPOWIEDZI

Wybór modelu chatbota jest trudnym zadaniem. Wymaga poszukania kompromisu pomiędzy złożonością obliczeniowo-pamięciową, a jakością odpowiedzi. Warto wspomnieć, że sam proces generowania poprawnej odpowiedzi jest mocno uwarunkowany dziedziną, formą konwersacji czy długością spodziewanej interakcji. Wypowiedzi generowane przez program od podstaw wymagają nauczenia go składni języka bądź wgrania reguł logicznych, które pozwolą mu na zachowanie poprawności gramatycznej. Problem ten jest bardzo widoczny w języku polskim, ze względu na jego skomplikowanie. Metoda oparta o generowanie wypowiedzi od podstaw jest także bardziej skomplikowana od strony obliczeniowej. Jej zastosowanie wydaje się uzasadnione w przypadku użycia bota do bardzo długich rozmów na różne tematy, ponieważ pozwala na unikanie powtórzeń odpowiedzi, np. poprzez dodanie wartości losowej do danych wejściowych modelu, co bez wątpienia może poprawiać odczucia użytkownika względem „sztuczności" rozmowy.

Innym podejściem jest generowanie odpowiedzi na podstawie cytowań wybranych fragmentów z bazy wiedzy. Podejście to zapewnia poprawność gramatyczną na poziomie poprawności tekstów zawartych w bazie, jednak liczba możliwych odpowiedzi jest także ograniczona przez tę bazę. W konsekwencji długie rozmowy przy małej bazie wiedzy mogą wydawać się sztuczne dla użytkownika. Ze względu na charakter programu w złożonym systemie JSA, który opiera się o krótkie rozmowy polegające na trafnych, zwięzłych i merytorycznych odpowiedziach jednocześnie z małą liczbą kwestii niezwiązanych z tematyką JSA zastosowano drugi wariant oparty o cytowania.

MODEL OBLICZENIOWY

Kolejnym problemem związanym z działaniem chatbota był wybór fragmentów tekstu, które zawierały będą odpowiedzi na zadane pytanie. Tu znowu można wyróżnić kilka podejść. Pierwsze z nich opiera się o metody uczenia maszynowego, które na wejściu otrzymują dokument z bazy wiedzy oraz pytanie i na podstawie tych danych zwracają fragment dokumentu. Sam wybór cytowania jest poza kontrolą twórców, oparty o uczenie maszynowe w formie „czarnej skrzynki". Metoda ta wymaga dodatkowej na-

uki modelu, a co za tym idzie przygotowania dużego zbioru danych, dodatkowo analiza dużych baz wiedzy wiąże się z dużą złożonością obliczeniową.

Kolejnym podejściem jest przejrzenie całej bazy wiedzy i ocena każdego z poszczególnych fragmentów względem dychotomii: „dobra-zła odpowiedź". Oba rozwiązania mogą być oparte o nowoczesne metody uczenia głębokiego, jednak wiążą się z dużą złożonością obliczeniową, ponieważ każdorazowo wymagają analizy całej bazy. Zastosowana w systemie metoda opiera się na hybrydowym podejściu wykorzystania metod uczenia głębokiego z metodami opartymi o analityczne funkcje matematyczne mające na celu m.in. efektywne obliczenie odległości pomiędzy sformułowaniami opisanymi jako wektory liczbowe w wielowymiarowych przestrzeniach. Zastosowane rozwiązanie pozwala na ograniczenie liczby skomplikowanych obliczeń macierzowych do złożoności $O(n)$ względem aktualizacji bazy wiedzy, a nie jak we wcześniej wspomnianych metodach $O(n)$ względem każdego zapytania. Taka adaptacja rozwiązania uczenia głębokiego pozwala na kompromis pomiędzy nowoczesnymi rozwiązaniami przetwarzania języka naturalnego oraz wysoką wydajnością w środowisku biznesowym jednocześnie ograniczając koszty infrastruktury sprzętowej.

W drodze eksperymentu metoda oparta o model RoBERTa [13] wykazała najwyższą skuteczność, deklasując metody oparte o lematyzację oraz klasyczne zanurzenia *word2vec* [15]. RoBERTa to statystyczna reprezentacja języka naturalnego, do budowy której wykorzystano głęboką architekturę BERT (Bidirectional Encoder Representations from Transformers). To zbiór reguł pozwalających na wstępne wytrenowanie sieci neuronowej, zaprezentowany po raz pierwszy w 2018 roku przez Google'a, a później rozwinięty przez Facebooka i Uniwersytet Waszyngtoński i udostępniony pod nazwą RoBERTa. Dla języka polskiego istnieje model wyuczony na dużym korpusie dokumentów [7]. Zastosowanie go pozwala na tzw. transfer uczenia [24, 19].

WERYFIKACJA EKSPERYMENTALNA

O ile znajomość skuteczności metod opartych o RoBERTa jest dobrze przebadana dla języka angielskiego, to badania dla języka polskiego zostały sprawdzone w ograniczonym zakresie w określonych zadaniach w [7]. Zastosowanie modelu hybrydowego, wykorzystującego ekstrakcje cech w oparciu o RoBERTa jest specyficzne dla zastosowań biznesowych, dlatego aby potwierdzić skuteczność tego modelu wykonane zostały dodatkowe eksperymenty.

W pierwszym eksperymencie wykorzystany został zbiór 34 artykułów. Z każdego z artykułów wybrano losowo 5 paragrafów w taki sposób, aby każdy z nich składał się z 5 zdań. Tworząc w ten sposób zbiór 170 pięciozdaniowych paragrafów. W ramach eksperymentu zastosowano walidację krzyżową, w taki sposób, że z każdego z paragrafów wybrano 1 zdanie i przeniesiono je do zbioru testowego. Zadaniem modelu było wskazanie, z którego paragrafu pochodzi zdanie ze zbioru testowego. W ten sposób zadanie zostało sprowadzone do zadania klasyfikacji z 170 klasami. Zastosowanie metody walidacji krzyżowej pozwoliło na klasyfikację 850 zdań. W badaniu udział wzięły dwa

modele. Model oparty o zanurzenia *word2vec* oraz bibliotekę Morfologik[6] oraz model hybrydowy oparty o RoBERTa. Wyniki badania dokładności modeli pokazane w tabeli 2.5 wykazały, że model hybrydowy poradził sobie z zadaniem znacząco lepiej.

Tabela 2.5. Wyniki eksperymentu klasyfikacji paragrafów

Model	Dokładność
Morfologik + *word2vec*	0,694
Model hybrydowy RoBERTA	**0,815**

Kolejny eksperyment miał na celu potwierdzenie skuteczności modelu w zadaniach powiązanych z wymaganiami biznesowymi nałożonymi na wirtualnego asystenta. Badanie to opierało się o rzeczywiste dane znajdujące się w bazie wiedzy chatbota JSA. Na potrzeby eksperymentu przygotowany został zbiór pytań o różnym stopniu trudności. W zbiorze znalazły się parafrazy pytań FAQ oraz nowe pytania do treści artykułów Centrum Pomocy JSA. Pytania zostały podzielone na 5 poziomów w zależności od stopnia trudności. Poziomy zostały przedstawione za pomocą wartości liczbowych od 1 (treść jest dokładnym cytatem z bazy wiedzy) do 5 (treść zawiera synonimy, literówki, zwroty kolokwialne, słowa nie związane z wyszukiwaną pozycją z bazy wiedzy, przestawienia w szyku oraz błędy ortograficzne i gramatyczne).

Przykładowy podział ze względu na poziomy:

- Poziom 1: „Co to jest i skąd wezmę numer UID?”
- Poziom 2: „Co to jest UID i skąd wezmę jego numer?”
- Poziom 3: „Co to jest i skąd wezmę numer UID kierunku?”
- Poziom 4: „Co to jest i gdzie w systemie znajdę numer identyfikacyjny z systemu POL-on?”
- Poziom 5: „Gdzie w systemie znajdę numer identyfikacyjny z POL-onu?... :)”

W eksperymencie przebadano dokładność trzech modeli: *word2vec* + Morfologik, model hybrydowy oparty o RoBERTa oraz model hybrydowy oparty o RoBERTa i FastText [2]. Badanie oparte było o 1502 zapytania. Wyniki eksperymentów zostały przedstawione w tabelach 2.6, 2.7 oraz 2.8. Analizując je można zauważyć, że model oparty o *word2vec* (w2v) wypada zdecydowanie gorzej od dwóch pozostałych. Widoczna jest tendencja lepszego radzenia sobie z pytaniami o wyższym stopniu trudności przez model RoBERTa+FastText. Jedną z prawdopodobnych przyczyn jest wyższa odporność na zmiany szyku zdań.

[6] morfologik.blogspot.com (dostęp: 18.11 .2020)

Tabela 2.6. Wyniki eksperymentu dla wszystkich zapytań

Poziom	Liczba zapytań	Model (dokładność)		
		w2v+Morfologik	RoBERTa	RoBERTa+FastText
1	52	1,0	1,0	1,0
2	377	0,671	0,801	**0,803**
3	373	0,622	**0,823**	0,802
4	319	0,524	0,674	**0,702**
5	381	0,378	0,404	**0,498**

Tabela 2.7. Wyniki eksperymentu dla parafraz FAQ

Poziom	Liczba zapytań	Model (dokładność)		
		w2v+Morfologik	RoBERTa	RoBERTa+FastText
1	52	1,0	1,0	1,0
2	189	0,968	0,984	**0,994**
3	167	0,886	0,934	**0,982**
4	135	0,711	0,800	**0,911**
5	162	0,544	0,494	**0,618**

Tabela 2.8. Wyniki eksperymentu dla pytań dodatkowych

Poziom	Liczba zapytań	Model (dokładność)		
		w2v+Morfologik	RoBERTa	RoBERTa+FastText
2	188	0,372	**0,618**	0,611
3	206	0,408	**0,733**	0,655
4	184	0,386	**0,581**	0,549
5	219	0,255	0,338	**0,411**

ROZDZIAŁ 3

ARCHITEKTURA I TECHNOLOGIE

Daniel Karaś
Dawid Lubowiecki
Jakub Uszyński
Łukasz Podlodowski
Maciej Kowalski
Martyna Śpiewak
Paweł Szczepanowski

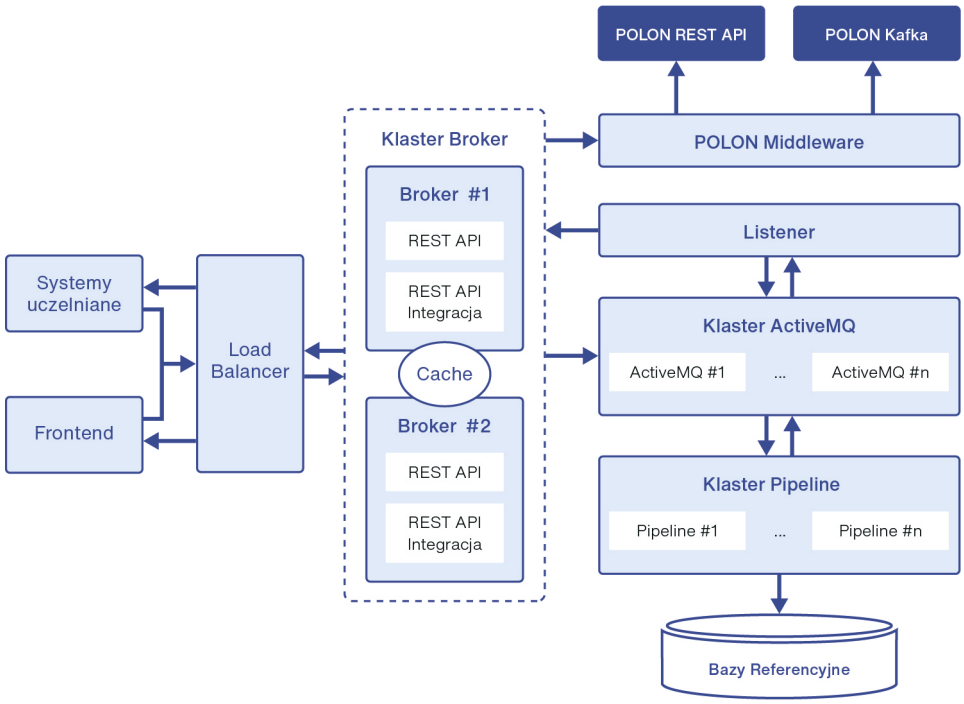
Poniższy rozdział skupia się na analizie systemu JSA z perspektywy jego architektury, zakresu funkcjonalnego jego komponentów oraz technologii jakie zostały użyte do jego zbudowania. Analiza ta bazuje na założeniu, że interesującym jest jakie rozwiązania zostały zastosowane by zbudować skalowalny system pracujący na terabajtach danych przetwarzając setki tysięcy prac dyplomowych. W kolejnych podrozdziałach poruszane są kwestie architektury, zakresu integracji, czy technologii użytych w określonych komponentach.

3.1. Architektura Systemu JSA

Ujęcie architektury systemu JSA w jednym, zgrabnym słowie wymagałoby użycia zwrotu „hybrydowa". Obecnie przyjęło się wyróżniać dwie główne filozofie tworzenia aplikacji webowych, a mianowicie architekturę monolityczną oraz opartą na mikroserwisach. Pierwsza z nich zakłada agregację wszystkich komponentów wymaganych do działania aplikacji w jedną, niezależną, niepodzielną całość. Druga zaś wyodrębnia każdą, możliwie niezależną, składową do oddzielnych, luźno powiązanych ze sobą serwisów, tak by każdy realizował stosunkowo małą część funkcjonalności całej aplikacji. Można by długo dywagować nad zaletami i wadami każdego z podejść. Z jednej strony, gdy w przypadku mikroserwisów wyniknie konieczność komunikacji między poszczególnymi serwisami, wytworzy to narzut czasowy wynikający z konieczności użycia do tejże komunikacji dodatkowego medium pod postacią szyny pokroju ActiveMQ czy RabbitMQ bądź wywołań REST API. Może to prowadzić do odczuwalnych dla użytkownika opóźnień w czasie odpowiedzi, którego to problemu nie przejawia monolit z uwagi na to, że wszystkie serwisy współistnieją ze sobą lokalnie. Z drugiej strony, jeśli rozważyć zagadnienia związane z utrzymaniem i rozwojem aplikacji, mikroserwisy pozwalają na dużo bardziej precyzyjne skalowanie horyzontalne. Możliwe staje się wyodrębnienie

konkretnej składowej, która narażona jest na największe obciążenia i wytworzenia jej dodatkowych instancji, pomiędzy którymi obłożenie rozkładać będzie odpowiedni load balancer. W przypadku monolitu wymagałoby to utworzenia klonów całej aplikacji, uwzględniając komponenty, które tego skalowania nie wymagają, co z kolei przekłada się na nieefektywne wykorzystanie zasobów. Kolejnym, wartym odnotowania i bardzo polaryzującym aspektem jest transakcyjność. Z uwagi na lokalny charakter pod-serwisów wchodzących w skład monolitu, transakcje w naturalny i prosty sposób mogą okalać swoim zakresem operacje pochodzące z kilku pod-serwisów. W przypadku alternatywnej („konkurencyjnej") architektury wymagałoby to wprowadzenia złożonego i podatnego na błędy rozwiązania z zakresu rozproszonych transakcji, jak chociażby two-phase-commit. Nie bez powodu sformułowanie „konkurencyjny" zostało ujęte w cudzysłów. Należy w tym miejscu podkreślić, że nie można mówić jednoznacznie o wyższości jednej architektury nad drugą. Jak pokazują wyżej przytoczone przykłady, wybór konkretnej architektury powinien być podyktowany przede wszystkim specyfiką realizowanego projektu, a nie tylko osobistymi przekonaniami czy modą. Dlatego też JSA realizuje model hybrydowy.

3.1.1. Model hybrydowy realizowany przez JSA

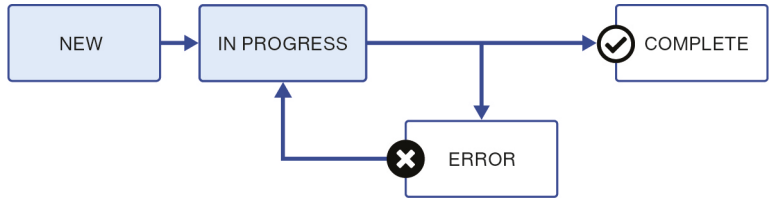


Ilustracja 3.1. Schemat architektury JSA

Analiza powyższego schematu architektury JSA, już na pierwszy rzut oka, pozwala stwierdzić, że nie mamy w tym przypadku do czynienia ani z klasyczną architekturą mikroserwisową, ani monolityczną. Sam fakt istnienia kilku niezależnie skalowalnych komponentów wyklucza przypadek realizacji monolitu sensu stricto. Na próżno szukać na ilustracji odrębnych serwisów obsługujących zarządzanie użytkownikami, zarządzanie badaniami czy odpowiedzialnych za autentykację – w tym wypadku broker dokonuje enkapsulacji wszystkich wymienionych funkcjonalności. Podobna zależność zachodzi w przypadku udostępnianego przez brokera API integracyjnego – nierzadko w przypadku mikroserwisów można natknąć się na dodatkowy byt tzw. *API gateway*. Pośredniczy on w ruchu pomiędzy klientami a serwisami, równocześnie rozpinając nad nimi dodatkową warstwę abstrakcji. Wygodnym jest myślenie o API gateway jak o realizacji programistycznego wzorca projektowego „fasada". W przypadku brokera API gateway jest w niego „wbudowany".

3.1.2. Cykl życia dokumentu

Przed przystąpieniem do dalszej dyskusji odnośnie odpowiedzialności poszczególnych komponentów systemu, należy przyjrzeć się bliżej cyklowi życia badanego dokumentu, gdyż jest on ściśle związany z przyjętym modelem przepływu informacji przez system.



Ilustracja 3.2. Cykl życia dokumentu

Cykl życia dokumentu opisywany jest następującymi stanami:

- New – w obecnej wersji systemu, status ten ma jedynie znaczenie „historyczne". Początkowo, broker umieszczał w bazie danych dokument ze statusem new tak, by zasygnalizować listenerowi, że pojawiła się nowa jednostka pracy do przekazania poprzez ActiveMQ do *pipeline'u*. Listener podejmował wtedy dany dokument, zmieniał jego status na „in progress", opakowywał treść dokumentu oraz jego metadane w odpowiednio sformatowaną wiadomość i umieszczał w kolejce ActiveMQ.
- In progress – obecnie można go uznać za stan „wyjściowy".
- Complete – badanie zakończone sukcesem.
- Error – wystąpił błąd podczas przetwarzania dokumentu.

Konkretne mechanizmy sterujące przejściem pomiędzy wymienionymi stanami zostaną opisane w dalszej części rozdziału. Ciekawostkę natury architektonicznej stanowi fakt, że od strony implementacyjnej statusy dokumentów realizują, bardzo popularny

w architekturze opartej na mikroserwisach, wzorzec *event sourcing*. Oznacza to, że status dokumentu nie jest jego bezpośrednią właściwością, a raczej jest wypadkową statusów do niego przypisanych. Nie tylko pozwala to na śledzenie zmian w statusie dokumentu, ale potencjalnie pozwala unikać wielu problemów związanych z równoległą modyfikacją konkretnej encji bazodanowej przez kilka procesów – w związku z tym, że każdy proces zapisze w tym wypadku własną encję korespondującą ze statusem, ewentualne konflikty można rozwiązać na poziomie wnioskowania o ostatecznym statusie (np. odrzucić przejścia pomiędzy stanami, które nie powinny mieć miejsca). Event sourcing może również pozytywnie wpływać na wydajność – z uwagi na swoją naturę, redukuje on do minimum konieczność korzystania z blokad czy transakcji na wysokości bazy danych. Pozwala on również na sporą dozę elastyczności – struktura bazodanowa w tym przypadku w mniejszym stopniu „narzuca" postać danych, które można zagregować i pokazać użytkownikowi. Przykładowo: jeśli przyjąć, że data i godzina ostatniej zmiany statusu zapisywana jest jako kolumna typu timestamp w tabeli modelującej dokument, to gdyby użytkownik poczuł potrzebę uzyskania informacji o dacie i godzinie pierwszego statusu, wymagałoby to utworzenia dodatkowej kolumny przechowującej tę właśnie informację. Nie dałoby się odzyskać tejże daty dla dokumentów dodanych do bazy przed wprowadzeniem opisanej zmiany. Problem ten nie występowałby modelując dokument i jego statusy jako relację jeden-wiele. Jest to oczywiście jedynie jeden przykład z szerokiej gamy problemów prezentowania danych użytkownikowi, które mogą zostać elegancko rozwiązane przy użyciu event sourcingu.

3.1.3. Cykl życia badania i próby

W JSA badanie wyrażone jest z użyciem hierarchicznej struktury, gdzie każde badanie posiada 0..N prób, z kolei każda próba posiada 1..M dokumentów, z czego każdy z nich posiada 1..K stanów. W równie hierarchiczny sposób przebiega wnioskowanie o ostatecznym statusie danego badania. Aby ustalić status próby, statusy wszystkich dokumentów podlegających próbie są sortowane – nie tylko z uwagi na kolejność dodania, ale również ich priorytet, a następnie wybierany jest pierwszy z tak posortowanej listy statusów – staje się on wypadkowym statusem próby. Klaryfikacji może wymagać koncepcja „priorytetu". Rozpatrzmy to na przykładzie – jeśli dana próba posiada 2 dokumenty, z czego jeden został przed chwilą pomyślnie przebadany, natomiast drugi tkwi w statusie error od pewnego czasu, to wypadkowym statusem próby nadal będzie *error*. Nie można mówić o pozytywnym zakończeniu próby jeśli badanie któregoś z pozostałych dokumentów jeszcze się nie zakończyło. W przypadku badania mechanizm jest prostszy – dziedziczony jest bezpośrednio status najnowszej próby. Wyjątek od opisanej metodyki stanowi specjalny status *Registered* oraz *Accepted*. Pierwszy z nich przewidziany jest dla badania, które zostało zarejestrowane bez żadnych prób, natomiast drugi wynika bezpośrednio z interakcji użytkownika z systemem (użytkownik zaakceptował próbę). Jednocześnie – dowolna zaakceptowana próba wymusza na całym badaniu status accepted.

3.1.4. Pipeline

Już na bardzo wczesnych etapach realizacji projektu stało się oczywiste, że najbardziej złożonym obliczeniowo, wymagającym pod kątem zasobów i w ogólnym rozumieniu – krytycznym komponentem będzie tzw. *pipeline*, czyli serwis, którego zadaniem jest przyjęcie na swoim wejściu treści badanej pracy i zwrócenie fragmentów tejże pracy, wraz ze źródłami, w przypadku których zachodzi podejrzenie zajścia plagiatu.

Dlatego też, aby wyjść naprzeciw oczekiwaniom związanym z niezawodnością i czasem odpowiedzi, kluczowym stało się utworzenie rozwiązania, w którym możliwym będzie zręczne skalowanie horyzontalne tychże komponentów oraz zapewnienie luźnego powiązania z resztą systemu tak, by awaria jednego pipeline'u nie paraliżowała pracy całego systemu. Cele te udało się osiągnąć, stawiając w centralnym punkcie systemu szynę ActiveMQ, która pośredniczy w komunikacji pomiędzy głównymi składowymi systemu. Oznacza to, że w przypadku konieczności zapewnienia większej przepustowości w kontekście liczby badanych prac, wystarczy powołać do życia nową instancję pipeline'u i wskazać jej adres odpowiedniej szyny, by instancja ta mogła natychmiast przejąć część obciążenia – bez konieczności modyfikacji czy ponownego wdrażania reszty systemu.

Analogicznie wygląda sytuacja odłączenia pipeline'u od szyny – na skutek awarii czy konieczności alokowania zasobów sprzętowych w inny sposób. Może to naturalnie rodzić wątpliwości – czy ActiveMQ nie jest w tym przypadku tzw. single point of failure? W istocie, potencjalna całkowita awaria jedynej instancji ActiveMQ oznaczałaby częściowy paraliż JSA – konkretnie funkcjonalności związanych z przyjmowaniem nowych badań oraz ładowaniem wyników nowo przeprowadzonych badań. Ryzyko takiego zajścia jest jednak znikome, gdyż system opiera swoje działanie na klastrze ActiveMQ w trybie master-slave, co oznacza, że w przypadku awarii jednej szyny działającej w trybie master, z pozostałych instancji działających w trybie slave zostanie wyłoniony nowy „master". Jednocześnie, aby zapewnić spójność stanu pomiędzy poszczególnymi instancjami, przechowywane przez nie wiadomości zapisywane są we współdzielonym, rozproszonym systemie plików (GlusterFS). Zatem nowy „master" jest w stanie rozpocząć pracę dokładnie w tym samym punkcie, w którym skończył ją były „master".

Nazwa „pipeline" jest nieprzypadkowa. Obserwując dwa końce tego „rurociągu", na jednym z nich można dostrzec badaną pracę, natomiast u jego wyjścia fragmenty prac źródłowych, które mogą wskazywać na plagiat. Nie wyczerpuje to jednak tematu i w tym ujęciu pipeline jawi się bardziej jak czarna skrzynka, niż faktyczny rurociąg. W rzeczywistości, badana praca „przepływa" przez szereg rozwidleń – takich jak analiza typu *extrinsic* (wyszukiwanie kandydatów w korpusie prac źródłowych, zasilanym przez indeksy rozpięte nad bazami ORPPD, NEKST, itd.), analiza *intrinsic* (typowanie stylometryczne plagiatu nie korzystając z zewnętrznego korpusu) oraz *text alignment* (wyznaczanie konkretnych fragmentów w podejrzanej pracy względem wytypowanych kandydatów w kroku extrinsic). Wymienione składowe zostaną szerzej opisane w dalszej części rozdziału.

3.1.5. Listener

W JSA można wyróżnić dwie naczelne warstwy – warstwę obliczeniową (pipeline) oraz warstwę biznesową (broker). Jeśli przyjąć taki podział, to komponent Listener można by określić mianem *middleware* (oprogramowania pośredniczącego). Jego zakres odpowiedzialności zmieniał się wraz z rozwojem aplikacji, jednak główna idea pozostała niezmieniona – odbieranie wyników z kolekcji pipelinów, ich transformacja i zapis w formie wymaganej przez brokera.

Jednym z ciekawszych problemów, które rozwiązuje listener jest takie przetwarzanie fragmentów tekstów źródłowych, by dany fragment tekstu podejrzanego o plagiat był okupowany jedynie przez jeden pasaż źródłowy.

Algorytm przetwarzania fragmentów tekstu jest następujący: w pierwszej kolejności zostają odrzucone fragmenty, które nie przekraczają pewnej, zadanej przez użytkownika, wartości progowej współczynnika podobieństwa. Wszystkie pozostałe źródłowe fragmenty są sortowane według pewnej przyjętej metryki – przykładowo priorytetyzacja z uwagi na rodzaj źródła (ORPPD, internet, akty prawne), długość fragmentu czy wysokość współczynnika podobieństwa wykazanego przez text alignment. Krok ten jest w pełni generyczny i konfigurowalny, co pozwala na dużą dozę elastyczności i wyjście naprzeciw wymaganiom użytkowników. Przyjęta metryka zmieniała się na przestrzeni lat – na chwilę obecną brany pod uwagę jest jedynie wskaźnik podobieństwa. Następnie, zgodnie z ustaloną kolejnością, są one nanoszone na przestrzeń tekstu pracy podejrzanej o plagiat.

Trywialny przypadek występuje, gdy umieszczany fragment nie koliduje z innym, już naniesionym fragmentem – wtedy używana jest pełna długość źródła. Pozostałe cztery przypadki zależą od relatywnego położenia rozpatrywanych fragmentów w tekście – „kandydat" może przecinać jeden lub więcej już naniesionych fragmentów (nieformalnie) „od lewej", „od prawej", być zawieranym lub zawierać inny pasaż. W pierwszych dwóch przypadkach, dokonana zostanie operacja na pozycji początku lub końca „kandydata" – zostanie ona ustalona tak, by „stykać" się z już naniesionym fragmentem (dla ułatwienia dalszej dyskusji – przyjmijmy, że pasaże te będą korespondować kolejno z nazwami suspnew i suspold). W przypadku gdy suspnew jest w pełni zawarty przez pewien suspold, nie istnieje żadna potencjalna pozycja początku czy końca suspnew, która pozwoliłaby na bezkolizyjne umieszczenie suspnew w zbiorze suspold, więc taki suspnew zostaje odrzucony. Zdecydowanie ciekawszy jest przypadek odwrotny, tj. gdy suspnew zawiera pewien suspold. Oznacza to, że proste operacje na początku czy końcu suspnew nie wchodzą w grę, gdyż mogą prowadzić do potencjalnie dużych strat informacji – przykładowo – gdyby stosunkowo krótki suspold występował pośrodku długiego suspnew, wymagałoby to „zrównania" suspnew z granicami suspold, co prowadziłoby do utraty blisko połowy informacji. Dlatego też, suspnew dzielony jest wtedy na dwa pochodne fragmenty, z czego każdy z nich z osobna przechodzi proces ponownego „przycinania". Należy przy tym mieć na uwadze, że takie operacje mogą prowadzić do powstania nowych pasaży irrelewantnych dla użytkownika końcowego (z uwagi na ich długość).

Dlatego też, przed umieszczeniem danego fragmentu w zbiorze suspold, każdy z nich badany jest pod kątem długości w znakach oraz liczby zawartych w nim zdań. Tak uzyskany zbiór suspold zapisywany jest do bazy danych, skąd może później zostać pobrany przez brokera w celu utworzenia raportu pdf (proces opisany w oddzielnym podrozdziale) czy umożliwienia pracy promotorowi w zakresie odrzucania wskazań, których nie uzna on za plagiat. W ostatnim kroku, listener zmienia status dokumentu na „complete" i informuje o tym fakcie brokera, by ten mógł podjąć dalsze czynności takie, jak wysyłanie powiadomień o zakończeniu badania drogą mailową czy przesłanie pliku z raportem na wskazany przez uczelnię adres końcówki REST. Niestety, na chwilę obecną, rzeczona komunikacja pomiędzy brokerem i listenerem nadal odbywa się za pośrednictwem wywołań REST, a nie za pośrednictwem ActiveMQ, co pozwoliłoby na wytworzenie lepszej warstwy abstrakcji pomiędzy wymienionymi komponentami.

Do zadań listenera należy również zarządzanie cyklem życia dokumentu (a zatem również całego badania). Przejawia się to w dwojaki sposób – pasywny i aktywny. O trybie pasywnym można mówić w przypadku reakcji listenera na wynik otrzymany za pośrednictwem ActiveMQ z pipeline'u. Jeśli sprawdzanie dokumentu zakończyło się sukcesem – listener ustawia status dokumentu na complete. Może się jednak zdarzyć, że z różnych powodów (jak choćby awaria jednej z baz referencyjnych), pipeline poinformuje o błędzie – wtedy listener ustawia status dokumentu na „error". Ponadto, listener cyklicznie sprawdza bazę danych pod kątem dokumentów, które przebywają w konkretnym stanie zbyt długo. Każdy ze stanów ma zdefiniowaną pewną wartość progową czasu, po którym listener reaguje, wysyłając dokument do ponownego badania (przy jednoczesnym ustawieniu statusu dokumentu z powrotem na: „in progress"). Przykładowo, jeśli dokument przebywa w statusie „in progress" dłużej niż parę godzin, to można założyć, że w wyniku awarii, pipeline który skonsumował tę konkretną jednostkę pracy, nie był w stanie dokończyć sprawdzenia i umieścić wyniku zwrotnego na ActiveMQ.

3.1.6. Broker

Broker stanowi najbardziej „monolityczny" element całej architektury JSA. Jest on konglomeratem kilku serwisów odpowiadających za konkretne warstwy funkcjonalności biznesowej, takich jak zarządzanie użytkownikami, zarządzanie badaniami, zarządzanie uprawnieniami, aplikowanie uprawnień do zwracanych treści czy autentykacja. Funkcjonalności te udostępnia za pośrednictwem REST API dla aplikacji frontendowej oraz systemom uczelnianym w ramach integracji. Podobnie jak w przypadku większości elementów systemu, również broker występuje w kilku instancjach, tak by zapewnić jak największą dostępność i przepustowość. Aby dodatkowo przyśpieszyć czas przetwarzania żądań pomiędzy instancjami brokera zostało wprowadzone dodatkowe, współdzielone, rozproszone cache – mechanizm ten został opisany szerzej w jednym z kolejnych podrozdziałów.

3.2. Technologie wspomagające

3.2.1. ActiveMQ

ActiveMQ (AMQ) stanowi technologię, która zapewnia szynę danych – komunikację pomiędzy brokerami, listenerem i jednostkami przeliczeniowymi (pipeline). W wersji produkcyjnej za pomocą AMQ został utworzony klaster (3 węzły), z których automatycznie wybierany jest węzeł nadrzędny, dwa pozostałe węzły pełnią funkcję zastępczą. W przypadku awarii głównego węzła system przełącza funkcjonalność kolejki na kolejny wolny węzeł bez utraty danych. Zostało to zrealizowane poprzez zamontowanie rozproszonego systemu plików między maszynami, które utrzymują instancje AMQ oraz zapisywanie stanu kolejek w tym zasobie.

3.2.2. Wildfly

Podstawą każdego węzła obliczeniowego (pipeline) jest serwer aplikacyjny Wildfly. Na każdym serwerze aplikacji umieszczona jest wielowątkowa aplikacja do odpytywania indeksów zawierających dane z internetu, ORPPD i Wikipedii, aktów prawnych etc., pobierająca odpowiednie treści dokumentów oraz licząca przecięcia pomiędzy dokumentami. Każda z tych aplikacji jest połączona z kolejką AMQ, co pozwala na asynchroniczne przetwarzanie danych – zadanie porównania dokumentu najpierw trafia na kolejkę, następnie kolejka informuje wolny pipeline o możliwości pobrania zadania.

3.2.3. Load Balancing

Aby zapewnić niezawodność w okresach wzmożonego użycia JSA zdecydowano się zwielokrotnić instancje brokera. Docelowo na środowisku produkcyjnym uruchomione są trzy takie instancje. Aby zapewnić transparentność tego rozwiązania dla użytkowników, wszystkie trzy instancje widnieją pod adresem jsa-broker.opi.org.pl, którym zarządza instancja NGINX[7]. Gdy do serwera przychodzi zapytanie, serwer NGINX na podstawie zdefiniowanych reguł, kieruje te zapytania do odpowiednich instancji tak, aby wyrównać ich obłożenie.

3.2.4. Postgres

Wszystkie dane zapisane są na dyskach serwerów na bazach Postgresql. JSA posiada kilka baz danych, między innymi do składowania zawartości tekstowej stron internetowych (NEKST, Wikipedie, baza danych aktów prawnych, orzeczeń), zawartości

[7] nginx.com (dostęp: 18.11.2020)

tekstowej prac dyplomowych (ORPPD), zawartości tekstowej publikacji Open Access oraz danych powstałych w samym systemie (np. użytkownicy, zlecenia badania prac, wyniki badania).

3.2.5. ELK Stack

ELK Stack[8] to akronim trzech projektów open source: Elasticsearch, Logstash i Kibana. Aby zapewnić skalowalne przeszukiwanie danych w poszukiwaniu plagiatu, użyty został rozproszony silnik wyszukiwania pełnotekstowego Elasticsearch, na którym wykonywane są zapytania odszukujące fragmenty tekstów z baz referencyjnych. Struktura klastra została opisana w jednym z poprzednich podrozdziałów. Dodatkowo zostały w projekcie uruchomione dedykowane klastry Elasticsearch służące do przechowywania logów aplikacji (logi są zbieranie z wykorzystaniem narzędzia Logstash). W celu monitorowania stanu aplikacji – logi trafiają do klastra Elasticsearch, następnie, przez specjalny interfejs zapewniany przez Kibanę, można te logi przeglądać i monitorować stan całej aplikacji.

3.2.6. Wkhtmltox

Aby zapewnić możliwość drukowania raportów, wykorzystano oprogramowanie wkhtmltox, dzięki któremu z dokumentu HTML możemy wygenerować plik PDF. Broker posiada wbudowane szablony raportów, które następnie są wypełniane danymi badania.

Kod 3.1. Przykład pliku HTML raportu ze znacznikami, gdzie umieszczane są dane

```
<!DOCTYPE html>
<html>
<body>
<div class="header">
<table class="width100">
    <tr>
        <td class="width33 lefted">
            <img src="" style="height: 49px;"/>
        </td>
        <td class="width33 centered">
                <span
                     class="{item.2:DEMO_DISPLAY}">{item.s_TRANS:HEADER_DEMO}</span>
        </td>
        <td class="width33 righted" style="text-transform:
             uppercase;font-size:15px;">
            {item.0:VALUE1}
        </td>
    </tr>
    <tr>
        <td colspan="3" style="height:28px;"></td>
    </tr>
```

[8] elastic.co/what-is/elk-stack (dostęp: 18.11.2020)

```
        <tr>
            <td colspan="3" class="border-under">
            </td>
        </tr>
    </table>
    </div>
    </body>
    </html>
```

Tak przygotowany dokument jest zapisywany na serwerze JSA, następnie generowana jest jego reprezentacja PDF i przesyłana do klienta zadania.

3.3. Integracja z systemami zewnętrznymi

JSA do swojego poprawnego działania potrzebuje danych. Poniżej opisaliśmy kluczowe systemy zewnętrzne, z którymi na różny sposób zintegrowane jest JSA.

3.3.1. POL-on

Jednym z kluczowych elementów Jednolitego Systemu Antyplagiatowego jest spójność wykorzystanych danych z POL-onem – zintegrowana sieć POL-on to obszerne repozytorium danych o polskiej nauce i szkolnictwie wyższym. Zasadniczymi informacjami z POL-onu wykorzystywanymi przez JSA są:

- lista instytucji (w JSA ograniczamy się do uczelni publicznych, niepublicznych i kościelnych oraz instytucji naukowych);
- struktura instytucji wraz z jednostkami podrzędnymi;
- lista kierunków z uruchomieniami;
- lista studentów zarejestrowanych na dane uruchomienie kierunku;
- lista pracowników w instytucji.

Dane te są wykorzystywane w wielu miejscach systemu, m.in. przy tworzeniu nowego użytkownika (korzystamy z listy pracowników dostępnej w instytucji, nadajemy uprawnienia na instytucje lub jednostki podrzędne), czy przy dodawaniu bądź edycji badania (metadane w metryce w większości pochodzą z POL-onu – instytucje, jednostki, uruchomienia kierunków, studenci, promotorzy/recenzenci).

Jak widzimy z racji, że dane te są wykorzystywane w kluczowych miejscach JSA, ważnym było zapewnienie stałej, szybkiej i bezpiecznej integracji między naszymi systemami. W tym celu stworzyliśmy osobny serwis, który służy do integracji danych pomiędzy JSA a POL-onem. Początkowo integracja między tymi dwoma systemami odbywała się przy użyciu REST API, jednak takie podejście było mało efektywne, prowadziło do częstych przerw w dostępie do danych i nie zapewniało natychmiastowego uaktualniania danych w JSA. Szukając remedium na powyższe problemy zdecydowaliśmy się na integrację danych przy pomocy Apache Kafki – uzyskując szybki i bezpieczny przepływ danych między POL-onem a JSA. Schemat komunikacji jest bardzo prosty — nadawca

– POL-on – (ang. *producer*) przesyła komunikaty do Kafki, natomiast odbiorca – JSA – przy użyciu POLON Serwis (ang. *consumer*) pobiera wiadomości ze strumienia publikowanego przez Kafkę.

Publikowane komunikaty zostały pogrupowane w następujące tematy (ang. *topics*):

- instytucje i jednostki podrzędne;
- kierunki studiów i ich uruchomienia;
- pracownicy;
- studenci;
- obronione prace dyplomowe z Ogólnopolskiego Repozytorium Pisemnych Prac Dyplomowych (prace licencjackie, inżynierskie i magisterskie);
- obronione prace doktorskie z bazy postępowań awansowych.

Pierwsze cztery tematy są wykorzystywane do zapewnienia spójności danych między JSA a POL-onem, jak wyżej wspomnieliśmy, podczas zakładania lub edycji metryki badania, czy powiązania pracy z jej promotorem. Natomiast prace z ORPPD oraz bazy postępowań awansowych są wykorzystywane do budowy indeksów.

3.3.2. NEKST

Przez NEKST określamy dane z internetu (głównie polskiej części internetu) dostarczane przez zewnętrznego dostawcę IPI PAN. Dane te obejmują:

- polskie strony internetowe zbierane przez farmę robotów pracujących w IPI PAN;
- zestaw najpopularniejszych w Europie Wikipedii, w tym ujęta jest polska Wikipedia, oraz 9 innych;
- baza polskich orzeczeń sądowych;
- baza polskich aktów prawnych (ISAP) redagowana przez zespół specjalistów w Kancelarii Sejmu.

Dane są dostarczane w postaci oryginalnych stron, jak i gotowych wyekstraktowanych zawartości stron, które spełniają warunki:

- nie mogą zawierać treści pornograficznych;
- nie są związanymi z e-commerce, forami internetowymi;
- nie są klonami innych dokumentów;
- mają powyżej 500 znaków ciągłego czystego tekstu (tekstu niezabrudzonego znacznikami xml, xhtml, html etc.).

Przykładowe statystyki danych pozyskiwanych w ramach źródła NEKST:

- Liczba wszystkich dokumentów internetowych NEKST: 2.034.022.815.
- Liczba odfiltrowanych dokumentów z kategorii sklepy e-commerce: 2.870.849.
- Liczba odfiltrowanych dokumentów z kategorii fora internetowe: 31.681.627.
- Liczba odfiltrowanych dokumentów z kategorii erotyczne: 490.300.

- Liczba odfiltrowanych dokumentów niespełniających kryterium minimalnej zawartości czystego tekstu: 223.578.380.
- Liczba odfiltrowanych dokumentów oznaczonych jako klony innych dokumentów: 88.092.010.
- Ostateczna liczba dokumentów zakwalifikowanych do zrzutu danych dla JSA to 459 milionów. W tym dominują dokumenty formatu HTML ponad 450 milionów, na kolejnych miejscach pod względem liczności są PDF około 8 milionów i DOC około 1 miliona.

Całkowita objętość zrzuconych danych po skompresowaniu przekracza 10 TB.

Każda ze stron ładowana jest do referencyjnej bazy danych, która następnie służy do budowania indeksu przeszukiwania pełnotekstowego (Elasticsearch), a następnie użyta jest do wyszukiwania pełnych treści dokumentów, które wskazał algorytm JSA.

3.3.3. Bing

System JSA jest zintegrowany z międzynarodową wyszukiwarką internetową Bing. Jest to wyszukiwarka internetowa stworzona przez amerykańską firmę Microsoft, początkowo jako część portalu MSN. Integracja między JSA a Bingiem jest wykonana za pośrednictwem Chmury Azure, a dokładniej jej usług dostępnych w modelu SaaS (*Software As A Service*).

Chmura Azure to skalowalna platforma dostępna w internecie pozwalająca JSA przy pomocy dedykowanych interfejsów API korzystać m.in. z takich funkcjonalności jak: wyszukiwanie treści w internecie na podstawie słów kluczowych, analizy tekstu, w tym w szczególności ekstrakcji słów kluczowych oraz tłumaczenia maszynowego.

Chmurowa usługa wyszukiwania Bing umożliwia za pośrednictwem dedykowanego API dodawanie do naszych systemów inteligentnego wyszukiwania, opartego na zasobach miliardów stron internetowych zaindeksowanych przez wyszukiwarkę Bing.

3.3.4. OpenAccess

DOAJ (Directory of Open Access Journal) jest internetowo dostępną bazą danych dokumentów i artykułów naukowych w wielu językach, dostępnych na licencji Open Access. Bazę danych można ściągnąć ze strony doaj.org/public-data-dump/article (dostęp: 18.11.2020). Baza danych zawiera metadane na temat zawartych na stronie DOAJ artykułów, takie jak tytuł, autorzy, słowa kluczowe, abstrakty i link do samej publikacji. System JSA ściąga te publikacje, następnie ładuje je do referencyjnej bazy danych w celu indeksowania oraz przeszukiwania.

3.3.5. ORPPD

Ogólnopolskie Repozytorium Pisemnych Prac Dyplomowych (ORPPD) jest elementem Systemu Informacji o Szkolnictwie Wyższym POL-on. ORPPD gromadzi uprawnionym podmiotom pełne teksty prac dyplomowych obronionych na polskich uczelniach. Aktualnie ORPPD posiada ponad 3,3 mln prac dyplomowych. System JSA jest zintegrowany z zewnętrznym systemem do przechowywania pisemnych prac dyplomowych ORPPD, o czym napisaliśmy w sekcji dotyczącej integracji z POL-onem. W momencie, kiedy któraś uczelnia zapisze w tym systemie pracę, system JSA jest o tym fakcie informowany, przez co może taką pracę pobrać do swoich danych referencyjnych. Na podstawie tych danych budowany jest indeks wyszukiwania.

Indeksy przeszukujące JSA budowane są na żądanie.

Kod 3.2. Skład indeksów wykorzystywanych przez JSA w oparciu o Elasticsearch na dzień 10.11.2020

```
health  status  index       docs.count      store.size
green   open    ruwiki        27341907          45.6gb
green   open    nlwiki         9378060          11.4gb
green   open    eswiki        18297638          28.5gb
green   open    sejmisap      10667924           7.9gb
green   open    internet    7464948851           8.7tb
green   open    ukwiki        13796880            24gb
green   open    doaj         359263769         316.9gb
green   open    itwiki        17343990          27.7gb
green   open    dewiki        31744565          51.9gb
green   open    bcosa         46475542            44gb
green   open    enwiki       100178460         154.7gb
green   open    svwiki        46447208          57.1gb
green   open    frwiki        24113623            44gb
green   open    plwiki        12967101          21.3gb
green   open    orppd       2545167425           1.7tb
```

3.4. Aplikacja frontend

Aplikację przygotowano korzystając z najnowszej wersji frameworku Angular[9] – jest to otwarty framework i platforma do tworzenia SPA (ang. *single page application*), napisany w języku TypeScript i wspierany oraz rozwijany przez Google. O wyborze zaważyła jego duża popularność oraz fakt, że jest wspierana przez dużą firmę o stabilnej pozycji na rynku, czyli Google. Ta decyzja zagwarantowała stosunkowo dużą dostępność doświadczonych pracowników. Nie bez znaczenia był też fakt, że Angular zawiera zestaw komponentów kluczowych dla zbudowania złożonej aplikacji takich jak Router, Interceptory, klient http, moduł formularzy, czy mechanizm *Dependency Injection*. Wyeliminowało to w dużym stopniu konieczność implementacji własnych rozwiązań specyficznych dla pojedynczej aplikacji oraz zapewniło wysoki poziom unifikacji aplikacji stworzonych

[9] angular.io (dostęp: 18.11.2020)

w tym frameworku co dodatkowo ułatwiło wdrożenie nowych pracowników do projektu. Ponadto skorzystano z zestawu komponentów „Angular Material". O wyborze tej biblioteki zaważył fakt, że jest tworzona przez tą samą firmę, która stworzyła framework Angular, czyli firma Google. Daje to gwarancję, że kolejne aktualizacje zarówno frameworku Angular jak i ich bibliotek nie będą generowały błędów. Ponadto twórcy biblioteki deklarują wsparcie dla osób niepełnosprawnych.

3.4.1. Komunikacja z backendem

KONTRAKT

Korzystając z doświadczenia programistów odpowiedzialnych za stworzenie struktury projektu ustalono, że należy przygotować zestaw narzędzi, który pomoże uniknąć problemów z synchronizacją zmian interfejsów obiektów DTO (*Data Transfer Object*), które służą do komunikacji pomiędzy różnymi źródłami danych. Stworzono osobne repozytorium, w którym znalazły się pliki opisujące komunikację REST zgodnie ze specyfikacją OpenApi. Dzięki takiemu podejściu definicja kontraktu naturalnie stała się dokumentacją komunikacji z backendem. Następnie przygotowano skrypty, służące do generowania interfejsów w językach Java oraz TypeScript na podstawie uprzednio zdefiniowanego kontraktu. Wyeliminowało to rodzinę błędów związanych z komunikacją pomiędzy backendem, a aplikacją frontendową.

WARSTWA IMITUJĄCA BACKEND

Kolejną korzyścią wynikającą z możliwości automatycznego generowania interfejsów TypeScript, była możliwość przygotowania warstwy, która byłaby w stanie imitować prawdziwą implementację backendu. Było to ważne, ze względu na umożliwienie rozpoczęcia pracy programistom frontendu w momencie zakończenia definiowania kontraktu dla danej funkcjonalności, bez czekania na implementację backendu. Pierwsza część tego rozwiązania to serwer mockowy, który jest generowany automatycznie na podstawie definicji kontraktu. Posiada on wszystkie endpointy, które zostały zdefiniowane w kontrakcie, oraz zwraca dane zgodne z definicjami ich interfejsów. Ponadto na bieżąco sprawdza zgodność danych otrzymanych z aplikacji frontend z ich definicją w kontrakcie. Pozwoliło to uniknąć błędów wynikających z niepoprawnego przygotowania danych w aplikacji frontendowej.

Druga część to serwer proxy, przez który przechodzi cała komunikacja. Do jego implementacji wykorzystano środowisko node i bibliotekę „express". Dzięki temu programiści frontendu, którzy na co dzień używają języka JavaScript, mogli zająć się rozwojem tej części projektu. Serwer proxy pozwala podmienić dane wysyłane przez serwer mockowy oraz przez aplikację frontendową, co dało możliwość dowolnej modyfikacji zachowania aplikacji, a dostęp do interfejsów wygenerowanych bezpośrednio z kontraktu, dał gwarancję integralności podmienianych danych.

Aby dodatkowo zabezpieczyć się przed błędami związanymi z nieprawidłowym kształtem przesyłanych danych pomiędzy backendem i aplikacją frontedową, postanowiono, że w serwisach odpowiedzialnych za pobieranie obiektów DTO z backendu, są one przekształcane na instancje klas zawierające dane tego obiektu DTO, a w trakcie tego przekształcenia jest dodatkowo sprawdzana zgodność ich kształtu z zadeklarowanymi interfejsami. Dodatkową korzyścią wynikającą z zastosowania klas domenowych, była możliwość wzbogacania ich o dodatkowe metody, pozwalające przekształcać otrzymane dane zgodnie z aktualnymi wymaganiami. Aby ułatwić generowanie kodu niezbędnego do przygotowania tych klas stworzono bibliotekę, która udostępnia komendy do tworzenia klas typu „entity" oraz „value object". Różnią się one implementacją metody „equals", która służy do porównania dwóch instancji danego obiektu. Typ „entity" porównuje obiekty biorąc pod uwagę klucz „id" obiektu, natomiast typ „value-object" jest porównywany przez analizę struktury porównywanych obiektów. Dzięki temu podejściu zyskano gwarancję, że jeżeli pobrano obiekt DTO z backendu i udało się na jego podstawie stworzyć instancję klasy, to można jej bezpiecznie używać w innych częściach aplikacji.

3.4.2. Testy

Warstwa imitująca backend umożliwiła dodatkowo przygotowanie stabilnych testów end-to-end. Każdy test sprawdza, czy wykonane w jego trakcie czynności nie spowodowały pojawienia się błędu w aplikacji (dzięki zastosowaniu asercji w funkcjach). Przygotowano także rozszerzenie dla biblioteki protractor, które daje możliwość kontrolowania regresji wizualnej aplikacji. Żeby z tego skorzystać, w odpowiednich miejscach testów uruchamiana jest metoda do sprawdzenia regresji wizualnej, która tworzy zrzut ekranu aplikacji i zapisuje go pod zdefiniowaną wcześniej nazwą, a następnie porównuje go z wersją pliku, która została stworzona podczas pierwszego uruchomienia danego testu. Jeżeli w wyglądzie aplikacji wystąpiły znaczące zmiany, zostanie wygenerowany dodatkowy plik z zaznaczonymi różnicami. Dodatkową korzyścią tego rozwiązania jest dostęp do katalogu, w którym znajdują się aktualne zrzuty ekranu całej aplikacji, które mogą być przydatne w czasie przygotowywania prezentacji, bądź szkoleń.

3.4.3. Formularze

Jednym z problemów na jaki napotkano pisząc aplikację był brak typowania formularzy tworzonych przy wykorzystaniu frameworku Angular. W celu zwiększenia wygody w czasie programowania oraz w celu wyeliminowania błędów związanych z nieprawidłowym określaniem kształtu obiektów wewnątrz formularzy, napisano bibliotekę, której zadaniem jest rozszerzenie możliwości modułu formularzy dostarczonych przez framework Angular. Dodano nowe pomocnicze typy „TypedFormControl", „TypedFormGroup"

oraz „TypedFormArray". Dzięki temu umożliwiono skorzystanie z podpowiadania kształtu wartości przechowywanych w formularzach przez nowoczesne edytory.

3.4.4. Wersje językowe

Jednym z wymagań aplikacji JSA było wdrożenie obsługi kilku języków obcych, z możliwością łatwego rozszerzenia obsługi o kolejne języki, oraz możliwość płynnego przełączania aktywnej wersji językowej. Implementację modułu tłumaczeń postanowiono oprzeć na bibliotece ngx-translate. Żeby ułatwić pracę oraz zautomatyzować proces dodawania nowych fraz do tłumaczenia stworzono bibliotekę, która po uruchomieniu analizuje wszystkie pliki projektu w poszukiwaniu kodów używanych przez serwis tłumaczący do zwracania fraz w odpowiednim języku, a następnie buduje odpowiednie słowniki i interfejsy. Dzięki temu w projekcie można korzystać z podpowiadania składni dla kodów tłumaczonych fraz. Ponadto jest możliwość eksportu plików zawierających tłumaczenia do formatu, który jest oczekiwany przez osobę tłumaczącą frazy. Skorzystano w tym celu z programu BabelEdit[10], który pozwala wygodnie eksportować pliki do kilku najpopularniejszych formatów jak .xls lub .csv. Po otrzymaniu uzupełnionych plików z tłumaczeniami od tłumacza, program BabelEdit pozwala zaimportować gotowe tłumaczenia do formatu akceptowanego przez JSA.

3.4.5. Ograniczenia przeglądarek

Aplikacja JSA posiada podstronę, na której można obejrzeć statystki związane z badaną pracą. W związku z tym na ekranie musiano wyświetlić całą treść badanej pracy, łącznie ze źródłami zapożyczeń. W trakcie implementowania tych wymagań zespół programistów napotkał niespodziewany problem, okazało się, że niektóre przeglądarki mają problem z wyświetlaniem elementów HTML zawierających długie fragmenty tekstu (powyżej 2000 znaków). W celu rozwiązania tego problemu przygotowano funkcję, która analizuje tekst jaki do niej przekażemy i modyfikuje go w taki sposób, żeby przeglądarki nie miały trudności z jego wyświetleniem. Funkcja buduje najpierw w pamięci fragment drzewa DOM w taki sposób w jaki zrobiłaby to przeglądarka, a następnie przechodzi po wszystkich elementach tego drzewa i jeżeli natrafi na element, którego tekst jest dłuższy niż wartość przekazana w parametrze funkcji jako wartość maksymalna to dzieli tekst tego elementu na mniejsze części. Ponadto ta funkcja robi audyt bezpieczeństwa kodu wejściowego i blokuje wyświetlenie tekstu w przeglądarce, jeżeli natrafi na fragment kodu, który jest niezgodny z przyjętymi zasadami bezpieczeństwa.

[10] codeandweb.com/babeledit (dostęp: 18.11.2020)

3.4.6. Dostępność dla osób z niepełnosprawnościami

Jednolity System Antyplagiatowy (JSA) wpisuje się w koncepcję projektowania uniwersalnego (ang. universal design) tj. uwzględnia jak najszersze grono użytkowników – w tym osoby z niepełnosprawnościami. W 2018 roku w Polsce 2,5 mln osób posiadało orzeczenie o niepełnosprawności lub niezdolności do pracy. Narodowy Spis Powszechny Ludności i Mieszkań z 2011 roku wskazywał, że łączna liczba osób z niepełnosprawnościami wynosiła niemal 4,7 mln.

Użytkownicy internetu z niepełnosprawnościami to grupa różnorodna, która jest znacznie szersza niż grupa osób niewidomych. Strony internetowe powinny być projektowane z myślą o wszystkich osobach z niepełnosprawnościami tj.:

- z uszkodzonym wzrokiem;
- z uszkodzonym słuchem;
- z dysfunkcjami narządu ruchu (brak kończyn, uszkodzone kończyny, paraliż itd.);
- z zaburzeniami poznawczymi (choroby otępienne np. Alzheimer) i intelektualnymi.

Projektowanie uniwersalne przydatne jest również dla osób:

- z tymczasową niepełnosprawnością (np. złamanie ręki utrudnia korzystanie z myszki);
- ze złymi warunkami w otoczeniu (słońce, hałas, nakaz ciszy np. bibliotece) lub z uszkodzonym sprzętem (niedziałający głośnik lub uszkodzony touchpad);
- ze specyficznym ustawieniem monitora (np. przy dużej jasności nie widać jasnych elementów);
- przyjmujących leki powodujące otępienie;
- starszych;
- z ADHD i dysleksją;
- słabiej wykształconych (trudny język może być barierą np. teksty dotyczące środków unijnych).

Wyszukiwarka Google lepiej pozycjonuje strony projektowane uniwersalnie. Warto zaznaczyć, że dysfunkcji narządu wzroku jest wiele i to powoduje odmienne problemy w zależności od stopnia uszkodzenia wzroku. W tej grupie mamy osoby niewidome (czytnik ekranu), słabowidzące (powiększenie strony), z zaburzeniem rozpoznawania barw, z uszkodzonym wzrokiem (wysoki kontrast), głuchoniewidome (monitory brajlowskie). Pomocne w przygotowaniu uniwersalnej strony są międzynarodowe wytyczne WCAG (wytyczne dla dostępności internetowej). Publiczne strony internetowe muszą spełniać te normy ze względu na Ustawę o dostępności cyfrowej stron internetowych (stan prawny na listopad 2020 roku). W załączniku do dokumentu są wskazane obowiązkowe kryteria sukcesu, które musi spełniać strona internetowa. Jest ich 49 z 78 w WCAG 2.1. Dodatkowo ustawa nakazuje publikację Deklaracji Dostępności. Dostępność to proces ciągły i powinien być nieustannie monitorowany. Jednolity System Antyplagiatowy jest tworzony i monitorowany z uwzględnieniem wytycznych WCAG 2.1 (kryteria sukcesu wskazane w ustawie) oraz dobrych praktyk (np. testowanie trybu wysokiego kontrastu w systemie Windows).

Tworzone nowe elementy i aktualizowane stare są weryfikowane przy pomocy narzędzi:

- czytnik ekranu NVDA;
- HeadingsMap (narzędzie do weryfikacji poprawności nagłówków);
- tryb wysokiego kontrastu systemu Windows;
- WAVE Accessibility Extension (dodatek do Firefoxa wspierający badanie dostępności opracowany przez Uniwersytet Stanowy w Utah);
- narzędzie deweloperskie dostępne w przeglądarkach Firefox oraz Chrome (skrót do uruchomienia: F12);
- strona Contrast Checker (opracowana przez Uniwersytet Stanowy w Utah).

W lutym 2020 roku serwis przeszedł wewnętrzny audyt WCAG. Został on oparty o Metodologię badania dostępności strony www dla osób niepełnosprawnych, starszych i innych narażonych na wykluczenie cyfrowe w oparciu o WCAG 2.0. Dokument został opracowany przez Fundację Widzialni wraz z Uniwersytetem Śląskim. Metodologia została wybrana ze względu na duże doświadczenie Fundacji Widzialni w testowaniu usług publicznych pod kątem WCAG. Od kilku lat organizacja przygotowuje podsumowanie stron publicznych w postaci ogólnodostępnego Raportu Dostępności. Metodologia nie uwzględnia najnowszych wymagań tj. WCAG 2.1, dlatego pomocniczo zastosowano dwa dokumenty dot. WCAG 2.1:

- Lista kontrolna opracowana przez dawne Ministerstwo Cyfryzacji;
- Metoda oceny dostępności cyfrowej strony internetowej zgodnie z zasadami WCAG 2.1.

Przed audytem założono 49 zgłoszeń dotyczących WCAG (jedno zgłoszenie mogło zawierać wiele błędów WCAG). Błędy zostały poprawione. W 2020 roku przeprowadzono również testy Centrum Pomocy JSA. Do programistów zgłoszono 40 błędów. Obecnie kolejne aktualizacje serwisu są monitorowane i poprawiane pod kątem spełnienia wytycznych WCAG.

3.5. Chatbot

Chatbot w warstwie backendowej został podzielony na dwie części: serwis do komunikacji z klientem przeglądarkowym (chatbot broker), oraz serwis stricte obliczeniowy. Aplikacja przeglądarkowa łączy się z brokerem z użyciem REST API. Całość obliczeń związanych z interakcjami użytkowników z interfejsem wykonywana jest po stronie klienta przeglądarkowego. Komunikacja z brokerem pozwala na uzyskanie odpowiedzi chatbota, ocenę odpowiedzi oraz ocenę rozmowy.

3.5.1. Chatbot Broker

Chatbot Broker to serwis z interfejsem REST API, który odpowiada za komunikację z aplikacją przeglądarkową. Serwis ten zapewnia w tej komunikacji mechanizmy bezpieczeństwa. Dodatkowo serwis jest odpowiedzialny za procesy ETL (ang. Extract Transform Load). Baza wiedzy chatbota generowana jest automatycznie na podstawie treści zawartych na stronie Centrum Pomocy JSA opartej o Wordpress. W ramach ETL treści zawarte w bazie Wordpress są pobierane, przetwarzane a następnie zapisywane w lokalnej bazie danych. Dodatkowy proces ETL pozwala na wczytywanie dodatkowych treści do bazy wiedzy chatbota z plików csv oraz plików tekstowych. Dodatkowo serwer ten odpowiada za nadzór nad procesem zapisu historii rozmowy. Serwer oparty jest o stos technologii Java/Spring Boot/Spring Batch.

3.5.2. Serwis obliczeniowy

Serwis ten został wydzielony przez wzgląd na dwie kwestie: potencjalnie wysoka złożoność modeli może wymagać dodatkowej możliwości obliczeniowo-pamięciowych oraz rozproszenia, wsparcie metod uczenia maszynowego jest lepsze w środowisku związanym z Pythonem. Serwis ten odpowiada za matematyczną analizę pytań, przygotowanie modelu oraz jego naukę, a także elementy związane z przetwarzaniem treści zapytań. Jest to „mózg chatbota". Serwer komunikuje się wyłącznie z serwerem REST oraz bazą danych, co zapewnia dodatkowy poziom bezpieczeństwa. Serwer oparty jest o stos technologii Python/Flask/Fairseq.

Obliczenia macierzowe związane z trenowaniem modelu wykonywane są w oparciu o wysoko zoptymalizowane biblioteki. W przypadku chatbota zastosowany został Tensorflow. Optymalizacja modelu matematycznego względem złożoności obliczeniowo-pamięciowej pozwala na naukę zarówno na GPU jak i CPU w akceptowalnym czasie.

ZAKOŃCZENIE

Jednolity System Antyplagiatowy (JSA) jest jedynym prawnie usankcjonowanym i bezpłatnym sposobem weryfikacji oryginalności prac dyplomowych (prace inżynierskie, licencjackie, magisterskie) i doktorskich w Polsce. Decyzję o stworzeniu JSA podjęło Ministerstwo Nauki i Szkolnictwa Wyższego, w celu dostarczenia uczelniom i instytutom narzędzi do weryfikacji prac pod kątem występowania naruszeń prawa autorskiego.

Wprowadzenie Jednolitego Systemu Antyplagiatowego było rekomendowane przez Komisję Europejską. Jego powstanie było odpowiedzią na jedną z największych bolączek polskiego szkolnictwa, czyli braku powszechnego dostępu do systemu antyplagiatowego, który stanowiłby jednoznaczne odniesienie do jakości tekstów pisanych pod zwierzchnictwem uczelni. Kluczowymi zaletami proponowanego rozwiązania jest jednolitość danych, standaryzacja metodyki weryfikacji, jakość i wielkość baz referencyjnych oraz nieodpłatność usługi dla uczelni i instytutów. Dzięki JSA ustandaryzowano proces obrony prac na wielu uczelniach a dostęp do systemu stał się powszechny. Włączenie JSA w życie codzienne uczelni przyczyniło się do zmian w podejściu do dotychczasowego „przymykania oka" na naruszenia praw autorskich. Nareszcie na uczelniach zaczęto wymagać więcej samodzielności w pisaniu prac dyplomowych.

Skala zainteresowania pierwszymi konferencjami, na których prezentowano założenia i prototyp systemu oddawała skalę potrzeb i nadziei środowiska akademickiego na uporządkowanie podejścia do plagiatów na uczelniach. Ostatecznie, JSA działa na dwóch środowiskach wystawionych dla użytkowników. Wraz z szeregiem usług centrum pomocy funkcjonuje produkcyjnie nieprzerwanie od stycznia 2019 roku.

Zgodnie z zapisami ustawowymi, wszystkie pisemne prace inżynierskie, licencjackie, magisterskie i doktorskie dopuszczone do obrony są uprzednio badane za pomocą JSA. Obowiązkowość jego używania w procesie obrony pracy dyplomowej spowodowała, że JSA zbadał już prawie 1 milion prac dyplomowych i doktorskich. W czasie pierwszych dwóch lat istnienia systemu dotarł on do wszystkich polskich uczelni i instytutów naukowych w Polsce, czyli uzyskał prawie 100 tys. użytkowników w około 400 podmiotach naukowych. System oferuje swoim użytkownikom dwa kanały dostępu do usługi, za pomocą graficznego interfejsu użytkownika aplikacji internetowej lub za pomocą interfejsu maszynowego REST API.

Jednolity System Antyplagiatowy od momentu wdrożenia, na początku 2019 roku, przeszedł szereg zmian funkcjonalnych oraz usprawnień jakościowych i w dalszym ciągu prężnie rozwija się. Wszystkie zmiany systemu były wynikiem analizy jego działania przez zespół specjalistów w OPI PIB lub sugerowane przez środowisko akademickie.

Jednym z większych przedsięwzięć było pozyskanie dostępu do treści umieszczonych w bazach poza Ośrodkiem Przetwarzania Informacji – Państwowym Instytutem Badawczym. Sukcesywnie, system powiększał zakres źródeł referencyjnych jak na przykład publikacje naukowe otwartego dostępu, baza postępowań awansowych czy integracje z zewnętrznymi usługami chmury Azure, które odczuwalnie zwiększyły zakres wyszukiwania treści w internecie. Jedną z ostatnio dodanych funkcjonalności JSA jest tłumaczenie maszynowe tekstów badanych prac. Wszystko to ma na celu stale poprawiać jakość i rozszerzać działanie JSA w zakresie większego pokrycia danych źródłowych. W kolejnych latach, zaplanowano rozwój w kierunku nowych metod detekcji, dodawania kolejnych baz referencyjnych oraz optymalizacji interfejsu użytkownika.

SPIS ILUSTRACJI

1.1. JSA – ujęcie ogólne ... 12
1.2. JSA – główne cechy systemu ... 13
1.3. Harmonogram prac ... 15
1.4. Konkurs JSA – etapy ... 23
1.5. Konkurs JSA – zadanie nr 1 ... 25
1.6. Konkurs JSA – zadanie nr 2 ... 26
1.7. Konkurs JSA – zadanie nr 3 ... 27
1.8. Konkurs JSA – zadanie nr 4 ... 28
1.9. Podstawowy scenariusz wykorzystania JSA ... 30
1.10. Przykładowy proces wykorzystania API JSA ... 35
1.11. Statystyki uprawnień w JSA na dzień 15.11.2020 ... 35
1.12. JSA – aktywność użytkowników systemu w czasie, dane na dzień 10.11.2020 40
1.13. JSA – rozkład wartości PRP w podziale na progi i skumulowany, dane na dzień 10.11.2020 ... 42
1.14. JSA – informacje o użytkownikach posiadających konto w JSA i ich historii zakładania na dzień 10.11.2020 ... 42
2.1. Architektura klastra Elasticsearch użytego na potrzebę JSA ... 50
3.1. Schemat architektury JSA ... 72
3.2. Cykl życia dokumentu ... 73

SPIS TABEL

1.1. Tabela statusów badania 32
2.1. Wyniki ewaluacji algorytmu w zależności od obranej metodyki tokenizacji 49
2.2. Wyniki uzyskane w zadaniu 2, fazie 4 konkursu JSA 49
2.3. Wyniki algorytmu $\mathcal{A}_c(\cdot,\cdot)$ na zbiorze walidacyjnym z parametrem $c = 200$ dla metryki *plagdet* (2.9) i jej składowych *precision*, *recall*, *granularity* 59
2.4. Wyniki algorytmu $\mathcal{A}_c$ ze względu na różne poziomy podobieństwa (niskie, średnie, wysokie, bardzo wysokie) dla metryk *plagdet*, *precision*, *recall*, *granularity* 60
2.5. Wyniki eksperymentu klasyfikacji paragrafów 69
2.6. Wyniki eksperymentu dla wszystkich zapytań 70
2.7. Wyniki eksperymentu dla parafraz FAQ 70
2.8. Wyniki eksperymentu dla pytań dodatkowych 70

SPIS FRAGMENTÓW KODU

1.1. Przykład pliku xml z wynikami działania zgłoszonego rozwiązania 25
1.2. Przykład pliku xml z wynikami działania zgłoszonego rozwiązania 26
1.3. Przykład pliku xml z wynikami działania zgłoszonego rozwiązania 27
1.4. Przykład pliku xml z wynikami działania zgłoszonego rozwiązania 28
2.1. Przykład dokumentów w rozumieniu Elasticsearcha 51
3.1. Przykład pliku HTML raportu ze znacznikami, gdzie umieszczane są dane 79
3.2. Skład indeksów wykorzystywanych przez JSA w oparciu o Elasticsearch na dzień 10.11.2020 83

BIBLIOGRAFIA

[1] M. Bawa, T. Condie, P. Ganesan, *LSH Forest: Self-Tuning Indexes for Similarity Search*, *Proceedings of the 14th International Conference on World Wide Web*, WWW '05, Association for Computing Machinery, 2005, s. 651–660.

[2] P. Bojanowski, E. Grave, A. Joulin, T. Mikolov, *Enriching Word Vectors with Subword Information*, *Transactions of the Association for Computational Linguistics*, 2017, 5, s. 135–146, ISSN 2307-387X, DOI: 10.1162/tacl_a_00051.

[3] L. B. Bourque, V. A. Clark, *Processing data: The Survey Example*, SAGE Publications, 1992, ISBN 978-0803947412.

[4] S. Brin, J. Davis, H. García-Molina, *Copy Detection Mechanisms for Digital Documents*, *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, Association for Computing Machinery, 1995, s. 398–409, DOI: 10.1145/223784.223855.

[5] M. Y. M. Chong, *A study on plagiarism detection and plagiarism direction identification using natural language processing techniques*, Rozprawa doktorska, University of Wolverhampton, 2013.

[6] P. Clough, R. Gaizauskas, S. S. L. Piao, Y. Wilks, *METER: MEasuring TExt Reuse*, *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, 2002, s. 152–159, DOI: 10.3115/1073083.1073110.

[7] S. Dadas, M. Perełkiewicz, R. Poświata, *Pre-training Polish Transformer-Based Language Models at Scale*, *Artificial Intelligence and Soft Computing*, 2020, DOI: 10.1007/978-3-030-61534-5_27, s. 301–314.

[8] *Elasticsearch Guide: More like this query*, elastic.co/guide/en/elasticsearch/reference/current/query-dsl-mlt-query.html (dostęp: 18.11.2020).

[9] W. H. Gomaa, A. A. Fahmy, *A Survey of Text Similarity Approaches*, *International Journal of Computer Applications*, 2013, 68, s. 13–18, DOI: 10.5120/11638-7118.

[10] C. Gormley, Z. Tong, *Elasticsearch: The Definitive Guide. A distributed real-time search and analytics engine*, O'Reilly Media, 2015, ISBN 978-1449358525.

[11] V. N. Gudivada, D. L. Rao, A. R. Gudivada, *Chapter 11 – Information Retrieval: Concepts, Models and Systems*, *Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications*, vol. 38 of *Handbook of Statistics*, s. 331–401, DOI: 10.1016/bs.host.2018.07.009.

[12] D. Karaś, M. Śpiewak, P. Sobecki, *OPI-JSA at CLEF 2017: Author Clustering and Style Breach Detection*, *Working Notes of CLEF 2017 – Conference and Labs of the Evaluation Forum*, vol. 1866: CLEF 2017 Working Notes, 2017.

[13] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, *RoBERTa: A Robustly Optimized BERT Pretraining Approach*, arxiv.org/abs/1907.11692 (dostęp: 18.11.2020).

[14] U. Manber, G. Myers, *Suffix Arrays: A New Method for on-Line String Searches*, *SIAM Journal on Computing*, 1993, 22(5), s. 935–948, ISSN 0097-5397, DOI: 10.1137/0222058.

[15] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, *Distributed Representations of Words and Phrases and Their Compositionality*, *Proceedings of the 26th International Conference on Neural Information Processing Systems – Volume 2*, NIPS'13, 2013, s. 3111–3119.

[16] S. Piao, P. Clough, J. Arundel, *Proposing Basic Approaches to Detecting Text Rewrite*, 2001.

[17] M. Potthast, B. Stein, A. Barrón-Cedeño, P. Rosso, *An Evaluation Framework for Plagiarism Detection*, *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, Association for Computational Linguistics, 2010, s. 997–1005.

[18] J. Ramos, *Using TF-IDF to determine word relevance in document queries*, *Proceedings of the first instructional conference on machine learning*, vol. 242, 2003, s. 29–48.

[19] S. Ruder, M. E. Peters, S. Swayamdipta, T. Wolf, *Transfer Learning in Natural Language Processing*, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, 2019, s. 15–18, DOI: 10.18.653/v1/N19–5004.

[20] C. Sammut, G. I. Webb, *Encyclopedia of Machine Learning*, Springer Science & Business Media, 2011, ISBN 978-0387307688.

[21] M. A. Sanchez-Perez, A. Gelbukh, G. Sidorov, *Adaptive Algorithm for Plagiarism Detection: The Best-Performing Approach at PAN 2014 Text Alignment Competition*, *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, Springer International Publishing, 2015, s. 402–413.

[22] E. Stamatatos, *Intrinsic plagiarism detection using character n-gram profiles*, *Proceedings of the SEPLN'09 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse*, 2009.

[23] E. Stamatatos, *A survey of modern authorship attribution methods*, *Journal of the American Society for Information Science and Technology*, 2009, 60(3), s. 538–556, DOI: 10.1002/asi.21001.

[24] L. Torrey, J. Shavlik, *Transfer learning*, *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, s. 242–264.

[25] *Ustawa z dnia 20 maja 2016 r. o zmianie ustawy – Prawo o szkolnictwie wyższym oraz ustawy o zmianie ustawy – Prawo o szkolnictwie wyższym oraz niektórych innych ustaw*, Dz.U. 2016 poz. 908.

[26] *Ustawa z dnia 20 lipca 2018 r. – Prawo o szkolnictwie wyższym i nauce*, Dz.U. 2018 poz. 1668.