Wydział Elektroniki i Technik Informacyjnych

Politechnika Warszawska

Instytut Automatyki i Informatyki Stosowanej

Praca dyplomowa inżynierska

Mapa miejsc ciekawych

Michał Cybulski

Praca dyplomowa inżynierska
wykonana pod kierunkiem
dr inż. Mariusza Kamoli

Warszawa 2015

Streszczenie

Temat pracy: *Mapa miejsc ciekawych*

Słowa kluczowe: *sieci społecznościowe*, *geolokalizacja*, *geotagging*, *analiza danych*.

Celem pracy było stworzenie zestawu narzędzi (programów i skryptów wykonywalnych), który najpierw umożliwi pobranie i agregację dużej ilości danych geolokalizacyjnych zawartych w zdjęciach pochodzących z terytorium Polski z popularnego internetowego serwisu społecznościowego, a następnie pozwoli użytkownikowi na analizę zebranych danych w celu znalezienia wyróżniających się wyjątkowo ciekawych lub nieciekawych miejsc. Stworzone narzędzia pozwalają na dopasowanie informacji o liczbie zebranych próbek danych pogrupowanych w poszczególnych polskich gminach do zewnętrznych danych statystycznych pobranych oddzielnie z zasobów *Głównego Urzędu Statystycznego*. Umożliwiają one także wyświetlenie zebranych próbek na tle mapy Polski.

W pracy pokazano cały przebieg tworzenia narzędzi, wraz z najciekawszymi problemami i trudnościami, które trzeba było przezwyciężyć aby osiągnąć końcowy rezultat, oraz z ciekawymi rozwiązaniami i szczegółami implementacyjnymi, które zastosowano. Zaprezentowano także prosty przykład użycia powstałych w ramach pracy narzędzi, wraz z wynikami podstawowych analiz danych zebranych w ramach testu.

Abstract

Thesis subject: *The map of interesting places*

Keywords: *social networks*, *geolocation*, *geotagging*, *data analysis*.

The purpose of this thesis was the creation of a set of tools (programs and executable scripts), that would at first allow for the collection of large amounts of geolocational data included in photos from the territory of Poland from a popular social network, and afterwards let its users analyze the accumulated data in order to find exceptionally interesting or uninteresting places. The tools allow its users to match information regarding the amount of collected samples from different municipalities to external statistical data taken from the *Central Statistical Office of Poland*. They also include the utility of showing the collected samples on a map of Poland.

This thesis shows the entire process of how these tools were created, together with the most interesting problems and difficulties which had to be overcome to reach the final result, also including interesting solutions and details of how they were implemented. It also presents a simple example of possible uses of said tools, including the results of basic analyses of data collected as part of a test.

Spis treści

1. Cel i motywacja pracy

Zasadniczym celem wykonywanej pracy inżynierskiej było stworzenie zestawu narzędzi (tj. aplikacji i skryptów wykonywalnych), który umożliwiłby użytkownikowi przeprowadzenie badań regionów geograficznych wykorzystując dane geolokalizacyjne pochodzące z popularnej sieci społecznościowej w celu znalezienia „ciekawych", ale jeszcze mało znanych miejsc. Takie miejsca charakteryzowałyby się nieproporcjonalnie dużą popularnością w stosunku do reprezentatywnych danych statystycznych danego regionu.

Szukanie takich interesujących miejsc polegałoby na pobraniu dużej ilości danych geolokalizacyjnych zawartych w zdjęciach umieszczonych przez użytkowników internetu w popularnej internetowej sieci społecznościowej, dopasowaniu ich do danych statystycznych z zewnętrznego źródła pod względem pochodzenia z tych samych regionów geograficznych, a następnie na próbie znalezienia korelacji pomiędzy danymi z obu źródeł. W przypadku znalezienia między nimi jakiegoś rodzaju współzależności, można wyszukiwać pary próbek, które są najbardziej odchylone od typowych wartości. Skrajne odchylenia sugerowałyby, że dane miejsce jest wyjątkowo i nieproporcjonalnie popularne lub niepopularne w sieci społecznościowej w stosunku do danej statystycznej wielkości, z którą następuje porównanie — takie miejsca można by nazwać wyjątkowo „ciekawymi" lub „nieciekawymi". Głównym zadaniem tworzonych narzędzi miało być więc pomaganie w prowadzeniu badań analizujących wzajemny związek danych umieszczanych w sieciach społecznościowych z innymi danymi statystycznymi — ze szczególnym uwzględnieniem regionów geograficznych.

W przypadku pracy nad programami, które korzystają z zasobów udostępnianych przez zewnętrzne systemy, należy wziąć pod uwagę wydajność tych systemów i ograniczyć ewentualny wpływ tworzonych aplikacji na ich działanie. Trzeba pamiętać o zasadach *netykiety*[1]. Dlatego też, jednym z założeń było ograniczenie w miarę możliwości generowania sieciowego ruchu i obciążania zewnętrznych systemów. Analogicznie, podczas pracy nad narzędziami założeniem było uszanowanie prywatności użytkowników sieci społecznościowej, z której pochodzić miały dane. Żadna z funkcjonalności tworzonego programu nie miała mieć więc na celu jej naruszania.

Jednym z założeń przy tworzeniu narzędzi było umożliwienie użytkownikowi dodatkowego opcjonalnego agregowania danych pobranych z sieci społecznościowej w kontekście podanych przez niego haseł (tzw. „tagów"), tak żeby do dalszej analizy brać pod uwagę tylko powiązane z nimi próbki. Takie hasła byłyby słowami-kluczami

[1]Netykieta (zbitka wyrazowa: „*net*" (ang. „*sieć*") i „*etykieta*") — zbiór niepisanych zasad przyzwoitego zachowania w Internecie, swoista etykieta obowiązująca w sieci.

opisującymi analizowane dane, na przykład: „ptaki", „góry", „architektura" lub „morze". Znalezione w ten sposób wyróżniające się miejsca geograficzne byłyby w danym kontekście ze względu na wyjątkową popularność lub niepopularność z jakichś powodów „ciekawe" bądź „nieciekawe". Posiadając dane o tagach, oraz o ich geograficznym rozmieszczeniu można także analizować sytuację od drugiej strony — skupić się na konkretnym regionie i badać popularność poszczególnych haseł na jego obszarze.

Wprawieni w analizie danych ludzie często potrafią od razu zauważyć pewne wzorce i nietypowe sytuacje w rozkładzie próbek danych, gdy tylko zobaczą je zaprezentowane w graficznej formie. Przedstawianie danych w ten sposób jest dla człowieka znacznie bardziej intuicyjne niż prezentowanie mu tylko i wyłącznie tabelek wypełnionych liczbami — zwłaszcza, gdy ilość próbek jest rzędu dziesiątek lub setek tysięcy. W celu umożliwienia końcowemu użytkownikowi swobodnego rozeznania się w ogólnej ilości danych pobranych z sieci społecznościowej i ich rozłożeniu na rozpatrywanym obszarze, tworzone narzędzia powinny udostępniać funkcjonalność interaktywnego wyświetlania zebranych próbek na tle mapy analizowanych regionów. Już nawet taka wstępna analiza może prowadzić do pewnych konkluzji.

Analiza, dlaczego dane miejsce jest „ciekawe" lub „nieciekawe", jest bardzo interesującym zagadnieniem, ale wykracza już poza zakres pracy — jest to już zajęcie dla końcowego użytkownika aplikacji, który najlepiej będzie rozumiał kontekst prowadzonych przez siebie badań. Tworzone narzędzia miały być pomocą do przeprowadzania tego typu analiz, powinny umożliwić zebranie danych oraz wygodne nimi zarządzanie.

Tworzone narzędzia miały być przeznaczone dla użytkowników, którzy zaznajomieni są z użyciem programów działających w konsoli systemu operacyjnego oraz znają się na samodzielnej analizie danych. Przygotowane w ramach pracy programy i skrypty są więc głównie aplikacjami konsolowymi i nie posiadają rozbudowanego graficznego interfejsu użytkownika. Mogą być natomiast dzięki temu wygodnie używane w skryptach tworzonych przez końcowego użytkownika. Umożliwiają w prosty, automatyczny i wygodny sposób przygotowanie całego środowiska gotowego do zbierania, grupowania i analizowania danych, a następnie automatyczne pobieranie dużej ilości danych z sieci społecznościowej, dopasowanie ich do danych statystycznych z zewnętrznego zaufanego źródła podanych w ustalonym formacie i eksport rezultatu do popularnego, uniwersalnego formatu danych. Rezultat ten można byłoby potem wygodnie przetwarzać i analizować dalej w sposób, do którego przyzwyczajony jest końcowy użytkownik (np. W środowiskach *Octave* lub *Matlab*).

W razie pojawienia się pomysłów bazujących na użyciu danych geolokalizacyjnych pobranych z sieci społecznościowych (a więc założeniach bardzo podobnych do

założeń tej pracy), dla realizacji których z jakichś powodów funkcjonalność stworzonych narzędzi nie byłaby wystarczająca, miała istnieć możliwość jej względnie prostej rozbudowy. Trzeba pamiętać, że zarówno internet jak i sieci społecznościowe są środowiskami ulegającymi ciągłym i bardzo dynamicznym zmianom. Pojawianie się nowych ciekawych możliwości i rozwiązań technicznych następuje cały czas. Tworzony zestaw narzędzi musiał więc być projektowany z myślą aby umożliwić jego prostą rozbudowę w przyszłości, w przypadku wystąpienia takiej potrzeby. Założeniem było więc tworzenie kodu źródłowego narzędzi zgodnie z dobrymi praktykami programistycznymi i inżynierskimi. Ponadto kod źródłowy tworzonej aplikacji (oraz pomocniczych skryptów) miał być dobrze dokumentowany, aby ułatwić zorientowanie się w strukturze programu nawet osobie niezwiązanej od początku z projektem. Przy tworzeniu narzędzi użyte miały zostać uniwersalne technologie i standardy tak, aby nie ograniczać możliwości ich użycia do jednego wybranego systemu operacyjnego i konkretnych dostawców danych.

2. Zastosowanie oraz istniejące rozwiązania

2.1. Potencjalne zastosowania

Potencjalnych zastosowań dla narzędzi tworzonych w ramach tej pracy jest wiele. Narzędzia te mogą być przydatne zarówno w sektorze zastosowań prywatnych — na przykład biznesowych i marketingowych — jak i w sektorze publicznym. Mogą wnieść jakiegoś rodzaju zysk wszędzie, gdzie istnieje potrzeba badania atrakcyjności regionów geograficznych, lub badania aktualnych trendów u części społeczeństwa, która korzysta z sieci społecznościowych. Należy pamiętać, że do dyspozycji użytkownika poza funkcjonalnością aplikacji dopasowującą próbki danych geolokalizacyjnych z sieci społecznościowej do cech statystycznych według pochodzenia z tych samych regionów, dostępna jest także obszerna baza danych zawierająca hasła (tagi), którymi użytkownicy sieci społecznościowej opisali próbki danych. Wraz z dodatkowymi informacjami takimi jak ilość wyświetleń próbek opisanych danym hasłem czy liczba komentarzy ich dotyczących, może to być potencjalnie dobrym źródłem wiedzy o internetowych trendach, które bardzo często i dynamicznie się zmieniają. Przykładowe zastosowania stworzonego zestawu narzędzi, to:

- *Wspomaganie zarządzania regionami administracyjnymi kraju* — prowadzenie badań atrakcyjności regionów może pomóc w wykrywaniu regionów o wyjątkowo niskiej popularności w stosunku do regionów sąsiadujących. Może to być oznaka złego zarządzania lub istnienia jakiegoś problemu na rozpoznanym w ten sposób obszarze. Takie przypadki można by dokładniej analizować i w przypadku wykrycia faktycznych nieprawidłowości można by im przeciwdziałać.
- *Promowanie atrakcji i ciekawych regionów o niedostatecznej jeszcze infrastrukturze* — znalezienie obszarów, których popularność wskazuje na niedostateczną ilość dostępnej infrastruktury (na przykład turystycznej, ale problem jest bardzo ogólny) może pomóc w podjęciu decyzji inwestycyjnych. Mogłoby się okazać, że taki region ma w rozpatrywanym kontekście bardzo duży potencjał i inwestycja w jego rozwój może stać się bardzo opłacalna.
- *Przeprowadzanie kampanii reklamowych dostosowanych do popularnych haseł w danym rejonie* — dostępność w bazie danych haseł (tagów) skojarzonych z danymi geolokalizacyjnymi, a także z danymi o dacie umieszczenia danych w sieci społecznościowej pozwala na analizowanie najpopularniejszych haseł w danej chwili na danym obszarze. Tą metodą możliwe jest wykrywanie nowych trendów zanim

staną się zauważalne innymi metodami analizy. Takie informacje mogą być wykorzystane w celach marketingowych — mogą umożliwiać na przykład wprowadzenie w przeanalizowanym regionie kampanii reklamowej dostosowanej pod popularne w tym rejonie hasło.

- *Analiza atrakcyjności turystycznej* — przy założeniu, że zagraniczni użytkownicy częściej opisują zdjęcia hasłami w językach obcych w kontekście analizowanego obszaru, porównując rozbieżności w rozkładach danych dla języków obcych i lokalnie używanych można analizować atrakcyjność danego regionu dla zagranicznych turystów. Regiony o niskiej atrakcyjności dla osób z zagranicy można dodatkowo promować.

Użytkownik końcowy aplikacji może użyć danych zebranych przez stworzone na potrzeby tej pracy narzędzia w innych celach lub na inne sposoby, niż te które były pierwotnie w zamyśle autora. Aby na to pozwolić stworzona aplikacja zbiera szerszy zakres danych niż tylko ten, który później jest przez nią wykorzystywany. Wynika to z tego, że „dodatkowe" dane często dostępne są razem z „podstawowymi" danymi i ich ściągnięcie powoduje pomijalnie mały dodatkowy narzut i obciążenie (także sieciowe) — warto więc zapisywać także i je. Narzędzia stają się dzięki temu bardziej uniwersalne i ich zastosowanie może znacznie wybiegać poza cele pierwotnie zaplanowane.

2.2. Istniejące rozwiązania

W momencie pisania pracy nie znalazłem żadnych istniejących już rozwiązań, o założeniach takich jak tworzone w jej ramach narzędzia — istniały natomiast rozwiązania, których funkcjonalność w pewnym stopniu je przypominała, bądź też częściowo zapewniała. Pod uwagę biorę wziąłem więc kilka popularnych systemów lub aplikacji, które moim zdaniem mogą być uznane za podobne pod pewnymi względami.

- *Walk Score* (adres `https://www.walkscore.com/`) — serwis, którego celem jest ułatwienie użytkownikowi podjęcia decyzji o wyborze miejsca zamieszkania. Stara się oszacować jak wiele miejsc zapewniających podstawowe codzienne usługi można znaleźć w odległości możliwej do bezproblemowego pokonania na piechotę. Podobnie jak tworzone w ramach pracy narzędzia do przeprowadzenia analizy serwis ten używa i łączy dane z wielu źródeł[2] w celu zaprezentowania użytkownikowi wyniku analiz na mapie. Nie umożliwia przeprowadzania badań innego typu.

[2] Źródła danych dla *Walk Score* to między innymi: *Google*, *Education.com*, *Open Street Map* oraz *Localeze*. Źródło: Dustin T. Duncan, Jared Aldstadt, John Whalen, Steven J. Melly, Steven L. Gortmaker [1]

- *Twitter.com* (adres `https://twitter.com/`) — serwis społecznościowy opierający się na idei rozgłaszania przez użytkowników krótkich haseł (tagów) i swoich opinii na rozmaite tematy. Pozwala monitorować najpopularniejsze hasła — zarówno w kontekście globalnym, jak i na zadanym terytorium. Dokładność udostępnianej geolokalizacji jest jednak dość mała (treści umieszczane przez użytkowników grupowane są do ogólnych miejsc pochodzenia, jak na przykład „Warszawa" lub „Polska"), a informacje o geolokalizacji są zupełnie opcjonalne i bardzo często pomijane przez użytkowników. Serwis udostępnia więc w pewien sposób funkcjonalność związaną z treściami udostępnianymi przez użytkowników internetu w kontekście geolokalizacji, ale o bardzo ograniczonych możliwościach w kontekście badawczym.

- Funkcja „*World Map*" w serwisie *Flickr.com* (adres `https://www.flickr.com/map`) — jedna z funkcjonalności serwisu społecznościowego skoncentrowanego na dzieleniu się zdjęciami, która umożliwia wyświetlenie na wybranym fragmencie świata popularnych zdjęć dotyczących zadanych haseł. Funkcja ta używa więc danych geolokalizacyjnych zawartych w zdjęciach umieszczonych w sieci społecznościowej, umożliwia także agregowanie zdjęć pod względem zadanych haseł. Niestety rozwiązanie to może być traktowane tylko jako ciekawostka, gdyż nie pozwala przeprowadzać żadnych badań, i jest jedynie swego rodzaju małą galerią zdjęć z wybranego obszaru.

Pomysły wykorzystania danych o geolokalizacji pochodzących od użytkowników sieci społecznościowych, oraz łączenia ogólnych danych geolokalizacyjnych z wielu źródeł w celach jakiegoś rodzaju wnioskowania, pojawiły się już wcześniej, ale jednoczesne połączenie obu tych pomysłów — czyli wykorzystanie danych geolokalizacyjnych pobranych z sieci społecznościowej do porównań z danymi statystycznymi z innych źródeł i dalszej analizy — w połączeniu z możliwie dużą automatyzacją procesu badawczego za pomocą specjalnie do tego przygotowanych narzędzi jest podejściem nowatorskim, które może okazać się bardzo przydatne w badaniach związanych z wieloma różnymi dziedzinami.

3. Wprowadzenie do tematyki pracy

3.1. Internetowe sieci społecznościowe

3.1.1. Ogólne informacje

Internetowa *sieć społecznościowa* to serwis internetowy, który istnieje i funkcjonuje w oparciu o zgromadzoną wokół niego społeczność. Serwisy takie mogą zajmować się rozmaitą tematyką oraz mogą im przyświecać różne cele, ale istnieją cechy wspólne dla każdej sieci społecznościowej — serwisy takie umożliwiają:

- Tworzenie internetowej tożsamości użytkownika — publicznego profilu, który jest jego wirtualną, internetową wizytówką i reprezentacją w danym systemie. Profile mogą zwierać opisy zainteresowań, a także rozmaite informacje o użytkowniku, takie jak jego miejsce zamieszkania lub data urodzenia.
- Tworzenie listy kontaktów z innymi użytkownikami serwisu. Stopniowo rozrastające się listy „wirtualnych znajomych" tworzą w systemie swego rodzaju sieć użytkowników połączonych różnymi relacjami.
- Interakcję społeczną z innymi użytkownikami, zwłaszcza tymi znajdującymi się na listach kontaktów. Interakcja ta może wyglądać różnorako — może na przykład polegać na wymianie poglądów politycznych, lub dzieleniu się własną twórczością i przemyśleniami.

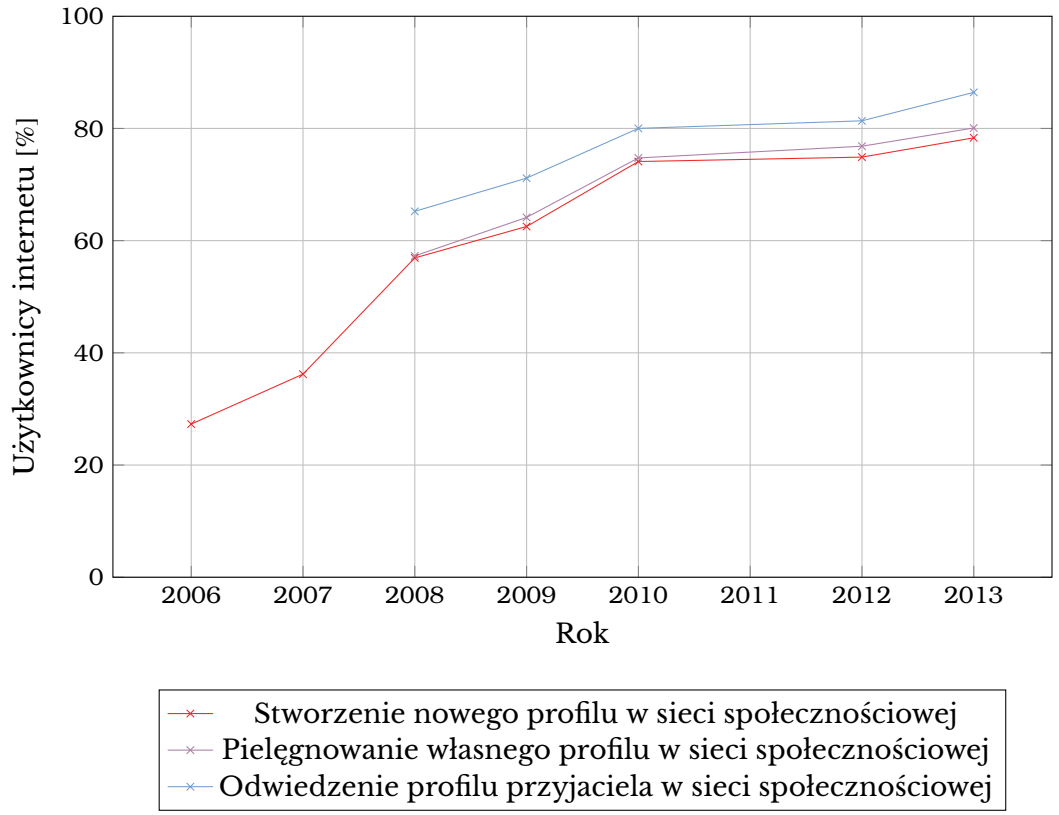
Ze względu na istnienie modelu relacji i na możliwość interakcji pomiędzy użytkownikami sieci społecznościowych, serwisy takie mogą być bardzo dobrym źródłem danych o części społeczeństwa, która z nich korzysta — a ich liczba ciągle rośnie, i jest to trend utrzymujący się od czasu pojawienia się pierwszych takich serwisów, jak pokazuje wykres 3.1.

3.1.2. Przegląd sieci społecznościowych

Głównym źródłem danych dla tworzonych narzędzi miały być metadane zawarte w zdjęciach umieszczanych w internecie przez użytkowników popularnych portali społecznościowych. Z tego względu chciałbym przedstawić przegląd popularnych serwisów tego typu, biorąc pod uwagę serwisy najpopularniejsze oraz serwisy mniej popularne, ale których podstawowym celem jest umożliwienie dzielenia się zdjęciami:

- *Facebook* (`https://www.facebook.com/`) — Najpopularniejszy na świecie i dla większości użytkowników podstawowy[3] portal społecznościowy.

[3] *Źródło: PewResearch Internet Project [3]*



Rys. 3.1. *Procent użytkowników internetu korzystających z poszczególnych funkcji sieci społecznościowych w latach 2008-2013.* Źródło: Opracowanie własne na podstawie Universal McCann [2]

- Posiada ogromną bazę użytkowników: średnio około 864 milionów użytkowników dziennie[4].
- Umożliwia użytkownikom zbudowanie pomiędzy sobą sieci relacji różnego typu — od znajomych do relacji rodzinnych.
- Jest bardzo wszechstronny — umożliwia użytkownikom dość niezawodną komunikację, dzielenie się przebiegiem dnia, opiniami na wszelkie tematy, udostępnianie zdjęć i filmów, śledzenie aktualności obserwowanych osób i firm, oraz wiele innych.
- Posiada minimalną użyteczność dla osób niezarejestrowanych, oraz nie pozwala użytkownikom dowolnie przeglądać zasobów — funkcjonalność serwisu skupiona jest na komunikacji pomiędzy użytkownikami między którymi są zdefiniowane relacje.
- Wszystkie wysłane zdjęcia są przetwarzane i pozbawiane oryginalnych metadanych — w tym tych odpowiedzialnych za geolokalizację. Umożliwia jednak użytkownikom ręczne ustawianie lokalizacji, w której zdjęcie zostało zrobione.

[4] *Źródło (11.01.2015):* `http://newsroom.fb.com/company-info/`

  - Jest dość mocno zamknięty — udostępnia ubogie *API*[5] dla programistów, za pomocą którego można przeglądać głównie zasoby stworzone przez siebie oraz przez „znajomych" — brakuje możliwości efektywnego przeszukania danych stworzonych przez obcych ludzi.

- *Instagram (`http://instagram.com/`)* — Popularny i modny serwis społecznościowy, który skupiony jest na dzieleniu się ze znajomymi zdjęciami robionymi za pomocą *smartfonów*.
  - Posiada bardzo dużą bazę użytkowników: około 300 milionów[6] osób.
  - Pozwala opisywać wysyłane zdjęcia hasłami (proces ten nazywa się *tagowaniem*), a potem w ograniczonym zakresie wyszukiwać zdjęcia ze względu na te hasła.
  - Dla niezarejestrowanego użytkownika serwis jest praktycznie całkowicie zamknięty.
  - Wysyłane do serwisu zdjęcia pozbawiane są metadanych. Mimo wszystko autorzy zdjęć mają możliwość ręcznego ustawiania geolokalizacji.
  - Posiada *API*, które pozwala w pewnym zakresie przeglądać zasoby serwisu — da się na przykład wyszukać najpopularniejsze zdjęcia skojarzone z danym hasłem, albo wyszukać zdjęcia zrobione w bardzo niedalekiej odległości (kilka kilometrów kwadratowych) od zadanego miejsca. Skorzystanie z *API* wymaga rejestracji w serwisie i ograniczone jest do 5000 zapytań na godzinę.

- *Flickr (`https://www.flickr.com/`)* — Portal społecznościowy, którego podstawowym celem jest udostępnienie użytkownikom wygodnego systemu do dzielenia się stworzonymi przez siebie zdjęciami. Skierowany jest do osób trochę znających się na fotografii, nie próbuje jednak ograniczać zbioru użytkowników wyłącznie do profesjonalnych fotografów. Serwis nie skupia się na budowaniu zamkniętych sieci kontaktów, lecz na rozbudowanych możliwościach zarządzania zdjęciami.
  - Baza użytkowników liczy około 92 milionów[7] osób
  - Pozwala opisywać zdjęcia tagami, udostępnia zaawansowaną wyszukiwarkę z nimi związaną.

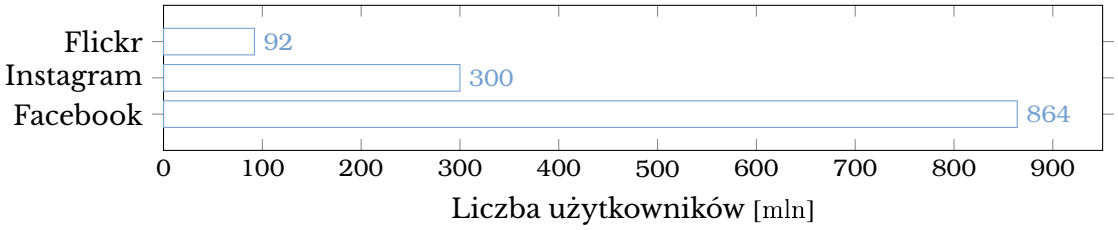
[5] API (ang. *„Application Programming Interface"*) — Interfejs programistyczny aplikacji, czyli zestaw ściśle określonych zasad i reguł według których programy komunikują się ze sobą.

[6] *Źródło (11.01.2015):* `http://mashable.com/2014/12/10/instagram-300-million-users/`

[7] *Stan na luty 2014. Źródło:* `http://techcrunch.com/2014/02/10/flickr-at-10-1m-photos-shared-per-day-170-increase-since-making-1tb-free/`

– Zdjęcia umieszczane w serwisie nie są pozbawiane metadanych. Użytkownicy mają dostęp do metadanych, w tym do oryginalnych danych geolokalizacyjnych ustawionych pierwotnie przez aparat fotograficzny.
– Nie ogranicza dostępu do zasobów niezarejestrowanym użytkownikom. Zarejestrowani użytkownicy zyskują dostęp do funkcjonalności związanych z budowaniem między sobą relacji i aktywnym udzielaniu się w społeczności.
– Udostępnia rozbudowane *API* związane z wyszukiwaniem zdjęć — zarówno w kontekście zadanych wyszukiwarce haseł, jak i w kontekście obszarów geograficznych. Interfejs programistyczny pozwala zawęzić wyszukiwanie zdjęć do obszaru ograniczonego przez tak zwany „bounding boks", czyli obszar ograniczony czterema wierzchołkami rozstawionymi na rozpatrywanej powierzchni. *API* nie posiada sztywno zdefiniowanych ograniczeń dotyczących generowania ruchu sieciowego, ale serwis zastrzega sobie prawo do blokowania dostępu użytkownikom i aplikacjom, które byłyby w praktyce uciążliwe.

Na wykresie 3.2 znajduje się wizualne porównanie liczby użytkowników poszczególnych sieci.



Rys. 3.2. *Szacowana liczba użytkowników poszczególnych serwisów społecznościowych.*

3.2. Zdjęcia jako źródło danych

3.2.1. Informacje ogólne o formacie *EXIF*

Zdjęcia umieszczane przez użytkowników internetu w sieciach społecznościowych intuicyjnie nie wydają się być dobrym źródłem danych geolokalizacyjnych, lecz są to tylko pozory. Nowoczesne cyfrowe aparaty fotograficzne automatycznie dodają do samych danych o obrazie bardzo wiele dodatkowych metadanych informujących o okolicznościach utrwalania fotografii. Funkcjonalność taka realizowana jest za pomocą formatu *EXIF* (jego techniczne szczegóły opisane są w rozdziale 3.2.4). Lista zapisywanych informacji różni się w zależności od konkretnego modelu aparatu, ale istnieje wiele ogólnych informacji, które są zapisywane przez praktycznie każde urządzenie. Między innymi są to informacje o modelu i producencie aparatu,

dacie wykonania zdjęcia, czasie naświetlania, ustawieniach obiektywu, i wiele, wiele innych.

W szczególności, urządzenia wykorzystujące moduł *GPS*[8] mają możliwość umieszczania w tworzonych przez siebie zdjęciach metadanych związanych z współrzędnymi geograficznymi urządzenia w momencie wykonywania fotografii. Warto zauważyć, że możliwość taką ma zdecydowana większość nowoczesnych telefonów komórkowych, które są coraz częściej wykorzystywane przez zwykłych ludzi jako podstawowy i zawsze noszony przy sobie aparat fotograficzny. Należy także dodać, że w większości modeli telefonów funkcje zapisywania danych o geolokalizacji urządzenia podczas zapisywania fotografii są domyślnie włączone. W połączeniu z rosnącą popularnością zarówno tego typu urządzeń (wykres 3.3) obsługujących standard *GPS* jak i serwisów społecznościowych umożliwiających użytkownikom dzielenie się zdjęciami, sprawia to, że zdjęcia umieszczane w internecie mogą być uważane za źródło danych o bardzo dużym (i co najważniejsze — rosnącym) potencjale dla różnorakich badań.

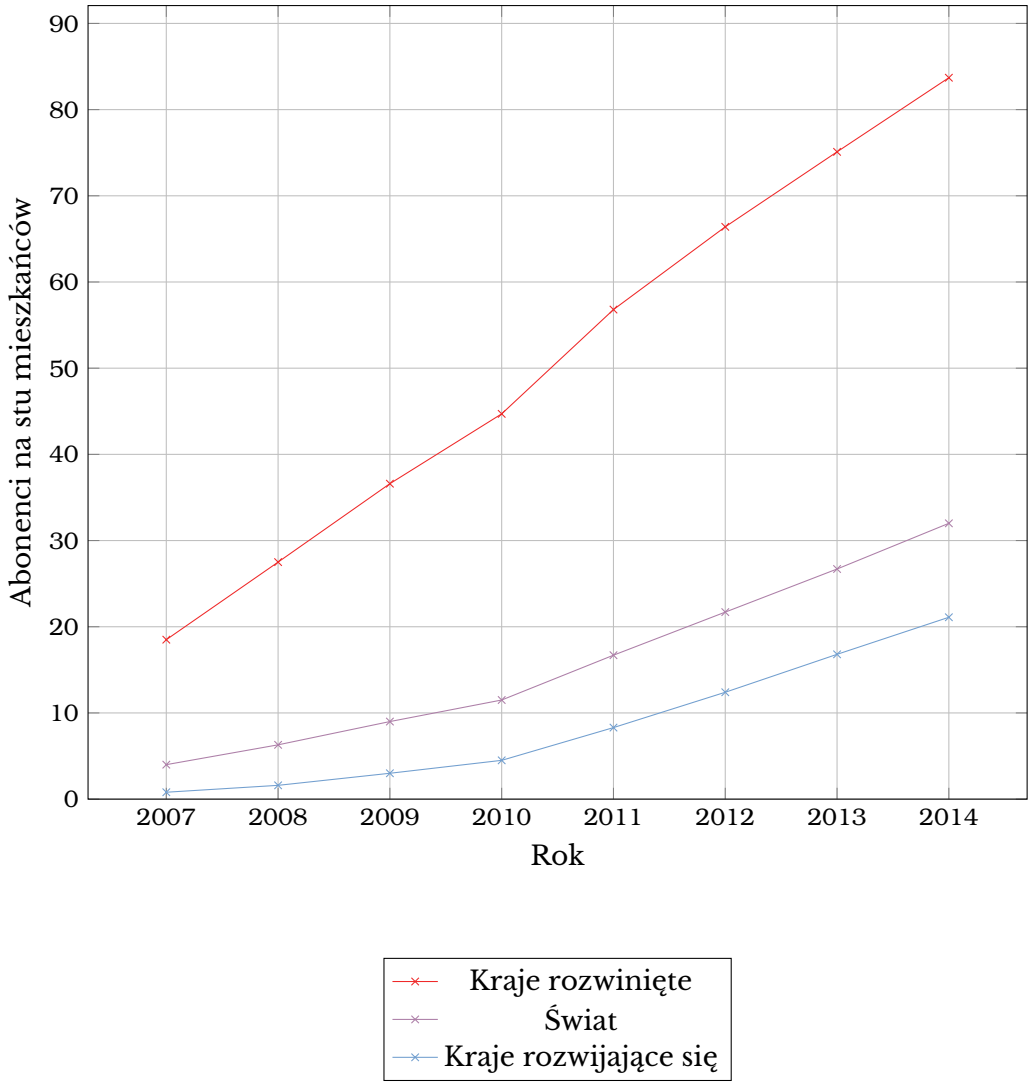
3.2.2. Popularność urządzeń wspierających technologię

Międzynarodowy Związek Telekomunikacyjny (ang. *ITU — „International Telecommunication Union"*) co roku udostępnia statystyki dotyczące między innymi ilości aktywnych abonentów usług mobilnych na sto osób, z podziałem na kraje rozwinięte, rozwijające się, oraz na średnią światową. Niestety udział *smartfonów* w rynku takich urządzeń nie jest dosłownie wyszczególniony, ale w udostępnionych dokumentach istnieje podział na ilość ogólnych usług mobilnych, oraz na ilość szerokopasmowych usług mobilnych. Znakomita większość urządzeń używanych przy usługach drugiego typu to właśnie *smartfony*, w których obecność modułu *GPS* jest standardem. Wykres obrazujący wspomniane statystyki znajduje się na rysunku 3.3. Można zaobserwować na nim wyraźny wzrost popularności tego typu usług — a co za tym idzie, popularności urządzeń obsługujących te technologie.

3.2.3. Zagadnienia dotyczące prywatności

Istnieje pewien negatywny aspekt tej sytuacji, który może budzić moralne wątpliwości. Ze względu na to, że opcja zapisywania informacji o współrzędnych geograficznych w większości urządzeń obsługujących tę funkcję jest domyślnie włączona, prawdopodobnie wielu ich użytkowników nie jest świadomych, że takie informacje są zapamiętywane w wynikowych plikach z fotografiami. Dotyczy to

[8] GPS (ang. *„Global Positioning System"*) — jeden z systemów nawigacji satelitarnej, stworzony przez *Departament Obrony Stanów z jednoczonych*, obejmujący swoim zasięgiem całą kulę ziemską



Rys. 3.3. *Liczba abonentów szerokopasmowych usług mobilnych na stu mieszkańców.* Źródło: International Telecommunication Union [4]

zwłaszcza osób nie posiadających dużej wiedzy o nowoczesnych technologiach. Prowadzi to do sytuacji, w której dane udostępniane publicznie w internecie przez takich użytkowników mogą być potencjalnie użyte ze złymi intencjami w celu naruszenia ich prywatności. Niestety jedynym sposobem na walkę z tego typu zagrożeniami jest uświadamianie takich użytkowników internetu o kwestiach związanych z bezpieczeństwem i prywatnością. Narzędzia tworzone w ramach pracy nie mają służyć do jakiejkolwiek ingerencji w prywatność użytkowników sieci społecznościowej — dane przez nie pobierane służą tylko i wyłącznie do analizy statystycznej, bez skupiania się na konkretnych użytkownikach sieci społecznościowej.

3.2.4. Techniczne szczegóły formatu *EXIF*

Format *EXIF* (ang. *„Exchangeable image file format”*) jest standardem metadanych dla plików z obrazami[9] obsługiwanym i używanym przez prawie wszystkie cyfrowe aparaty fotograficzne (wliczając w to *smartfony*) i skanery, oraz aplikacje i systemy, które później przetwarzają pliki stworzone przez te urządzenia. Format *EXIF* w zdecydowanej większości przypadków nie jest używany jako samodzielny plik, lecz jako dodatkowy element „wstrzykiwany” do istniejącego już pliku w innym formacie. Oficjalna specyfikacja zakłada użycie następujących zewnętrznych formatów[5]:

- *JPEG* — dla plików z obrazami, stratnie skompresowanych.
- *TIFF Rev. 6.0* — dla plików z obrazami, nieskompresowanych.
- *RIFF WAV* — dla plików dźwiękowych, nieskompresowanych.

Każdy z tych zewnętrznych formatów posiada w specyfikacji nadmiarowe, nieużywane domyślnie miejsce, które w zamyśle miało umożliwić techniczne rozwiązania tego typu — czyli osadzanie wewnątrz plików dodatkowych, nieprzewidzianych wcześniej metadanych.

3.2.5. Przykładowa analiza metadanych *EXIF*

Poniżej (Listing 1) znajduje się przykład metadanych *EXIF* związanych z geolokalizacją zawartych w prawdziwym zdjęciu wykonanym za pomocą *smartfona* (model *Samsung Galaxy SIII Mini*). Wszystkie widoczne dane zostały zapisane podczas uwieczniania fotografii z domyślnymi ustawieniami systemowej aplikacji służącej do robienia zdjęć. W załączniku do pracy (Zał. B) znajduje się pełny zrzut metadanych *EXIF* tego samego zdjęcia.

Rys. 3.4. Przykładowe analizowane zdjęcie miasta nocą.

[9]Format *EXIF* w założeniach miał być powszechnie używany także w połączeniu z plikami dźwiękowymi, lecz nie przyjął się jako standard.

```
exif:GPSAltitude: 0/1
exif:GPSAltitudeRef: 0
exif:GPSDateStamp: 2013:08:10
exif:GPSInfo: 750
exif:GPSLatitude: 50/1, 20/1, 42993/1000
exif:GPSLatitudeRef: N
exif:GPSLongitude: 15/1, 55/1, 44190/1000
exif:GPSLongitudeRef: E
exif:GPSMapDatum: WGS 84
exif:GPSProcessingMethod: NETWORK
exif:GPSStatus: V
exif:GPSTimeStamp: 19/1, 20/1, 4/1
exif:GPSVersionID: 2, 2, 0, 0
```

Listing 1. Fragment wyeksportowanych tagów *EXIF* dotyczący współrzędnych *GPS*. Dane te wskazują na miejsce: $(50.345276°N; 15.928942°E)$ — zdjęcie pochodzi więc z miasta Jaromer w Czechach.

3.3. Analizowanie regionów geograficznych

3.3.1. Ogólna problematyka

Problematyka wyszukiwania „ciekawych" miejsc w sposób, którego dotyczy ta praca jest bardzo ogólna i może tak naprawdę służyć do znajdowania takich miejsc na całym świecie — o ile spełnione są pewne warunki konieczne do przeprowadzenia takich badań. Zaproponowany algorytm szukania „ciekawych" lub „nieciekawych" regionów:

1. Wybór głównego rozpatrywanego regionu świata[10], wewnątrz którego szukane mają być „ciekawe" i „nieciekawe" miejsca. Region ten powinien spełniać założenia:
   (a) Musi dzielić się na podregiony, które będą rozpatrywane jako docelowe „ciekawe" i „nieciekawe" miejsca.
   (b) Musi istnieć elektroniczna reprezentacja kształtów podregionów składających się na główny region. Byłaby to pewnego rodzaju elektroniczna mapa głównego regionu i podregionów.
   (c) Muszą istnieć zewnętrzne dane statystyczne dotyczące podregionów, które użyte będą do porównań.
   (d) Musi istnieć wspólny identyfikator podregionów, który używany jest zarówno w elektronicznej mapie, jak i w zewnętrznych statystycznych danych.

[10] Należy zauważyć, że głównym regionem może być nawet cała powierzchnia Ziemii, jeśli spełnione byłyby wszystkie założenia

2. Pobranie z sieci społecznościowej jak największej ilości danych pochodzących z rozpatrywanego głównego regionu wraz ze wszelkimi dodatkowymi potrzebnymi metadanymi.
3. Pogrupowanie pobranych próbek danych pod względem pochodzenia z poszczególnych podregionów.
4. Dopasowanie danych o liczbie próbek w podregionach do odpowiadających im pod względem pochodzenia zewnętrznych danych statystycznych za pomocą wspólnych identyfikatorów.
5. Odfiltrowanie par próbek będących na granicy błędu statystycznego.
6. Znalezienie korelacji pomiędzy dwoma dopasowanymi zbiorami danych.
7. Znalezienie podregionów, z których próbki objawiają największe odchylenie od średnich wyników.

Znalezione podregiony ze względu na swoją wyjątkową popularność lub niepopularność w sieci społecznościowej względem zewnętrznych danych statystycznych są odpowiednio rozumiane jako „ciekawe" lub „nieciekawe".

3.3.2. Ograniczenia

Warto zwrócić uwagę na ograniczające aspekty takiego podejścia do badań. Mogą one być utrudnione albo uniemożliwione gdy:

- Popularność sieci społecznościowej na danym regionie jest zbyt mała — w przypadku niemożliwości zagregowania wystarczającej ilości próbek danych z sieci społecznościowej pochodzących z danego regionu, ilości danych w badanych podregionach mogą oscylować w granicach błędu statystycznego. W takim wypadku, na podstawie niewystarczającej ilości danych niemożliwe jest wysnuwanie praktycznych wniosków.
- Dostępność danych statystycznych o podregionach jest ograniczona — do przeprowadzenia badań w zaproponowany sposób wymagane są dokładne dane statystyczne z zaufanego źródła. W przypadku braku dostępu do dobrej jakości danych nie byłoby do czego przyrównać danych pochodzących z sieci społecznościowej.
- Dane z jednego lub obu tych źródeł są bardzo zakłócone — jeśli w danych wystąpiłyby duże niedokładności, mogłyby przeszkodzić w przeprowadzeniu badań, zakłócając wyniki i sprawiając, że stałyby się one zafałszowane.

4. Realizacja pracy i rozwiązywanie problemów

4.1. Przygotowania do realizacji i podsumowanie założeń

4.1.1. Wyszczególnienie celów pracy

Realizacja pracy inżynierskiej zaczęła się od podsumowania założeń, które definiowały jej cel. Na tej podstawie możliwe było wyszczególnienie dokładnego zakresu pracy i przeprowadzenie analizy środowiska, a następnie dobór narzędzi i implementacja rozwiązania. Podstawowym celem pracy miało być stworzenie zestawu narzędzi (skryptów wykonywalnych i aplikacji) wspomagającego prowadzenie badań opartych na danych z internetowych sieci społecznościowych, który:

1. Umożliwiłby w sposób zautomatyzowany pobranie dużej ilości danych geolokalizacyjnych zawartych w zdjęciach umieszczonych w internecie przez użytkowników wybranej sieci społecznościowej. Dane te miały być połączone z hasłami (tagami), aby pozwolić na filtrowanie próbek bazując na podanych słowach-kluczach.
2. Potrafiłby grupować zebrane dane pod względem powiązań z określonymi hasłami podanymi przez użytkownika.
3. Pozwoliłyby na wyświetlenie pobranych danych na tle mapy rozpatrywanego regionu. W ten sposób użytkownik mógłby w bardzo intuicyjny sposób badać rozkład próbek na badanym obszarze i wstępnie ocenić jakość zebranych danych.
4. Potrafiłby grupować zebrane dane pod względem pochodzenia z konkretnych regionów geograficznych.
5. Zapewniłyby możliwie automatyczny sposób na dopasowanie danych pobranych z sieci społecznościowej do danych statystycznych z zewnętrznego źródła.
6. Pozwoliłby na eksport dopasowanych danych do uniwersalnego formatu *CSV*[11], który będzie mógł być dalej przetwarzany w profesjonalnych narzędziach służących do analizy danych.

[11] *CSV* (ang. „*Comma Separated Values*") — prosty tekstowy format pliku pozwalający przechowywać dane o naturze tabelarycznej. Kolejne kolumny oddzielone są ustalonym separatorem (w zależności od szczegółowego formatu zwykle separatorami są przecinek lub średnik), a kolejne wiersze tabeli znajdują się po prostu w kolejnych wierszach pliku tekstowego.

7. Pozwoliłby na eksport danych geolokalizacyjnych o zdjęciach zebranych z sieci społecznościowej w postaci zrozumiałej przez większość narzędzi do wyświetlania map. Ze względu na swoją popularność miał to być format *Shapefile*[12]. Format ten jest najpopularniejszym standardem jeśli chodzi o przechowywanie danych przestrzennych w postaci plików, obsługiwanym przez większość aplikacji przeznaczonych do prowadzenia badań danych przestrzennych.

W ramach pracy należało także przedstawić przykład użycia stworzonych narzędzi przeprowadzając przykładową analizę zebranych danych, udowadniając tym samym ich działanie i potencjalną przydatność.

4.1.2. Doprecyzowanie założeń — wybór analizowanego regionu

Problematyka pracy w takim sformułowaniu okazała się wciąż zbyt ogólna i wymagała doprecyzowania przed przystąpieniem do technicznej realizacji zadania. Koniecznym było zrobienie pewnych dodatkowych założeń szczegółowo zawężających zakres pracy. Podstawowym, kluczowym aspektem pracy, który wymagał doprecyzowania było wyszczególnienie obszaru geograficznego, którego badanie miały umożliwiać tworzone narzędzia. Mimo, że samo zagadnienie badania regionów geograficznych w założony sposób jest bardzo ogólne i może być przeprowadzone w kontekście właściwie każdego miejsca na Ziemii, techniczne ograniczenia tego zagadnienia opisane w rozdziale 3.3 wymagały podjęcia takiej kluczowej decyzji już na samym początku prac.

Wybór obszaru, którego badanie wspomagać miały projektowane narzędzia, niósł za sobą szereg implikacji związanych z technicznymi kwestiami. Jako, że do przeprowadzania badań wybranego regionu w założony sposób wymagane były dane statystyczne jego dotyczące, wybór obszaru wiązał się bezpośrednio z ograniczeniem możliwości wyboru dostawcy tych danych. Trzeba było wziąć także pod uwagę, że wybrany obszar musiał dzielić się na podregiony (rozpatrywane później jako podstawowe jednostki, wśród których wyszukiwane miały być „ciekawe miejsca") identyfikowane pewnego rodzaju oficjalnym identyfikatorem, który potrzebny był w celu dopasowania danych pobranych z sieci społecznościowej oraz danych statystycznych z zewnętrznego źródła. Musiały istnieć także elektroniczne mapy dokładnie opisujące rozpatrywany obszar i jego podregiony.

[12] *Shapefile* — popularny w zastosowaniach przestrzennych format grafiki wektorowej pozwalający przechowywać informacje o przestrzennych obiektach. Jest to otwarty standard regulowany i utrzymywany przez *Environmental Systems Research Institute*. Pliki w tym formacie standardowo oznacza się rozszerzeniem „*.shp*".

Ze względu na wymagania dotyczące jednolitego sposobu identyfikacji podregionów ewentualnego wybranego obszaru oraz na wymóg istnienia dodatkowego, dokładnego i zaufanego źródła danych statystycznych, a także powszechnego i nieskrępowanego dostępu do internetu, wybór głównego obszaru badań postanowiłem zawęzić do wyboru jednego spośród rozwiniętych europejskich krajów. Naturalnym pierwszym intuicyjnym wyborem było dla mnie rozpatrzenie Polski jako głównego obszaru do przeprowadzania analiz.

4.1.3. Podział administracyjny Polski

Z wstępnej analizy środowiska wynikało, że Polska spełnia wszystkie założenia potrzebne do przeprowadzenia planowanych badań:

1. Polska dzieli się na oficjalne podregiony — obecnie w Polsce obowiązuje trójstopniowy podział administracyjny:
   (a) i stopień — 16 województw;
   (b) II stopień — 314 powiatów;
   (c) III stopień — 2479 gmin:
      - 306 miejskich — gminy zawierające się w administracyjnych granicach miasta;
      - 602 miejsko-wiejskich — gminy składające się z miasta i kilku wsi;
      - 1571 wiejskich — gminy nie zawierające żadnego miasta;
2. W Polsce działa *Główny Urząd Statystyczny* (w skrócie *GUS*), czyli centralny organ administracji rządowej zajmujący się zbieraniem i udostępnianiem informacji statystycznych na temat większości dziedzin życia publicznego i niektórych stron życia prywatnego obywateli kraju. Udostępnia on publicznie i bezpłatnie dokładne dane statystyczne dotyczące poszczególnych regionów kraju przeznaczone do dowolnego użytku.
3. W Polsce w oficjalnych statystycznych badaniach używane są głównie dwa sposoby identyfikacji regionów:
   (a) Rejestr *TERYT* (*Krajowy Rejestr Urzędowy Podziału Terytorialnego Kraju*) — jawny rejestr urzędowy podziału terytorialnego kraju prowadzony przez *Główny Urząd Statystyczny*. Jego identyfikatory stanowią obowiązujący standard identyfikacji terytorialnej dla organów prowadzących urzędowe rejestry i systemy informacyjne administracji publicznej. Pozwalają także innym rejestrom i systemom odnoszącym się do polskich jednostek terytorialnych integrację danych gromadzonych w tych systemach — idealnie wpasowują się więc w rolę wymaganego wspólnego identyfikatora.

(b) Klasyfikacja *NTS* (*Nomenklatura Jednostek Terytorialnych do Celów Statystycznych*) — klasyfikacja jednostek terytorialnych, stosowana przede wszystkim przez *Główny Urząd Statystyczny* dla celów statystyki regionalnej. Odpowiada ona *Klasyfikacji Jednostek Terytorialnych do Celów Statystycznych* (*NUTS*), obowiązującej w krajach Unii Europejskiej.

Oba sposoby identyfikacji są powszechnie używane przez *Główny Urząd Statystyczny* jako identyfikator jednostek terytorialnych w udostępnianych opracowaniach statystycznych.

Rys. 4.1. Mapa Polski na poziomie klasyfikacji *NTS 5* (zaznaczone są granice wszystkich gmin). *Źródło:* `http://stat.gov.pl/statystyka-regionalna/jednostki-terytorialne/nomenklatura-nts/nts-5-2686/`

Ze względu na to, że Polska spełnia wszystkie założenia stawiane przed potencjalnym badanym obszarem, a dodatkowo była dla mnie intuicyjnie pierwszym wyborem — między innymi ze względu na moje pochodzenie, a więc i zrozumienie panujących na jej terytorium warunków i zjawisk — założyłem, że na potrzeby projektowania pierwszej wersji narzędzi ograniczę obszar możliwego do analizy świata do terytorium Polski. Podstawową jednostką administracyjną, której dotyczyć miały badania miała być gmina, ze względu na jej stosunkowo mały rozmiar w skali całego

kraju, oraz na powszechny dostęp do danych statystycznych pogrupowanych na tym poziomie.

Warto podkreślić, że ewentualna rozbudowa możliwości planowanych narzędzi o możliwość analizy innych krajów miała być w założeniu stosunkowo prosta i niewymagająca dużych nakładów pracy. Zmiana taka miała ograniczyć się do dostosowania tworzonej aplikacji do użycia innej mapy z badanymi regionami, i innej nazwy identyfikatorów regionów. Używane zewnętrzne dane statystyczne oczywiście też musiałyby pochodzić z innego źródła, ale to — dzięki planowanemu użyciu uniwersalnych formatów wejściowych — w ogóle nie wymagałoby zmian w programie.

4.2. Wybór bazowych technologii

4.2.1. Język programowania

Po ustaleniu podstawowych założeń pracy należało wybrać technologię implementacji rozwiązania. Zestaw narzędzi miał być tworzony z myślą o jego użyciu na komputerach osobistych klasy PC. Podstawowym dylematem, definiującym kierunek rozwoju pracy było wybranie języka programowania, w którym napisana miała zostać główna aplikacja. Pod uwagę brałem głównie dwa języki: *Java* oraz *C++*. Znajomość obu technologii wyniosłem ze studiów, znałem więc plusy i minusy obu rozwiązań. Oba rozpatrywane przeze mnie języki realizowały założenia paradygmatu obiektowego, w którym wszystkie podstawowe elementy, z których składają się programy, są obiektami. Mimo to, między technologiami tymi występują znaczące różnice.

Największą różnicą pomiędzy nimi jest metoda kompilacji i uruchamiania programów w nich napisanych. Język *C++* (podobnie jak jego prekursor — język *C*) kompilowany jest prosto do kodu maszynowego, którego instrukcje są bezpośrednio wykonywane przez procesor. Programy napisane w języku *Java* kompilowane są do bajtkodu, czyli reprezentacji kodu używanej przez maszynę wirtualną. Maszyna wirtualna jest więc „pośrednikiem" pomiędzy instrukcjami zapisanymi przez programistę, a instrukcjami wykonywanymi przez procesor. Dzięki temu programy napisane w języku *Java* można wykonywać na każdym systemie, na którym można uruchomić wirtualną maszynę *JVM*[13]. Maszyna ta wspierana jest na wszystkich dużych systemach operacyjnych.

C++ jest rozszerzeniem proceduralnego języka *C*, i jest z nim w dużej mierze wstecznie kompatybilny, co oznacza zachowanie pewnych archaicznych rozwiązań, które potrafią niepotrzebnie skomplikować pisane programy i sprawić, że są one trudniejsze w późniejszym utrzymaniu. Twórcy języka *Java*, tworząc go od zera,

[13] *JVM* (ang. *„Java Virtual Machine"*) — *Maszyna Wirtualna Javy*, czyli wieloplatformowe środowisko uruchomieniowe zdolne do uruchamiania programów w języku *Java*

przykładali bardzo dużą wagę do usprawnień i ułatwień sprawiających, że utrzymanie i późniejszy rozwój tworzonych aplikacji jest znacznie prostsze, niż utrzymanie aplikacji tworzonych w języku *C++*. Przykładem mechanizmu usprawniającego programowanie w języku *Java* może być tak zwany *Garbage Collector*, który zapewnia obsługę zwalniania nieużywanej już przez obiekty pamięci, zwalniając z tego obowiązku programistę — wyeliminowując w standardowych sytuacjach tak zwane „wycieki pamięci".

Jednym z założeń, które miałem podczas przygotowań do realizacji celów pracy, było stworzenie możliwie jak najbardziej uniwersalnego i wygodnego rozwiązania. Projektowane narzędzia miały pomagać w prowadzeniu badań, a w planowanym procesie analizy pojawiało się użycie narzędzi służących do analizy, do których użytkownik jest już przyzwyczajony, i które dobrze zna. Nie chciałem więc ograniczać możliwości prowadzenia badań do jednego, konkretnego systemu operacyjnego, ani do użycia konkretnych, wybranych przeze mnie narzędzi. Narzędzia miały służyć do prowadzenia badań, a więc ich interaktywność i ogromna wydajność nie były priorytetem. Dodatkowo, ogromna ilość darmowych bibliotek napisanych specjalnie do użytku z technologiami opartymi na języku *Java* oraz bardzo łatwo dostępne wsparcie techniczne (i świetna dokumentacja w stylu *Javadoc*) spowodowały ostatecznie, że do realizacji założonych celów zdecydowałem się użyć właśnie tego języka.

4.2.2. Technologie wspomagające zarządzanie projektem

Zgodnie z dobrymi praktykami zarządzania projektem oraz dla zwiększenia bezpieczeństwa, do prowadzenia projektu postanowiłem wykorzystać system kontroli wersji *Git*. Jest to bardzo popularne i uniwersalne narzędzie, które pozwala kontrolować wprowadzane w kodzie źródłowym zmiany, umożliwiając w każdej chwili przywrócenie kodu źródłowego do wybranego przeszłego zapisanego stanu w bardzo prosty i kontrolowany sposób. Bardzo wskazane jest przechowywanie kopii zapasowej kodu źródłowego tworzonych aplikacji na zewnętrznych szyfrowanych repozytoriach, aby zabezpieczyć projekt przed możliwością straty postępów w wyniku awarii. Narzędzie *Git* powszechnie używane jest także do wspomagania pracy zespołowej umożliwiając proste łączenie kodu źródłowego stworzonego przez wielu programistów — choć w tym projekcie nie miałem okazji skorzystać z tej funkcjonalności.

Aby uniezależnić kod programu i jego budowanie od konkretnego środowiska programistycznego, a także ułatwić korzystanie z dodatkowych bibliotek programistycznych, skorzystałem z systemu budowania *Apache Maven*. Narzędzie to umożliwia zdefiniowanie sposobu budowania aplikacji, ułatwia tworzenie środowiska programistycznego oraz pozwala na zadeklarowanie zależności tworzonego projektu od

zewnętrznych bibliotek, które przez narzędzie ściągane są automatycznie z publicznych repozytoriów i dołączane do projektu, bardzo ułatwiając cały proces. Projekt oparty na tej technologii musi mieć ściśle zdefiniowaną, standardową hierarchię katalogów, co ułatwia rozeznanie się w projekcie osobom początkowo z nim nawet nie związanych. Wszystkie opcje i zależności definiowane są w plikach *XML* o ustandaryzowanej strukturze. Dzięki temu możliwa jest prosta kontrola nad wykorzystanymi bibliotekami i procesem budowania aplikacji.

4.3. Architektura rozwiązania

4.3.1. Ogólny zamysł

Tworzona aplikacja w zamyśle miała być narzędziem głównie konsolowym, pracującym w trybie wsadowym, a nie interaktywnym. Z tego względu nie wystąpiła potrzeba projektowania zaawansowanego interfejsu użytkownika, a w związku z tym automatycznie udało się ominąć wiele związanych z tego typu rozwiązaniami problemów. Użytkownik końcowy miał więc komunikować się z aplikacją głównie poprzez przekazywane w trybie konsolowym argumenty, z którymi mógł ją wywoływać. Wyjątkiem miała być część aplikacji odpowiedzialna za wyświetlanie zebranych z sieci społecznościowej zdjęć na tle mapy, opisana w podrozdziale 4.7 — w tym przypadku konieczne było umożliwienie wyświetlenia interaktywnego okienka graficznego.

Na wstępnym etapie prac postanowiłem, że realizowane przeze mnie narzędzia mają składać się z:

- Aplikacji napisanej w języku *Java*, która miała stanowić główną część pracy, oraz zapewniać większość funkcjonalności. To ona ostatecznie miała komunikować się z użytkownikiem, zbierać, grupować oraz udostępniać dane, a także umożliwiać wyświetlenie zebranych danych na tle mapy rozpatrywanego regionu.
- Bazy danych przechowującej zebrane dane. Baza miała umożliwiać efektywne i wydajne grupowanie i filtrowanie potencjalnie bardzo dużych ilości danych.
- Zestawu skryptów przygotowujących środowisko uruchomieniowe aplikacji — ściągających i instalujących wszystkie potrzebne narzędzia, a w szczególności instalujących bazę danych oraz jej nakładki, a także przygotowujące struktury bazy danych do działania aplikacji.

4.3.2. Architektura głównego programu w języku *Java*

Podczas projektowania oraz implementowania rozwiązania kluczowe dla mnie było umożliwienie względnie prostej rozbudowy funkcjonalności powstałej w ramach pracy aplikacji napisanej w języku *Java*, która stanowiła najważniejszą część pracy. Ważne było też zapewnienie łatwości jej utrzymania. W tym celu dużo czasu poświęciłem na przemyślenie architektury tworzonego programu w myśl działania zgodnie z dobrymi praktykami inżynierskimi i skorzystałem z wielu przydatnych wzorców projektowych i gotowych bibliotek. Mądre projektowanie aplikacji było ważne również z tego względu, że podczas pracy nad programem wielokrotnie zmieniałem rozwiązania technologiczne i implementację poszczególnych jego fragmentów. Było to związane z charakterem pracy, który sprawiał, że ciągle spotykałem się z nowymi dla mnie technologiami i problemami. Kod źródłowy wielokrotnie przechodził refaktoryzację[14] spowodowaną przez zauważane przeze mnie podczas pracy coraz lepsze i ciekawsze rozwiązania, które mogłem wykorzystać w walce z napotkanymi problemami.

Z tych powodów w wielu miejscach w aplikacji klasy zarządzające poszczególnymi modułami aplikacji komunikują się ze sobą tylko przy użyciu interfejsów. Umożliwia to w razie potrzeby prostą podmianę implementacji poszczególnych fragmentów kodu, praktycznie przezroczystą dla całej reszty aplikacji. Niektóre z technologii, które opisane są w kolejnych rozdziałach także zdecydowanie przyczyniły się do odizolowania od siebie konkretnych implementacji, co okazało się w niektórych przypadkach zbawienne.

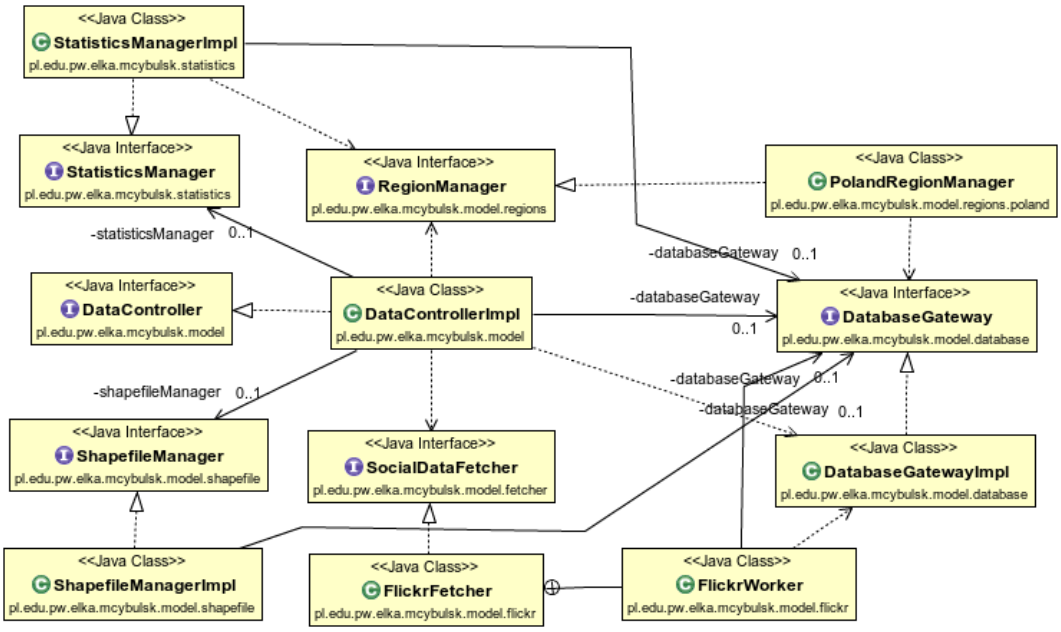
Na Rys. 4.2 znajduje się diagram *UML* przedstawiający główne interfejsy oraz hierarchię klas aplikacji, i połączenia między nimi. Aplikacja podzielona została na moduły logicznie odpowiadające za różne części funkcjonalności:

- Moduł centralny, potrafiący komunikować się z pozostałymi modułami — interfejs *DataController*. Tworzy warstwę-„tłumacza" pomiędzy komendami użytkownika a wyspecjalizowanymi obiektami wykonującymi pewne procesy na modelu danych.

- Moduł odpowiadający za zbieranie danych z określonej sieci społecznościowej — interfejs *SocialDataFetcher*. Jego implementacje tworzone są przez fabrykę[15] *SocialDataFetcherFactory*, która w zależności od podanego przez użytkownika aplikacji kodu sieci społecznościowej (np. „*flickr*") zwraca odpowiednią implementację „zbieracza danych". Dzięki takiemu mechanizmowi dodawanie

[14] *Refaktoryzacja* — proces wprowadzania zmian w projekcie informatycznym, w wyniku którego nie zmienia się zasadniczo funkcjonalność tworzonej aplikacji.

[15] W programie postanowiłem wykorzystać tak zwaną *Prostą Fabrykę* (ang. „*Simple Factory*"). W ten sposób nazywa się klasę, która w zależności od sytuacji potrafi zwrócić różne implementacje ustalonego interfejsu.



Rys. 4.2. Diagram *UML* przedstawiający hierarchię podstawowych klas i połączeń między nimi.

obsługi kolejnych sieci społecznościowych sprowadza się do zadeklarowania nowego kodu sieci społecznościowej w zwykłym typie *enum SocialDataFetcherCode*, stworzenia nowej implementacji interfejsu *SocialDataFetcher*, a następnie dodanie obsługi tworzenia obiektów nowego typu do fabryki. Fabryka potrafi wypisać listę wszystkich wspieranych przez siebie sieci społecznościowych.

- Moduł odpowiedzialny za operacje związane z konkretnym badanym regionem (na przykład za stworzenie listy regionów, z których „zbieracz danych" powinien pobierać dane) — interfejs *RegionManager*. Jego implementacje tworzone są przez fabrykę *RegionManagerFactory*. Mechanizm obsługujący tworzenie konkretnych obiektów implementujących ten interfejs jest analogiczny jak w przypadku interfejsu *SocialDataFetcher*. Fabryka *RegionManagerFactory* zwraca odpowiednią implementację interfejsu w zależności od regionu podanego przez użytkownika przy uruchamianiu programu (np. „poland").
- Moduł odpowiedzialny za połączenie z bazą danych — na ten moduł składają się interfejsy:
  - Interfejs *DatabaseGateway* — odpowiadający za znakomitą większość operacji na bazie danych. Są to ogólne operacje, niezwiązane z konkretnymi dostawcami danych ani analizowanymi regionami.
  - Interfejs *RegionDatabaseHelper* — odpowiadający za operacje na bazie danych związane z konkretnym analizowanym regionem. W zamyśle jest

mocno sprzężony z interfejsem *RegionManager* i służy do wydzielenia fragmentów obsługi bazy danych zależnych od konkretnego analizowanego regionu świata. Obiekty implementujące interfejs tworzone są analogicznie jak w przypadku interfejsów *SocialDataFetcher* oraz *RegionManager* za pomocą fabryki — w tym przypadku za pomocą klasy *RegionDatabaseHelperFactory*.

- Moduł dopasowujący zebrane dane społecznościowe do zewnętrznych danych statystycznych — interfejs *StatisticsManager*. Udostępnia funkcje dopasowania zebranych danych do danych zewnętrznych w odpowiednim formacie bazującym na plikach *CSV* opisanym w rozdziale 4.9.1, a następnie wyeksportowanie dopasowanych danych do pliku w bardzo podobnym formacie, możliwym do prostego przetworzenia w zewnętrznych aplikacjach służących do analizy danych.
- Moduł wyświetlający zebrane dane geolokalizacyjne na tle mapy — klasa *MapViewer*. Potrafi wyświetlić proste interaktywne okienko stworzone przy pomocy biblioteki *GeoTools*[16], w którym wyświetlane są dane zebrane z sieci społecznościowej na tle map z projektu *OpenStreetMap*[17].

W zaplanowanej przeze mnie architekturze programu bardzo widoczne jest poleganie na obiektach-fabrykach, które w zależności od życzenia użytkownika umożliwiają pracę na różnych implementacjach tego samego interfejsu, zapewniając różne metody zbierania i przetwarzania danych. Plusem takiego podejścia jest to, że dodanie obsługi nowej funkcjonalności (na przykład dodanie obsługi nowego serwisu społecznościowego) jest bardzo proste i szybkie, a co najważniejsze — działa bez zmiany żadnych mechanizmów w reszcie programu.

Podział aplikacji na tego typu moduły (większość modułów znajduje się w specjalnie dla siebie wydzielonych *pakietach*) pozwala też na proste rozeznanie się w strukturze programu. Kod źródłowy klas odpowiadających za podobne funkcjonalności znajduje się najczęściej bardzo blisko siebie. Dodatkowo, rozmieszczanie klas w *pakietach* zwiększa ich bezpieczeństwo — pola i metody z domyślnym modyfikatorem dostępu są niewidoczne poza *pakietem*, w którym znajduje się ich klasa.

[16] *GeoTools* — biblioteka *open-source* stworzona dla języka *Java*, która jest bardzo pomocna przy tworzeniu aplikacji opartych o badania geograficzne z użyciem wszelkiego rodzaju elektronicznych map. Jest zgodna ze standardami *OGC* (*Open Geospatial Consortium*). Strona projektu: `http://www.geotools.org/`

[17] *OpenStreetMap* — projekt społeczności internetowej mający na celu stworzenie darmowej, swobodnie dostępnej mapy całej kuli ziemskiej. Tworzona mapa jest edytowalna przez zarejestrowanych użytkowników. Serwis udostępnia możliwość eksportu wybranych wycinków map. Adres projektu: `http://www.openstreetmap.org/`

Struktura aplikacji jest bardziej złożona, niż tylko wymienione w liście powyżej moduły, ale to one stanowią „szkielet" i bazę aplikacji. Tworząc narzędzia, zapewniłem przykładową implementację każdego z zaprojektowanych interfejsów, co pokazuje, że architektura przeze mnie zaprojektowana może być z powodzeniem wygodnie wykorzystana i rozwijana.

4.4. Przechowywanie danych

4.4.1. Przechowywanie danych geograficznych w bazie danych

Standardowo, bazy danych nie są kojarzone z operacjami przestrzennymi, i zazwyczaj używane są do przechowywania prostych typów danych takich jak ciągi znaków oraz liczby, oraz do przeprowadzania operacji na tego rodzaju typach danych. Istnieją jednak bazy danych specjalnie przystosowane do obsługi danych geometrycznych lub geograficznych — bazy danych takie nazywane są *przestrzennymi bazami danych* (ang. „*spatial database*"). Organizacja *Open Geospatial Consortium*[18] opracowała zbiór otwartych standardów dotyczących przechowywania i przetwarzania danych i usług przestrzennych, z których niektóre opisują dodawanie przestrzennych funkcjonalności do systemów bazodanowych.

Przestrzenne bazy danych są zoptymalizowane do składowania oraz przetwarzania danych powiązanych z przestrzennymi obiektami, takimi jak punkty, linie i wielokąty. Bazy takie, poza standardowymi zapytaniami *SQL* pozwalają również wykonywać *zapytania przestrzenne*, które:

- używają geometrycznych typów danych takich jak punkty, linie i wielokąty;
- uwzględniają przestrzenne relacje między obiektami reprezentowanymi przez te typy;

Innymi słowy, tego rodzaju bazy danych potrafią obsługiwać zapytania takie jak na przykład „znajdź wszystkie punkty leżące w obszarze o nazwie »Warszawa«" — oczywiście zapisane w odpowiednim, obsługiwanym dialekcie języka *SQL*. Zazwyczaj takie bazy danych obsługują bardzo wiele funkcji związanych także z wyznaczaniem miar (na przykład obliczaniem pola zadanego obszaru), tworzeniem nowych obiektów przestrzennych, czy określaniem relacji między istniejącymi obiektami (na przykład relacja wzajemnego zawierania się, bądź odległości pomiędzy obiektami). Wszystkie te funkcje przewidziane są w standardach *OGC*,

[18] *Open Geospatial Consortium* (w skrócie *OGC*) — międzynarodowa organizacja non-profit zrzeszająca 509 (stan na dzień 26. stycznia 2015, źródło: `http://www.opengeospatial.org/ogc`) firm, agencji rządowych oraz uczelni wyższych.

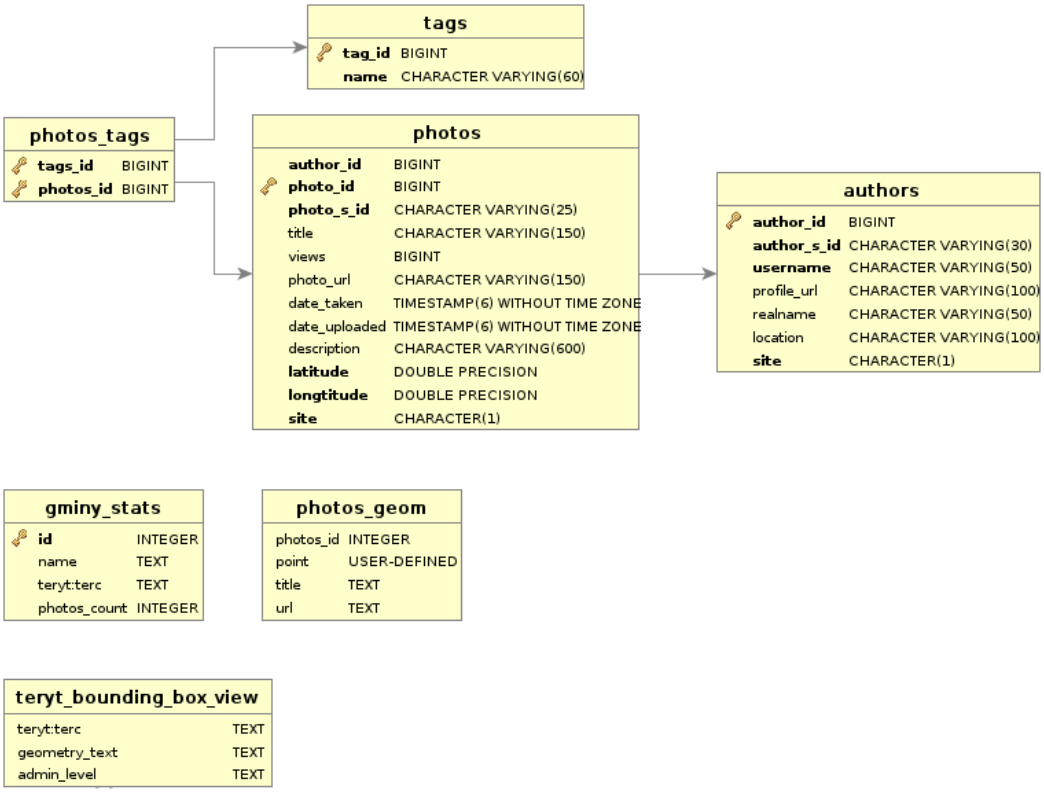
Użycie standardowych indeksów wykorzystywanych w zwyczajnych relacyjnych bazach danych nie daje zadowalających rezultatów w przypadku operacji na danych przestrzennych, dlatego *przestrzenne bazy danych* obsługują zazwyczaj typy indeksów specjalnie przystosowane do pracy z tego rodzaju danymi. Indeksy te zazwyczaj polegają na mechanizmie porównywania tak zwanych „bounding-boksów" obiektów przestrzennych, czyli prostokątów zawierających w całości rozpatrywany obiekt. W mechanizmie tym wykorzystywany jest fakt, że zbadanie przecinania się dwóch prostokątów jest nieporównywalnie tańsze obliczeniowo, niż wykonanie takiego badania na bardzo złożonych wielokątach. Bazując na tym mechanizmie, indeksy przestrzenne budowane są zazwyczaj w oparciu o pewnego rodzaju drzewiastą strukturę danych (na przykład *B-drzewo*). Tak jak w przypadku standardowych indeksów, indeksy przestrzenne potrafią wielokrotnie przyspieszyć wykonywanie się obliczeń i porównań w bazie danych.

Na potrzeby wykonywanych narzędzi postanowiłem wykorzystać rozszerzenie relacyjno-obiektowej bazy danych *PostgreSQL* o nazwie *PostGIS*. Narzędzie to udostępniane jest na zasadach *open-source* i dodaje możliwość przechowywania danych przestrzennych w zwykłej bazie danych. Przechowywanie danych przestrzennych polega na zadeklarowaniu pewnych kolumn jako kolumn przechowujących obiekty geometryczne. Mogą to być bardzo ogólne typy, jak na przykład typ *GEOMETRY* — w kolumnie takiej można umieszczać obiekty takie jak punkty, linie, oraz nawet bardzo złożone wielokąty — ale także bardziej konkretne typy, jak na przykład typ *POINT*, który odpowiada zwykłemu punktowi na płaszczyźnie (istnieją także typy przestrzenne obsługujące trzy wymiary).

4.4.2. Schemat bazy danych

Na Rys. 4.3 znajduje się diagram głównych tabel bazy danych wykorzystywanej przez aplikację. Dla uproszczenia nie zawierają się na nim tabele powstałe w wyniku importu danych geograficznych z projektu *OpenStreetMap* — temat opisany jest szerzej w rozdziale 4.7. Podstawowe tabele tworzonego systemu dzielą się na:

1. Tabele przeznaczone do zbierania i filtrowania danych, a także wszelkich analiz standardowych (nie przestrzennych). Nie zawierają one żadnych kolumn przestrzennych:
    - *PHOTOS* — tabela przeznaczona do przechowywania informacji o poszczególnych zdjęciach. Zawiera zestaw atrybutów opisujących zdjęcia, takich jak tytuł, opis, czy data wykonania. Zawiera też dwie kolumny służące do przechowywania w podstawowy sposób danych o geolokalizacji zdjęć — są to kolumny *LATITUDE* oraz *LONGITUDE*, odpowiednio dla szerokości oraz długości geograficznej.



Rys. 4.3. Diagram podstawowych tabel tworzonego systemu (bez tabel zaimportowanych z projektu *OpenStreetMap*).

- *AUTHORS* — tabela przeznaczona do przechowywania podstawowych informacji o autorach zdjęć, na wypadek gdyby wystąpiła potrzeba włączenia autorów zdjęć do analizy danych (na przykład do filtrowania zdjęć od wybranych autorów). Zawiera jedynie podstawowe atrybuty opisujące osoby, aby umożliwić podstawowe filtrowanie zdjęć od nich pochodzących. Jest połączona z tabelą *PHOTOS* relacją *„wiele do jednego”* (każdy autor może posiadać wiele zdjęć, a każde zdjęcie tylko jednego autora).
- *TAGS* — tabela przeznaczona do przechowywania pojedynczych haseł (tagów) związanych ze zdjęciami. Jest połączona z tabelą *PHOTOS* relacją *„wiele do wielu”* (każde zdjęcie może być opisane wieloma tagami, a każdy tag może opisywać wiele zdjęć) za pomocą dodatkowej tabeli pośredniczącej *PHOTOS_ TAGS*.
- *PHOTOS_ TAGS* — bardzo prosta tabela wspierająca związek *„wiele do wielu”* pomiędzy tabelami *PHOTOS* i *TAGS*.

2. Tabele i widok przeznaczone do analizy przestrzennej zebranych danych. Wymagają zsynchronizowania z tabelami z punktu pierwszego — są napełniane przez program odpowiednimi, odfiltrowanymi danymi i pozwalają na

wydajne wyświetlanie danych i ich przetwarzanie w kontekście przestrzennym. Zawierają obiekty przestrzenne lub są z nimi bezpośrednio powiązane.

- *PHOTOS_GEOM* — podstawowa pomocnicza tabela aplikacji zwierająca obiekty przestrzenne. Służy do przedstawienia wybranych zdjęć z tabeli *PHOTOS* (na przykład powiązanych z konkretnymi tagami, bądź pochodzące tylko z wybranego regionu) w postaci geograficznej. Służy do tego kolumna *POINT* typu *point*, która przechowuje obiekty informujące o geolokalizacji konkretnego zdjęcia. Tabela nie posiada de facto klucza ani konkretnych identyfikatorów, ponieważ jej przeznaczeniem jest po prostu pomocnicze przechowywanie wybranych danych w odpowiedniej formie, aby umożliwić ich analizę przestrzenną.
- *GMINY_STATS* — tabela agregująca zdjęcia pod względem pochodzenia z poszczególnych gmin. Zawiera proste atrybuty opisujące gminy, czyli nazwę oraz identyfikator *TERYT*, oraz liczbę zdjęć pochodzącą z danego regionu. Tabela ta zasilana jest przez aplikację na podstawie danych z tabeli *PHOTOS_GEOM* oraz danych zaimportowanych z projektu *OpenStreetMap*. Baza danych wykonuje zapytania na zasadzie pobierania wszystkich punktów z tabeli *PHOTOS_GEOM* zawierających się w obszarze danej gminy.
- *TERYT_BOUNDING_BOX_VIEW* — widok zwracający techniczne informacje na temat regionów opisanych identyfikatorami *TERYT* na potrzeby aplikacji. W szczególności zwraca opis tak zwanych „bounding-boksów" zawierających powierzchnie poszczególnych regionów (w kolumnie *GEOMETRY_TEXT*).

4.4.3. Techniczne połączenie aplikacji z bazą danych — technologia *Mapowania obiektowo-relacyjnego*

Pracując nad aplikacją chciałem jak najbardziej uniezależnić ją od konkretnego dostawcy bazy danych, oraz od ręcznego tworzenia zapytań w języku *SQL* i późniejszego „przepisywania" wyników z obiektów typu *ResultSet* do poszczególnych tworzonych przez siebie ręcznie obiektów *POJO*[19] — często dodatkowo mapując bazodanowe typy danych na typy używane przez język *Java*. Praca w ten „klasyczny" sposób jest bardzo niewygodna i podatna na błędy, a dodatkowo bardzo mocno uzależnia tworzoną aplikację od konkretnego dostawcy bazy danych, a czasami nawet i jej konkretnej wersji — podejście to zupełnie nie nadaje się do tworzenia dużych, zaawansowanych projektów. Jest to oczywiście możliwe, ale najczęściej niepotrzebnie

[19] *POJO* (ang. „*Plain Old Java Object*") — potoczna nazwa zwyczajnych, prostych obiektów w języku *Java*. Nazwa ta sugeruje, że obiekty te nie są w żaden sposób specjalne (na przykład: nie korzystają z zaawansowanych funkcji języka, ani nie są częścią żadnego *frameworku*) i zazwyczaj służą do przechowywania prostych informacji w obiektowej formie.

komplikuje projekt, gdyż można posłużyć się dużo wygodniejszymi i pewniejszymi rozwiązaniami technicznymi.

W tej sytuacji idealna okazuje się być technika *Mapowania obiektowo-relacyjnego* (ang. „*Object-Relational Mapping*", w skrócie *ORM* lub *O-R Mapping*). Polega ona na automatycznym odwzorowywaniu obiektowej architektury systemu informatycznego na bazę danych o relacyjnym charakterze przy użyciu wyspecjalizowanych w tej kwestii narzędzi. Istnieje szereg technologii opartych na tego typu założeniach, oferujących programistom wygodny i pewny dostęp do bazy danych za pomocą abstrakcyjnych obiektów, całkowicie odcinających programistów od „ręcznej" pracy z bazą danych.

Na potrzeby tworzonej aplikacji zdecydowałem się użyć darmowego narzędzia *open-source Apache Cayenne* (strona internetowa projektu: `https://cayenne.apache.org/`). Narzędzie to pozwala na automatyczne rozpoznanie schematu bazy danych za pomocą *wstecznej inżynierii*[20], a następnie stworzenie modelu klas języka *Java* odpowiadających tabelom w bazie danych i wyeksportowanie go do gotowych plików z kodem źródłowym, od razu działających z tworzonym projektem. Obiekty te zawierają od razu odpowiednio dopasowane typy danych natywne dla języka *Java*, których wartości są automatycznie przypisywane przy pobieraniu obiektów z bazy danych za pomocą specjalnie przystosowanych do tego klas *Apache Cayenne*. Techniczne szczegóły mapowania można podejrzeć w automatycznie wygenerowanym pliku *XML* zawierającym konfigurację połączenia.

Jest to rozwiązanie bardzo wygodne i uniwersalne. Programista nie musi martwić się o dostawcę bazy danych, ani o jej konkretną wersję, ponieważ odpowiedzialność za obsługę połączenia z bazą danych, oraz konstruowanie zapytań oddelegowana jest do zewnętrznego narzędzia. Ze względu na to, że jest ono powszechnie używane oraz wspiera wszystkie popularne bazy danych, podczas pracy nad aplikacją można być spokojnym o wszechstronność tworzonych rozwiązań i nie trzeba bać się ewentualnych zmian, które bez używania podobnych technologii mogłyby oznaczać fiasko projektu. Przykładowy fragment kodu źródłowego tworzonej w ramach tej pracy aplikacji korzystający z techniki *O-R Mappingu* znajduje się wewnątrz Listingu 2.

[20] *Wsteczna inżynieria* (lub inaczej *Inżynieria odwrotna*) — technika polegająca na badaniu pewnego rodzaju zasobu (urządzenia, programu komputerowego, bazy danych, itp.) w celu ustalenia jak działa, a także w jaki sposób został zbudowany.

```java
    /**
     * Zwraca liste zdjec o podanym identyfikatorze pobranych z podanej sieci
    spolecznosciowej.
     *
     * @param siteId identyfikator zdjecia z sieci spolecznosciowej
     * @param siteCode identyfikator sieci spolecznosciowej
     * @param context kontekst obiektowy polaczenia z baza danych
     * @return lista zdjec
     */
    private List<Photos> getPhotos(String siteId, String siteCode,
            ObjectContext context) {
        Expression expression = ExpressionFactory.matchExp(
                Photos.PHOTO_SID_PROPERTY, siteId).andExp(
                ExpressionFactory.matchExp(Photos.SITE_PROPERTY, siteCode));
        @SuppressWarnings("unchecked")
        List<Photos> result = context.performQuery(new SelectQuery(
                Photos.class, expression));
        return result;
    }
```

Listing 2. Fragment kodu źródłowego aplikacji odwołującego się do bazy danych — klasa *DatabaseGatewayImpl*.

4.5. Sieć społecznościowa

4.5.1. Wymagania względem sieci społecznościowej

W rozdziale 3.1 opisane zostały ogólne teoretyczne rozważania na temat sieci społecznościowych. Przechodząc jednak do implementacji rozwiązania trzeba było skupić się na bardziej technicznej stronie sytuacji. Analiza celów stawianych przed tworzonymi narzędziami naturalnie doprowadziła do powstania wymagań, które musiała spełniać internetowa sieć społecznościowa, aby nadawała się do wykorzystania w tworzonych narzędziach — serwis tego typu musiał:

1. Umożliwiać darmowy dostęp do informacji — tworzone w ramach pracy narzędzia miały być w założeniu darmowe, anonimowe i uniwersalne. Płatne usługi bardzo utrudniłyby proces tworzenia oprogramowania w ramach pracy inżynierskiej. Ponadto wymóg nakładów finansowych ograniczyłyby też uniwersalność ich późniejszego użytkowania. Dużo trudniej przekonać ludzi do wypróbowania nowej technologii lub rozwiązania, gdy jest ono płatne. Dodatkowo — darmowy dostęp do serwisu zazwyczaj oznacza również jego znacznie większą popularność.

2. Posiadać bogatą społeczność — aby tworzone narzędzia umożliwiały skuteczne badania popularności regionów geograficznych, muszą polegać na dużej ilości danych generowanych przez użytkowników sieci społecznościowej. Danych musi być na tyle dużo, żeby dało się przeprowadzać ich sensowną analizę statystyczną.

3. Udostępniać metadane zdjęć umieszczanych przez użytkowników — cała planowana funkcjonalność tworzonych narzędzi polega na wykorzystaniu danych geolokalizacyjnych zawartych w metadanych plików z obrazami. Niestety — przynajmniej z punktu widzenia architekta tego typu aplikacji — znakomita większość sieci społecznościowych całkowicie usuwa wszelkie metadane zawarte w zdjęciach wysyłanych przez swoich użytkowników.

4. Nie blokować natężonego ruchu sieciowego — zebranie dużej ilości danych z serwisu społecznościowego musi wiązać się z generowaniem ruchu sieciowego większego niż średni. W przypadku serwisów z restrykcyjną polityką ograniczeń agregacja danych staje się bardzo problematyczna.

5. Udostępniać dodatkowe dane — w szczególności wszelkie hasła (tagi), którymi użytkownicy mogą opisywać dane umieszczane przez siebie w sieci społecznościowej, ale także wszelkie dodatkowe informacje takie jak komentarze, oceny, czy opisy. Zbieranie tego typu danych niewielkim kosztem zwiększa wszechstronność tworzonych rozwiązań, tworząc więcej potencjalnych zastosowań dla takich narzędzi.

Dodatkowo, bardzo dużą zaletą przemawiającą za daną siecią społecznościową byłby fakt udostępniania przez nią funkcjonalnego *API* dla programistów.

4.5.2. Wybór sieci społecznościowej

W wyniku analizy środowiska najbardziej odpowiedni do założonych celów okazał się być serwis *Flickr*[21], opisany wraz ze swoimi potencjalnymi konkurentami w rozdziale 3.1.2. Spełnił on wszystkie warunki stawiane w podrozdziale 4.5.1, wraz z funkcjonalnością udostępniania bardzo rozbudowanego i wygodnego *API* dla programistów. Dodatkowo, serwis ten nie prowadzi restrykcyjnej polityki ograniczeń — nie istnieją żadne formalne limity ilości zapytań mozliwych do wykonania przez aplikacje klienckie. Mimo wszystko, w regulaminie usługi istnieje zapis, że serwis rezerwuje sobie prawo do interwencji w przypadku ewidentnych nadużyć. Żadna inna rozpatrywana przeze mnie sieć społecznościowa (analiza została przeprowadzona w rozdziale 3.1.2) nie zbliżyła się do tego serwisu swoim potencjałem dla planowanego zastosowania.

[21]Adres URL serwisu: `https://www.flickr.com/`

4.6. Zbieranie danych

4.6.1. Metody pobierania danych

Pobieranie dużej ilości danych z internetowego serwisu jest zazwyczaj zagadnieniem bardzo problematycznym. Niepisane zasady *netykiety* sugerują aby w jak najmniejszym stopniu wpływać swoimi działaniami na wydajność innych systemów działających w sieci oraz na komfort przeglądania internetu przez innych jego użytkowników. W zależności od metody pobierania danych można postępować zgodnie z tymi zasadami, lub wręcz przeciwnie. W ogólności istnieją dwa główne sposoby automatycznego pobierania danych z serwisów internetowych przez aplikacje specjalnie do tego stworzone (aplikacje typu „*Web crawler*"[22]):

1. Odpytywanie serwisów internetowych o cały kod *HTML* strony, której zawartość chce się poznać — jest to metoda nazywana w języku angielskim „*Web scrapping*".
2. Korzystanie z *API* dla programistów udostępnionego przez serwis internetowy.

Technika „Web scrapping" polega na symulowaniu przez program zachowania zwykłego użytkownika w celu zebrania jak największej ilości danych z serwisu internetowego. Określenie „Web scrapping" w wolnym tłumaczeniu oznacza w języku polskim „internetowe skrobanie". Nazwa tej techniki jest nacechowana tak pejoratywnie nie bez powodu — pobieranie dużej ilości danych tym sposobem generuje niepotrzebnie duży ruch po stronie serwera i jest w porównaniu z ewentualnymi alternatywami metodą bardzo powolną. Wymaga ona pobrania przez robota kodu *HTML* całej strony, a następnie sparsowanie jej i wyszukanie interesujących go informacji. Mimo to, czasem nie ma innej możliwości na przeglądanie zasobów serwisu internetowego, jeśli ten nie udostępnia innych metod ku temu przeznaczonych. Aplikacje-roboty przeszukujące internet w ten sposób zazwyczaj starają się symulować zachowanie zwyczajnego użytkownika internetu, podszywając się pod znaną przeglądarkę i wykonując zapytania z opóźnieniem. Zachowują się tak dlatego, że serwisy internetowe w razie wykrycia tego typu „automatycznego użytkownika" blokują mu dostęp do serwisu, aby ograniczyć zużycie zasobów.

Korzystanie z *API*, czyli interfejsu przygotowanego przez internetowy serwis specjalnie w celu umożliwienia komunikacji ze sobą zewnętrznym systemom, ma z punktu widzenia obu stron same korzyści. Aplikacja-robot może korzystać ze znacznie bardziej efektywnego sposobu dostępu do danych, a serwis internetowy nie

[22] Ang. „*Web crawler*" — rodzaj programu-robota, który automatycznie porusza się po sieci internetowej zbierając dane z napotkanych stron.

jest obciążany niepotrzebnie generowaniem i przesyłaniem niepotrzebnych robotowi fragmentów danych (np. szablonu strony internetowej).

4.6.2. Opis *API* serwisu *Flickr*

Serwis *Flickr* udostępnia bardzo rozbudowane *API* dla programistów, umożliwiające bardzo efektywne wykonywanie zapytań związanych z funkcjonalnością tej sieci społecznościowej. Cała specyfikacja interfejsu znajduje się pod adresem: `https://www.flickr.com/services/api/`. Istnieje możliwość wybrania formatu wejściowego oraz wyjściowego spośród zbioru: *REST*, *XML-RPC* oraz *SOAP*. W przypadku wyjściowych formatów obsługiwane są jeszcze dodatkowo: *JSON* oraz *PHP*. Dla najpopularniejszych języków programowania istnieją implementacje udostępnianego interfejsu, które dostarczane są w postaci bibliotek i zapewniają bardzo wygodną obsługę interfejsu bez dodatkowych nakładów pracy. W tworzonych narzędziach wykorzystałem bibliotekę *flickrj-android* (adres internetowy projektu: `https://code.google.com/p/flickrj-android/`).

Korzystanie z udostępnionego przez *Flickr* interfejsu możliwe jest dopiero, gdy programista posiada klucz dostępu do usługi. Serwis bardzo chętnie daje do niej dostęp, choć dla zastosowań komercyjnych klucz może być płatny. Na potrzeby tworzonej pracy otrzymałem klucz bez żadnych komplikacji.

4.6.3. Opis funkcji *flickr.photos.search*

Najważniejsze w kontekście tworzonej pracy jest to, że interfejs udostępnia funkcje związane z wyszukiwaniem zdjęć dotyczących zadanych haseł z podanego obszaru geograficznego. Co ciekawe, w przeciwieństwie do każdego innego serwisu społecznościowego, który brany był pod uwagę jako alternatywne źródło danych, serwis *Flickr* pozwala określić przeszukiwany obszar nie tylko jako małe obszary o powierzchni zaledwie kilku kilometrach kwadratowych, ale także jako ogromne wycinki Ziemii ograniczone poprzez zdefiniowanie tak zwanego „bounding-boksa", czyli prostokąta otaczającego cały rozpatrywany obszar. Prostokąt ten może definiować dowolnie duży obszar.

Wspomniana funkcja to *flickr.photos.search*. Zwraca ona listę zdjęć z serwisu, które spełniają zadane kryteria. Posiada wiele opcjonalnych argumentów, ale w ramach pracy najbardziej interesujące były:

- *api_key* — jedyny wymagany argument; klucz dostępu do *API*;
- *tags* — słowa-klucze, z którymi powiązane są zdjęcia; oddzielone przecinkami;
- *bbox* — geograficzny „bounding-boks", czyli prostokąt, wewnątrz którego mają znajdować się wyszukiwane zdjęcia; podawany jako cztery wartości liczbowe

oddzielone przecinkiem — <minimalna szerokość geograficzna>, <minimalna wysokość geograficzna>, <maksymalna szerokość geograficzna>, <maksymalna wysokość geograficzna>;

- *has_geo* — wartość logiczna, jeśli prawda („1”), to zwrócone zostaną tylko zdjęcia zawierające metadane o geolokalizacji;
- *extras* — lista dodatkowych informacji, jakie serwis ma dodać do każdego zwracanego zdjęcia; pozwala zaoszczędzić zasoby i czas, który zostałby zmarnowany podczas zapytań o szczegóły każdego kolejnego zdjęcia;

Istnieje ograniczenie na liczbę zdjęć, które funkcja ta może zwrócić dla jednego zestawu ustawień filtrów — maksymalna liczba zdjęć to według specyfikacji 4000. Zdjęcia są zwracane w kolejności od najnowszych do najstarszych. Jeśli więc dla zadanych filtrów znalezionych zostanie ponad 4000 zdjęć, użytkownik dostanie informacje tylko o tych 4000 najświeższych.

4.6.4. Zbieranie danych z sieci społecznościowej

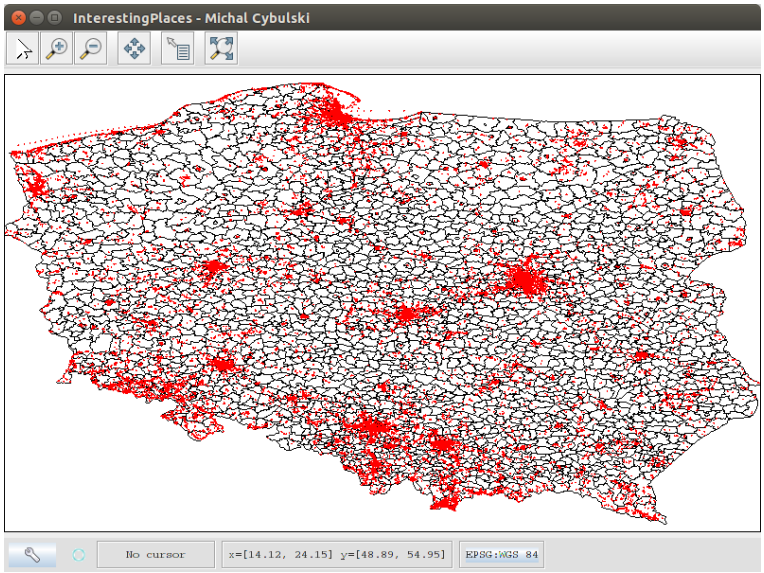
Założeniem pracy było umożliwienie automatycznego zebrania jak największej ilości danych geolokalizacyjnych z sieci społecznościowej, które dodatkowo byłyby opisane za pomocą słów-kluczy definiowanych przez użytkowników sieci. Moim pomysłem podejścia do tego problemu było skorzystanie z dwóch własności funkcji *flickr.photos.search* — umożliwienia wyszukiwania danych w zależności od zadanych haseł, oraz udostępnienia opcji ograniczenia przeszukiwanego obszaru geograficznego, wraz z wymuszeniem dodania dużej ilości dodatkowych danych, które można dodatkowo zapamiętać na wypadek, gdyby kiedyś się przydały do analizy. Zapamiętywane były dane o zdjęciach, połączonych z nimi tagach, oraz o autorach.

Pierwsza próba zbierania danych polegała na przygotowaniu pliku tekstowego z całym słownikiem języka polskiego, wczytanie go do prototypu tworzonej aplikacji i tworzeniu zapytań o każde kolejne słowo słownika jako zadany tag, wraz z ograniczeniem obszaru wyszukiwania do prostokąta zawierającego całe terytorium Polski, którego południowo-zachodni wierzchołek miał współrzędne $(48,879167N; 14,344482E)$, a północno-wschodni wierzchołek $(54,927142N; 24,122314E)$. W wyniku mozolnego zbierania danych przez kilka kolejnych dni udało się zebrać informacje o: *253961* zdjęciach połączonych z *107808* tagami, zrobionych przez *9482* autorów.

Podczas drugiej próby słownik języka polskiego zmieniony został na słownik języka angielskiego, bez modyfikacji pozostałych filtrów zapytania. Okazało się, że

udało się zebrać znacznie więcej danych (liczba zdjęć przekroczyła milion), lecz w wyniku zmiany w międzyczasie implementacji wewnętrznego działania funkcji po stronie serwisu *Flickr*, wyszukiwane były zdjęcia znajdujące się wewnątrz elipsy opisanej na zdefiniowanym prostokącie. Okazało się, że znakomita większość nowych danych to dane pochodzące z terytoriów sąsiadów Polski: Niemiec, Austrii oraz Czech. Liczba zdjęć w Polsce zmieniła się nieznacznie, przy czym ewidentnie było to spowodowane znacznie większą popularnością dzielenia się zdjęciami w zachodniej Europie.

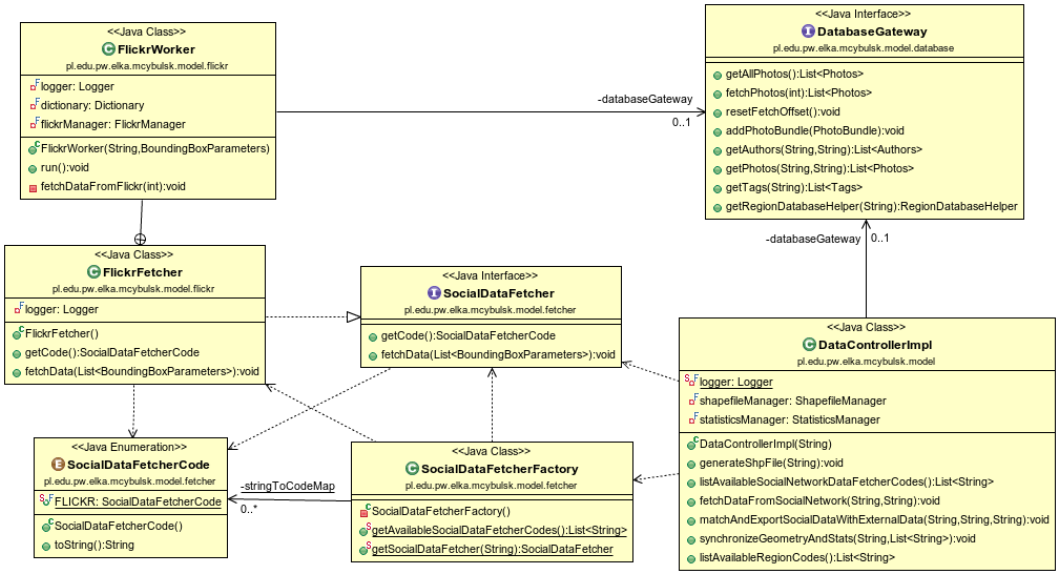
W tej sytuacji pojawiła się obawa, że popularność serwisu u sąsiadów Polski „przysłania" potencjalne wyniki pochodzące z jej terytorium, gdyż zdjęcia pochodzące z Polski mogą być wysyłane rzadziej, a przez to trafiać poza 4000 najświeższych zdjęć dla najpopularniejszych haseł. W tym celu zaimplementowałem rozwiązanie, które miało zminimalizować problem przypadkowego ściągania zdjęć z terytoriów spoza Polski. Moja implementacja polegała na równoczesnym ściąganiu zdjęć dla całego słownika języka angielskiego, dla każdego województwa oddzielnie. Prostokąty zawierające obszary województw znacznie mniej wychodziły poza granice Polski, a ponadto województwa przestały zakłócać wyniki sobie nawzajem dla najpopularniejszych haseł. Zrzut ekranu z aplikacji, na którym prezentowane są zebrane dane na tle mapy, znajduje się na Rys. 4.4.



Rys. 4.4. Okno skończonej aplikacji prezentujące zebrane dane. Każdy czerwony piksel to pojedyncze zdjęcie. Mapa-tło to granice wszystkich gmin w Polsce, która pochodzi z serwisu *OpenStreetMap* (`http://www.openstreetmap.org/`).

4.6.5. Techniczne rozwiązanie

Ściąganie danych z serwisu *Flickr* w sposób opisany w rozdziale 4.6.4 zostało zrealizowane za pomocą klasy implementującej interfejs *SocialDataFetcher*, która nazywa



Rys. 4.5. Diagram *UML* klas związanych z obsługą ściągania danych z sieci społecznościowej *Flickr*.

się *FlickrFetcher*. Aby pobrać dane z sieci społecznościowej kontroler aplikacji, pobiera obiekt „zbieracza danych" za pomocą fabryki *SocialDataFetcherFactory*, której przekazuje jedynie ciąg znaków oznaczający kod sieci społecznościowej, podany przez użytkownika jako parametr programu. Następnie, używając obiektu implementującego interfejs *RegionManager* (nie jest on wyszczególniony na schemacie 4.5, aby nie zaciemniać rysunku, ale mechanizm jego tworzenia i działania jest bardzo podobny do mechanizmu *SocialDataFetcher*) przekazuje „zbieraczowi" informacje o „*bounding-boksach*" regionów, z których obszarów „zbieracz" powinien pobierać zdjęcia z sieci społecznościowej.

Implementacja *FlickrFetcher* tworzy następnie tyle instancji prywatnej klasy wewnętrznej *FlickrWorker*, ile jest obszarów wyszczególnionych przez „*bounding-boksy*" przekazane mu przez głównego kontrolera aplikacji. Tworzone obiekty implementują interfejs *Runnable*, co pozwala uruchomić je w osobnych wątkach. Każdy obiekt *FlickrWorker* dostaje swój własny „*bounding-boks*" regionu, z którego ma ściągać dane, oraz własny obiekt implementujący interfejs *DatabaseGateway* — dzięki temu każdy wątek ma osobne połączenie z bazą danych. Przygotowane wątki są następnie uruchamiane, i główny wątek aplikacji zastyga w oczekiwaniu na zakończenie pracy wątków-potomnych.

Każdy z obiektów typu *FlickrWorker* ściąga dane z innego regionu, dzięki czemu liczba konfliktów związana z pracą na tych samych zdjęciach, autorach i tagach jest minimalizowana. Odrębne zasoby (dostęp do bazy danych oraz do słowników z tagami — każdy wątek posiada swój słownik) sprawiają, że nie istnieje ryzyko wyścigów. Wszelkie konflikty jakie mogą powstać w wyniku jednoczesnego zapisywania do bazy

tych samych zdjęć są bardzo dobrze rozwiązywane przez bazę danych, która przygotowana jest do pracy w środowisku wielodostępu.

Dzięki takiemu rozwiązaniu możliwe jest pobranie dużej ilości danych (rzędu setek tysięcy zdjęć, i milionów tagów z nimi powiązanych) w niedługim czasie, a przy tym istnieje możliwość bardzo prostej rozbudowy aplikacji o kolejne sieci społecznościowe.

4.7. Wyświetlanie danych geolokalizacyjnych

4.7.1. Problemy związane z wyświetlaniem danych geolokalizacyjnych

Jedną z założonych funkcjonalności tworzonych narzędzi było umożliwienie wyświetlenia zebranych danych (wszystkich, lub w kontekście zadanych haseł) na tle map regionu, z którego pochodzą. Zrealizowanie tej funkcjonalności wymagało rozwiązania dwóch problemów:

1. Problem znalezienia odpowiedniego źródła danych geograficznych — innymi słowy elektronicznej mapy, w formacie, który można by użyć w aplikacji.
2. Problem samego wyświetlenia danych w interaktywny, przyjazny użytkownikowi sposób.

4.7.2. Źródło danych o regionach geograficznych

W wyniku analizy dostępnych źródeł danych geograficznych, najbardziej odpowiedni okazał się projekt *OpenStreetMap* (strona domowa projektu: `http://www.openstreetmap.org/`). Jest to projekt tworzony i utrzymywany przez społeczność internetową, którego celem jest stworzenie otwartej i darmowej mapy całego świata. Dostęp do niego nie jest w żaden sposób ograniczony (nawet w przypadku zastosowań komercyjnych), a dodatkowo istnieje możliwość wyeksportowania (w wielu różnych formatach) map całego świata, lub wybranych jego fragmentów. Projekt ten cieszy się ogromną popularnością i skupia wokół siebie bardzo dużą społeczność aktywnie pomagającą go ulepszać. Wszystko to sprawia, że mapy tworzone w jego ramach są bardzo dokładne i idealnie nadają się do zastosowania jako źródło danych o regionach geograficznych.

Mapy wyeksportowane z projektu *OpenStreetMap* można zaimportować przy użyciu narzędzia *Osm2pgsql* do bazy danych *PostgreSQL* z nakładką *PostGIS*. Importowanie do bazy danych obszaru wielkości Polski trwa około kilku godzin. W wyniku importu danych tworzonych jest zbiór tabel zawierających wszystkie dane dotyczące importowanego regionu dostępne w serwisie *OpenStreetMap*. W ramach pracy wykonałem proces importu mapy Polski do bazy danych tworzonego systemu.

Warto także zwrócić uwagę na jeden bardzo ważny szczegół dotyczący mapy Polski udostępnianej przez *OpenStreetMap*. Użytkownicy z Polski zapewnili mapowanie wszystkich gmin (oraz jednostek terytorialnych o powierzchniach większych niż gminy) na odpowiednie kody *TERYT*. Dzięki temu tworzona w ramach pracy aplikacja ma dostęp zarówno do danych o obszarach poszczególnych podregionów Polski, jak i do oficjalnych identyfikatorów tych regionów. Dzięki temu istnieje możliwość tworzenia zapytań do bazy danych dotyczących konkretnych interesujących użytkownika regionów.

4.7.3. Wyświetlanie danych geograficznych

Do wyświetlania danych geograficznych zebranych w bazie danych przez tworzoną aplikację użyłem biblioteki *open-source* o nazwie *GeoTools* (adres projektu: `http://www.geotools.org/`). Jest to zbiór narzędzi pozwalający na prostą obsługę danych przestrzennych w programach tworzonych w języku *Java*. Jedną z funkcji biblioteki jest umożliwienie bardzo łatwej i szybkiej implementacji okienka, w którym wyświetlane są przygotowane przez programistę warstwy map.

Biblioteka *GeoTools* po podaniu parametrów połączenia sama obsługuje komunikację z bazą danych *PostGIS*. Programista ma możliwość wyszczególnienia, z których tabel w bazie program ma pobierać dane do wyświetlenia na ekranie. Istnieje także mechanizm filtrowania danych tak, aby w okienku wyświetlać tylko dane spełniające pewne warunki — na przykład tylko wielokąty oznaczone jako granice województw lub gmin.

Przy użyciu biblioteki *GeoTools* zaimplementowałem klasę *MapViewer*, która łączy się z bazą danych, a następnie przygotowuje dwie warstwy mapy do wyświetlenia użytkownikowi:

- Warstwę z mapą regionów — w przykładowej implementacji są to granice polskich gmin.
- Warstwę ze zdjęciami — źródłem zdjęć jest tabela *PHOTOS_ GEOM*, która może zostać zsynchronizowana z tabelą *PHOTOS* na różne sposoby (na przykład zostawiając tylko zdjęcia związane z podaną listą tagów).

Użytkownik ma możliwość dowolnego przybliżania i oddalania mapy, a także wybierania interesujących go punktów symbolizujących zdjęcia i dowiadywania się szczegółów ich dotyczących (między innymi tytułu zdjęcia oraz adresu *URL* prowadzącego do strony ze zdjęciem w sieci społecznościowej).

4.8. Dodatkowe źródło danych statystycznych — *Bank Danych Głównego Urzędu Statystycznego*

W wyniku założeń dotyczących ograniczenia rozpatrywanego w ramach pracy obszaru do terytorium Polski oczywistym pierwszym wyborem dotyczącym dodatkowego źródła danych statystycznych były systemy udostępniane przez *Główny Urząd Statystyczny* — w szczególności system *Bank Danych Lokalnych* (adres internetowy systemu: `http://stat.gov.pl/bdl/app/strona.html?p_name=indeks`). Umożliwia on przeglądanie danych statystycznych dotyczących najróżniejszych dziedzin życia, na przykład gęstości zaludnienia lub liczby osób zamieszkujących dany obszar, ale także danych takich jak liczba hoteli i turystów w danym regionie.

W *Banku Danych Lokalnych* istnieje możliwość wybrania interesujących użytkownika danych, pogrupowania ich w wybranych jednostkach administracyjnych Polski (na przykład gminach — choć różne dane udostępniane są na różnych poziomach szczegółowości), a następnie eksport znalezionych danych do pliku. Obsługiwane są między innymi format *XLS* (wspierany przez narzędzie *Microsoft Excel*) oraz format *CSV* — który wspierany jest przez narzędzia tworzone w ramach tej pracy.

Pliki wyeksportowane w ten sposób nie nadają się od razu do przetworzenia przez tworzoną w ramach pracy aplikację, ponieważ zawierają niepotrzebne niestandardowe nagłówki. Po ich usunięciu, gdy plik będzie już miał standardową strukturę pliku *CSV* dane można z powodzeniem zaimportować do aplikacji.

4.9. Łączenie danych z różnych źródeł

4.9.1. Wejściowy format plików

Wejściowy format pliku dla aplikacji miał być z założenia prosty i bardzo uniwersalny. Zdecydowałem się użyć w tym celu formatu *CSV* (ang. „*Comma Separated Values*"). Jest to prosty, mało sformalizowany format tekstowy, który pozwala na wygodne przechowywanie danych o charakterze tabelarycznym. Ze względu na prostotę nie pozwala na implementację bardzo wyszukanych mechanizmów, ale nie były one potrzebne w ramach tworzonych narzędzi. Przykładowy plik poprawnie interpretowany przez aplikację znajduje się na Listingu 3.

Ogólny format pliku obsługiwany przez aplikację najłatwiej zdefiniować jako pojedynczy wiersz w pliku. Obsługiwany format pokazany jest na Listingu 4.

4.9.2. Grupowanie danych z sieci społecznościowej pod względem regionów geograficznych

Grupowanie danych z sieci społecznościowej pod względem pochodzenia z poszczególnych regionów geograficznych zapewnione jest przez współpracujące ze

```
201011,Boleslawiec * (1),1680
201022,Boleslawiec * (2),49
201032,Gromadka * (2),21
201043,Nowogrodziec * (3),86
201052,Osiecznica * (2),17
201062,Warta Boleslawiecka * (2),76
202011,Bielawa * (1),861
202021,Dzierzoniow * (1),1715
202031,Pieszyce * (1),151
202041,Pilawa Gorna * (1),320
```

Listing 3. Początek pliku wejściowego w formacie *CSV* zawierającego gęstość zaludnienia na $1 km^2$

```
<identyfikator regionu>,<nazwa regionu>,<wartosc statystyczna>
```

Listing 4. Format pliku wejściowego.

sobą klasy implementujące interfejsy *StatisticsManager* oraz *RegionManager* (i jego klasy pomocnicze). Odpowiednia implementacja interfejsu *RegionManager* zwracana jest przez fabrykę *RegionManagerFactory* po przekazaniu ciągu znaków stanowiącego kod regionu (który to z kolei jest przekazywany przez głównego kontrolera aplikacji jako jeden z argumentów przekazanych przez użytkownika). W ramach tej pracy przygotowałem implementację *PolandRegionManager*, która specjalizuje się w dostarczaniu danych o polskich regionach.

Plik wejściowy podany przez użytkownika parsowany jest przez obiekt implementujący interfejs *StatisticsManager* przy pomocy biblioteki *Apache Commons CSV*. W wyniku tego procesu plik *CSV* zamieniany jest na obiektową reprezentację poprzez listę instancji klasy *Territory*. Klasa ta jest zwykłym, prostym obiektem typu *POJO* przechowującym informacje o identyfikatorze terytorium, jego nazwie, oraz o danych statystycznych z niego pochodzących.

Lista obiektów *Territory* jest następnie przetwarzana i wypełniana danymi poprzez użycie interfejsu *RegionManager*, który za pomocą metody *getNumberOfPhotosInTerritory(String territoryId)* zwraca liczbę zdjęć pochodzących z regionu o zadanym identyfikatorze. W przypadku Polski identyfikatorem tym jest kod *TERYT*. Wewnętrznie, obiekt implementujący interfejs *RegionManager* korzysta z obiektu pomocniczego implementującego interfejs *RegionDatabaseHelper*, który potrafi wykonywać bezpośrednie zapytania na bazie danych o liczbę zdjęć pochodzącą z danego regionu. Obiekt klasy *PolandRegionDatabaseHelper*, który jest domyślną implementacją tego interfejsu korzysta ze zmapowanej za pomocą techniki *O-R Mappingu* tabeli *GMINY_STATS*, która musi być zawczasu zsynchronizowana odpowiednią funkcją programu.

Gdy cała lista obiektów *Territory* jest już w pełni wypełniona danymi, obiekt implementujący interfejs *StatisticsManager* zamienia ją z powrotem na format *CSV*, tym razem z jedną dodatkową kolumną przechowującą informacje o liczbie zdjęć z serwisu społecznościowego na danym obszarze.

4.9.3. Wyjściowy format plików

Wyjściowy format pliku widoczny na Listingu 5 jest bardzo zbliżony do formatu wejściowego. Różnica polega na dodaniu przez program dodatkowej kolumny zawierającej liczbę zdjęć pochodzącą z danego regionu. Użytkownik końcowy może zaimportować plik w takim formacie do zewnętrznych aplikacji wspierających analizę danych takich jak środowiska *Octave* lub *Matlab* i przetwarzać je tam dalej.

```
0201011,Boleslawiec * (1),1680,336
0201022,Boleslawiec * (2),49,5
0201032,Gromadka * (2),21,12
0201043,Nowogrodziec * (3),86,145
0201052,Osiecznica * (2),17,1
0201062,Warta Boleslawiecka * (2),76,38
0202011,Bielawa * (1),861,31
0202021,Dzierzoniow * (1),1715,65
0202031,Pieszyce * (1),151,14
0202041,Pilawa Gorna * (1),320,24
```

Listing 5. Początek pliku wyjściowego w formacie *CSV* zawierającego gęstość zaludnienia na $1 km^2$ oraz liczbę zdjęć pochodzących z danej gminy.

5. Wyniki i testy

5.1. Wynik pracy

W wyniku pracy powstał zestaw narzędzi udostępniający całą założoną funkcjonalność opisaną w rozdziale 4.1.1. Domyślna implementacja narzędzi umożliwia użytkownikowi przeprowadzanie analiz związanych z badaniem liczby zdjęć:

- Pochodzących z serwisu *Flickr* i posiadających metadane geolokalizacyjne wskazujące na miejsce zrobienia zdjęcia.
- Zrobionych na terytorium Polski, ze szczegółowością analiz ustaloną na gminy. Gmina, jako podstawowa, najmniejsza jednostka administracyjna kraju, mogła być prawidłowo identyfikowana oficjalnym identyfikatorem regionu *TERYT*.

Szczegółowa instrukcja opisująca przygotowywanie środowiska uruchomieniowego dla narzędzi, oraz ich użycie, znajduje się w załączniku do pracy (Zał. A). Dodatkowo, instrukcje znajdują się także w tak zwanych plikach *README* dołączonych do zestawu z kodem źródłowym narzędzi.

Skrócony schemat użycia narzędzi przygotowanych w ramach pracy jest następujący:

1. Należy przy użyciu przygotowanych skryptów zainstalować potrzebne narzędzia — w szczególności wymagana jest baza danych *PostgreSQL* (narzędzia testowane były z wersją 9.3) oraz nakładka *PostGIS* (sprawdzona wersja to 2.1).
2. Należy przy użyciu skryptów stworzyć wymaganą bazę danych oraz jej strukturę. Interakcja ze strony użytkownika jest minimalna i sprowadza się do uruchomienia skryptów. Należy także zaimportować dane dotyczące Polski z *OpenStreetMap* — w praktyce także ogranicza się to, do wykonania skryptu.
3. Aplikacja napisana w języku *Java*, realizująca większość funkcjonalności, jest w tym momencie gotowa do użycia. Należy przygotować plik z listą tagów, z którymi ściągane zdjęcia mają być powiązane. W zestawie z narzędziami dołączony jest prosty słownik języka angielskiego, który może posłużyć jako źródło tagów.
4. Należy w pliku konfiguracyjnym programu wypełnić dane dotyczące połączenia z bazą danych — takie jak adres maszyny z bazą danych (na przykład „*localhost*”), port na którym baza nasłuchuje, oraz dane uwierzytelniające użytkownika. Nic nie stoi na przeszkodzie, żeby stworzyć programowi osobne konto użytkownika w bazie danych — ważne jest tylko, aby konto to posiadało uprawnienia dostępu do tabel wymaganych przez program.

5. Należy uruchomić aplikację w trybie ściągania danych i pozwolić jej pracować. Wypytanie serwisu społecznościowego o wszystkie tagi może potrwać nawet kilka dni — w zależności od regionu, popularności tagów i długości listy.

6. Po ściągnięciu danych należy wykonać synchronizację zebranych informacji o zdjęciach z przestrzennymi tabelami w bazie danych. Proces ten przetwarza proste informacje o geolokalizacji zdjęć na przestrzenne obiekty i zapisuje je w odpowiedniej tabeli, a następnie dodatkowo je agreguje pod względem pochodzenia z poszczególnych gmin. Proces ten wykonuje się za pomocą aplikacji w trybie synchronizacji. Istnieje możliwość sprecyzowania listy tagów, z którymi muszą być połączone zdjęcia, aby mogły być zsynchronizowane. Umożliwia to analizę danych o zdjęciach pod względem zadanych przez użytkownika haseł.

7. Po wykonaniu synchronizacji użytkownik ma możliwość dopasowywania zsynchronizowanych danych do danych z zewnętrznych źródeł podanych w odpowiednim formacie.

Każdy z kroków (od kroku trzeciego) można powtarzać wielokrotnie, zwiększając z czasem zasoby danych możliwe do analizy. Warto pamiętać, że serwisy społecznościowe są bardzo dynamicznymi środowiskami i wraz z upływem czasu możliwe do zebrania dane się zmieniają. Nowe dane mogą pojawiać się w sieci, a stare mogą z niej znikać — warto więc powtarzać proces pobierania danych raz na jakiś czas, agregując dane przez dłuższy okres czasu.

5.2. Testy

5.2.1. Zebrane dane — statystyki

Stworzone narzędzia należało odpowiednio przetestować i udowodnić ich działanie oraz wartość, dlatego przeszedłem cały proces wyszczególniony w rozdziale 5.1. Warto zaznaczyć, że informacje zawarte w tym rozdziale (5.2) opisują przykładowe dane, które udało mi się zebrać podczas tworzenia pracy. Ze względu na zmienność internetowych trendów, dane zbierane w innym czasie mogą diametralnie się różnić.

W ramach testów przeprowadziłem zbieranie danych: jako źródło danych najpierw wykorzystując załączony do zestawu z narzędziami słownik języka angielskiego, a następnie listę stu bardzo popularnych haseł znalezioną w internecie. Program zbierał dane przez około dwie doby, logując cały przebieg zbierania danych zarówno do konsoli systemu operacyjnego, jak i do zewnętrznego pliku, aby umożliwić diagnozę ewentualnych problemów w przypadku awarii. Po zakończeniu ściągania, zebrałem proste statystyki dotyczące zebranych danych. Tabela 5.1 zawiera informację o liczbie wierszy w poszczególnych tabelach w bazie danych.

Tab. 5.1. Liczba wierszy w poszczególnych tabelach w bazie danych po dwóch dobach ściągania danych z sieci społecznościowej.

Tabele w bazie danych			
PHOTOS	*AUTHORS*	*TAGS*	*PHOTOS_TAGS*
400418	9494	144116	4716329

Udało się zebrać ponad czterysta tysięcy zdjęć pochodzących z obszaru Polski, których autorem było prawie dziewięć i pół tysiąca osób. Ze zdjęciami powiązanych było ponad sto czterdzieści tysięcy haseł, a samych połączeń pomiędzy tagami a zdjęciami — prawie pięć milionów. Statystycznie jeden użytkownik sieci społecznościowej umieścił w niej około czterdzieści dwie fotografie z danymi geolokalizacyjnymi. Po zsynchronizowaniu danych z tabelami przestrzennymi, możliwe było także sprawdzenie, z których gmin pochodzi najwięcej zdjęć. Statystykę tę obrazuje Tabela 5.2a. Postanowiłem także sprawdzić, jakie tagi są najpopularniejsze wśród zebranych danych — informacje te znajdują się w Tabeli 5.2b.

Tab. 5.2. Statystyki obrazujące zebrane dane.

(a) Liczba zdjęć w poszczególnych gminach.

NAME	*TERYT:TERC*	*PHOTOS*
Warszawa	1465011	78047
Kraków	1261011	62471
Wrocław	0264011	25358
Gdańsk	2261011	17520
Łódź	1061011	16033
Sopot	2264011	12263
Poznań	3064011	10294
Katowice	2469011	7369
Gdynia	2262011	6040
Opole	1661011	5610

(b) Najpopularniejsze tagi.

NAME	*TAG_COUNT*
poland	209663
polska	120633
warsaw	48819
canon	36995
warszawa	32565
krakow	28393
europe	27672
canonefs18135mmf3556is	26864
canoneos550d	26855
polen	25488

Na uwagę zasługuje fakt, że duże i popularne miasta zdominowały zupełnie listę najpopularniejszych gmin pod względem pochodzenia fotografii. Większość najpopularniejszych tagów to nazwy opisujące miejsca, bądź sprzęt fotograficzny. W załączniku do pracy znajduje się dłuższa lista najpopularniejszych tagów (Zał. C).

5.2.2. Przykładowa analiza zebranych danych

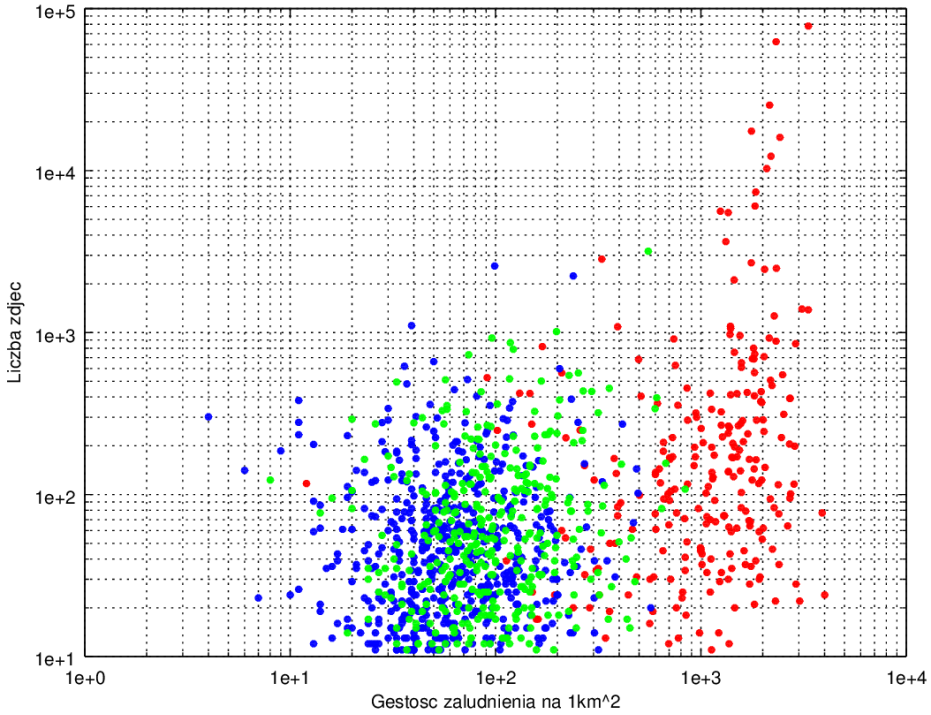
Zebrane i pogrupowane pod względem pochodzenia zdjęcia, można zgodnie z założeniami dopasować do danych statystycznych z zewnętrznego źródła i wyeksportować, a następnie badać innymi narzędziami. W ramach pracy przeprowadzona w ten sposób została przykładowa prosta analiza danych. Jako zewnętrzne źródło danych statystycznych użyty został *Bank Danych Lokalnych Głównego Urzędu Statystycznego*, a wszelkie analizy przeprowadziłem za pomocą darmowej aplikacji *Octave*.

Najbardziej intuicyjną formą analizy jest dla człowieka bez wątpienia analiza graficzna — z tego względu postanowiłem w pierwszej kolejności przedstawić zebrane dane w graficznej formie. Niestety od razu okazało się, że problemem jest stosunkowo mała ilość dostępnych do analizy danych pobranych z sieci społecznościowej — z wstępnej analizy wynikało, że w bazie danych nie ma ani jednego zdjęcia z obszaru aż 540 gmin. Postanowiłem, że do dalszych analiz nie będę brał pod uwagę gmin, z których obszarów nie udało się zebrać co najmniej dziesięciu zdjęć, gdyż nie wnosiłyby one nic do analizy i powodowałyby tylko zaciemnienie danych. Gmin takich było 1334. Nie wykluczałem przy tym dalszego zawężania rozpatrywanych regionów.

W pierwszej kolejności postanowiłem dopasować zebrane z sieci społecznościowej dane, do danych o gęstości zaludnienia w poszczególnych gminach. Patrząc na wykres, od razu okazało się, że ewentualna zależność (jeśli miała istnieć) między dostępnymi próbkami nie była liniowa. Postanowiłem przygotować wykres korzystając z logarytmicznej skali dla obu osi — w wyniku tego udało się sporządzić wykres, Rys. 5.1.

Rozłożenie danych okazało się być bardzo ciekawe, dlatego postanowiłem dodatkowo rozróżnić poszczególne gminy pod kątem ich typu — powstał więc podział na: *gminy miejskie*, *gminy wiejskie* oraz *gminy miejsko-wiejskie*. Analiza wykresu pozwala sądzić, że nie występuje korelacja pomiędzy liczbą zdjęć pochodzącą z danej gminy, a jej gęstością zaludnienia. Okazało się, że liczba gmin, z których pochodzi tysiąc albo więcej zdjęć jest bardzo ograniczona — wynosi dokładnie dwadzieścia osiem gmin, a większość z nich to gminy miejskie. Łatwo zauważyć również, że z rejonów gmin miejskich oraz miejsko-wiejskich nie pochodzi zazwyczaj więcej zdjęć niż z gmin wiejskich. Między gminami miejskimi, nawet tymi o bardzo dużej gęstości zamieszkania, widoczne są, trudne do wytłumaczenia, bardzo duże dysproporcje w liczbie zdjęć pochodzących z danych regionów.

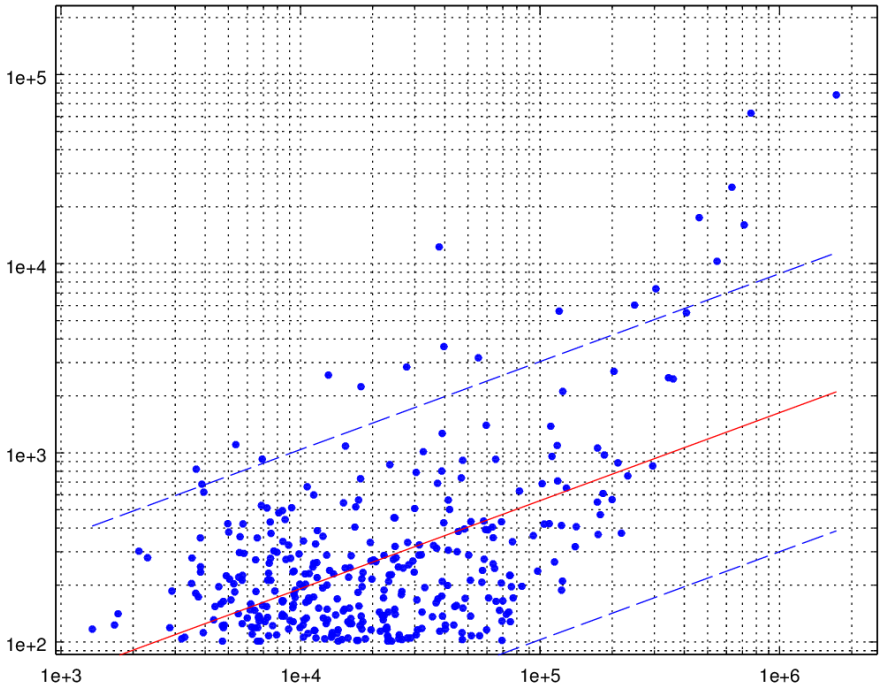
W ramach przykładowej analizy, postanowiłem postawić się w sytuacji turysty, który szuka ciekawego miejsca, w którym mógłby spędzić urlop. Postanowiłem dopasować zebrane z sieci społecznościowej dane do danych o liczbie ludności zamieszkującej poszczególne gminy, znowu przedstawiając dane na wykresie w skali



Rys. 5.1. Wykres przedstawiający zależność liczby zdjęć od gęstości zaludnienia.
Legenda — kolor:
czerwony — gminy miejskie;
zielony — gminy miejsko-wiejskie;
niebieski – gminy wiejskie;

logarytmicznej (skala ta okazała się najpraktyczniejsza w każdym przypadku). Skłoniło mnie do tego rozumowanie, że w gminach, z których pochodziło znacznie więcej zdjęć, niż wskazywałaby na to liczba ludzi w nich mieszkających, różnica w liczbie zdjęć mogła potencjalnie być spowodowana atrakcyjnością miejsca i ruchem turystycznym. Turyści, którzy uznaliby miejsce za ciekawe i warte uwagi mogli je uwieczniać na zdjęciach, którymi potem dzielili się w internecie. Powstały w wyniku dopasowania wybranych przeze mnie danych wykres, znacznie bardziej niż poprzedni wskazywał na możliwość istnienia jakiegoś rodzaju korelacji pomiędzy dwoma zestawami danych. Aby uwiarygodnić wyniki i pozbyć się „szumu", postanowiłem dodatkowo ograniczyć liczbę gmin branych pod uwagę w mojej analizie tylko do takich, z których pochodziło więcej o co najmniej sto zdjęć — pod uwagę brane już było tylko 361 gmin. W celu wytypowania potencjalnie ciekawych gmin użyłem regresji liniowej, sprowadzając uprzednio problem z postaci logarytmicznej do postaci liniowej.

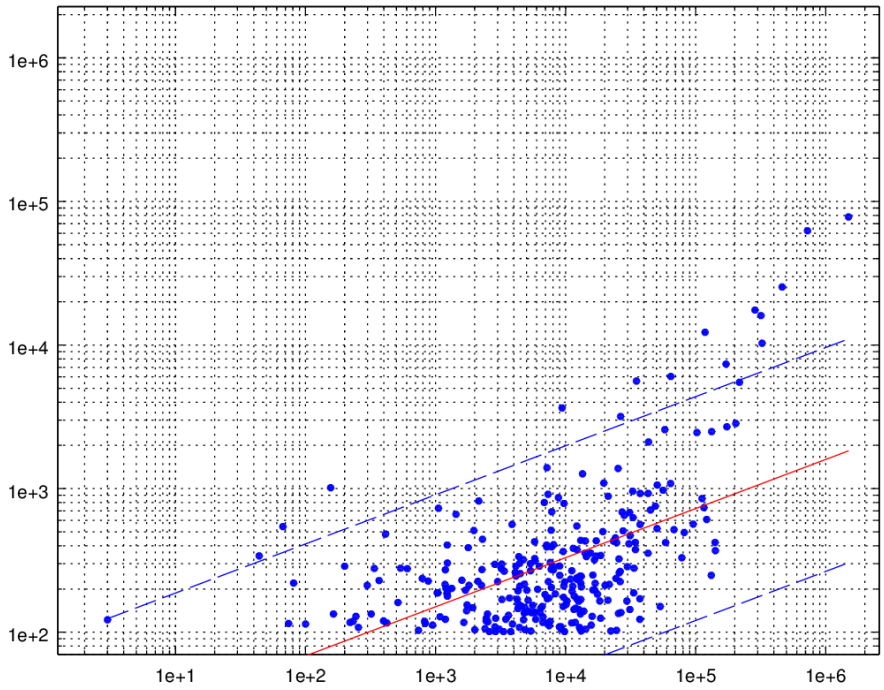
Współczynniki modelu statystycznego obliczyłem wykorzystując metodę najmniejszych kwadratów. Bardzo pomocne okazały się być funkcje wbudowane w środowisko *Octave*, co ogarniczyło wysiłek do skorzystania z gotowych rozwiązań. W wyniku analizy powstał wykres przedstawiony na Rys. 5.2.



Rys. 5.2. Wykres przedstawiający zależność liczby zdjęć od liczby mieszkańców gminy.

Prosta regresji zaznaczona jest kolorem czerwonym, a przerywaną linią oznaczyłem odległość równą $1,96$ estymowanego standardowego odchylenia (wartość ta standardowo dla rozkładu normalnego gwarantuje, że 95% próbek znajdzie się wewnątrz przedziału, a w rozpatrywanym przeze mnie przypadku okazało się to być dokładnie $94,708\%$). Niestety okazało się, że w tym przypadku między danymi z obu źródeł również nie występuje oczywista korelacja. Postanowiłem oznaczyć wszystkie gminy, które nie zmieściły się wewnątrz wybranego przeze mnie obszaru na wykresie, jako potencjalne ciekawe miejsca. Gmin takich było dziewiętnaście. Oznaczone gminy zamierzałem skonfrontować z wynikami dalszej analizy.

Kolejnym krokiem była analiza zależności danych pobranych z serwisu *Flickr* do wybranego statystycznego wskaźnika turystycznego. Postanowiłem w tym celu wykorzystać statystykę opisującą liczbę rezydentów (Polaków) korzystających z hoteli w poszczególnych gminach. Nie modyfikowałem metody analizy, ani jej parametrów. Wynikowy wykres znajduje się na Rys. 5.3.



Rys. 5.3. Wykres przedstawiający zależność liczby zdjęć od liczby rezydentów (Polaków) korzystających z hoteli w poszczególnych gminach.

Tym razem piętnaście gmin znalazło się poza obszarem ograniczanym przez dwie obrane przeze mnie proste określające maksymalną odległość od odchylenia standardowego (wewnątrz tego obszaru znajdowało się $94,718\%$ rozpatrywanych gmin). Zapisałem listę nowo wytypowanych ciekawych miejsc i porównałem je z miejscami wytypowanymi poprzednio, starając się dopasować gminy wybijające się popularnością w obu analizach. Dopasowane listy znajdują się w Tabelach 5.3 oraz 5.4.

Na podstawie tej małej próby można zanotować pewne obserwacje dotyczące zebranych danych. Po pierwsze, gminy wytypowane jako potencjalnie ciekawe zarówno w analizie biorącej pod uwagę liczbę ludności zamieszkującą rozpatrywane regiony, jak i tej skupiającej się na polskich turystach, wszystkie zdecydowanie należą do bardzo popularnych i wyjątkowych miejsc w Polsce. Z perspektywy turysty, wszystkie te gminy są miejscami wartymi odwiedzenia — pamiętając oczywiście o tym, że każde ma swoją specyfikę i trzeba brać ją pod uwagę. Można więc zauważyć, że popularność regionu w serwisie *Flickr* zazwyczaj dosyć dobrze odzwierciedla ogólną popularność danego miejsca — przynajmniej w kontekście najpopularniejszych gmin w Polsce.

Drugim zaobserwowanym, istotnym faktem jest, że danych pochodzących z obszaru Polski w serwisie *Flickr* jest niestety mało. Dla przypomnienia, gmin, z których

Tab. 5.3. Gminy wytypowane jako miejsca potencjalnie ciekawe i wyróżniające się w obu etapach analizy.

Dopasowane kody *TERYT*				
Nazwa gminy	*TERYT*	*Mieszkańcy*	*Turyści*	*Liczba zdjęć*
M. Wrocław	0264011	632067	462706	25358
M. Łódź	1061011	711332	317862	16033
Oświęcim	1213011	39664	9400	3644
Wieliczka	1219053	55231	26471	3180
M. Kraków	1261011	758992	724702	62471
M. Warszawa	1465011	1724404	1498581	78047
M. Opole	1661011	120146	35066	5610
M. Gdańsk	2261011	461531	286023	17520
M. Gdynia	2262011	248042	64459	6040
Sopot	2264011	37903	118164	12263
M. Katowice	2469011	304362	171288	7369
M. Poznań	3064011	548028	324287	10294

nie pochodziło ani jedno zdjęcie było aż 540. Gmin, na których obszarze znalazło się pomiędzy jednym a dziewięcioma zdjęciami było 570. Razem stanowi to prawie 45% wszystkich gmin w Polsce. Gmin, z których pochodziło sto lub więcej zdjęć było zaledwie 361. W kontekście prowadzenia badań, które miałyby dawać jednoznaczne rezultaty, stanowi to niestety mało danych.

Kolejna obserwacja związana jest z gminami, które zostały wytypowane jako potencjalnie ciekawe w tylko jednej z dwóch analiz[23]. Są to miejsca zdecydowanie mniej popularne i ważne w kontekście całego kraju, ale mimo to, były na tyle popularne w serwisie *Flickr*, że w pewnym kontekście wyróżniły się na tle ogółu. W zbiorze tym mogą pojawiać się zarówno naprawdę ciekawe miejsca, jak i niestety błędne dopasowania — wynikające albo z małej ilości danych z sieci społecznościowej, albo z błędnych danych statystycznych z zewnętrznego źródła. Uwaga ta dotyczy zwłaszcza danych z sieci społecznościowej, ale w danych statystycznych dostarczanych przez *Główny Urząd Statystyczny* także zdarzały się próbki, które moim zdaniem mogą być uznane za podejrzane. Dla przykładu, w innym zestawieniu danych pochodzącym z *Banku Danych Lokalnych* znajdowała się liczba turystów zagranicznych, którym udzielono noclegu w małej wiejskiej gminie o nazwie *Łagów* w roku 2013 — liczyła ona według statystyk *GUS* aż 38195 osób (czyli dziennie średnio około

[23]Gmina Oświęcim pojawiła się w obu zestawieniach, ponieważ kod *TERYT* „*1213011*" oznacza gminę miejską, a kod „*1213062*" gminę wiejską. Są to więc dwa różne regiony, mimo, że mają tę samą nazwę.

Tab. 5.4. Gminy wytypowane jako miejsca potencjalnie ciekawe i wyróżniające się w jednym z dwóch etapów analizy.

Niedopasowane kody *TERYT* — pochodzące z *analizy ludności*				
Nazwa gminy	*TERYT*	*Mieszkańcy*	*Turyści*	*Liczba zdjęć*
Kazimierz Dolny	0614043	6919	43113	926
Skierbieszów	0620102	5367	brak danych	1105
Oświęcim	1213062	17874	brak danych	2237
Zakopane	1217011	27721	203664	2843
Bukowina Tatrzańska	1217032	13070	57940	2576
Hel	2211011	3668	2148	820
Jastarnia	2211021	3874	31064	683
Niedopasowane kody *TERYT* — pochodzące z *analizy turystyki*				
Nazwa gminy	*TERYT*	*Mieszkańcy*	*Turyści*	*Liczba zdjęć*
Dobczyce	1209013	15114	67	544
Żukowo	2205083	32542	156	1014
Kętrzyn	2808032	8439	44	340

105 turystów z zagranicy znajdowało tu nocleg). Polacy nie byli uwzględnieni w tych statystykach. Sytuacja taka była oczywiście możliwa, ale wydawała mi się mało prawdopodobna.

Przy tego typu niepewnych źródłach danych trudno jest odróżnić, czy wytypowane miejsce naprawdę jest ciekawe, czy jest to po prostu błąd związany z małą ilością danych (bądź ich niską jakością). W tym przypadku bardzo ważne jest doświadczenie i intuicja badacza. Przykładowo, wytypowana w przeprowadzonym przeze mnie badaniu wiejsko-miejska gmina *Dobczyce* znalazła się w zestawieniu ze względu na to, że pomimo goszczenia w roku 2013 zaledwie 67 polskich turystów, była miejscem z którego pochodziły aż 544 zdjęcia. Z perspektywy turysty szukającego ciekawego miejsca zdecydowanie zainteresowałbym się skąd wzięła się taka dysproporcja — być może naprawdę udałoby się znaleźć miejsce w jakiś sposób wyjątkowo ciekawe.

Z drugiej strony, wytypowana poprzez „turystyczną analizę" wiejska gmina *Kętrzyn*, która według statystyk z *Głównego Urzędu Statystycznego* gościła zaledwie 44 polskich turystów, a z której pochodziło aż 340 fotografii również na pierwszy rzut oka wyglądała interesująco, ale okazało się, że jest to gmina wiejska tuż obok gminy miejskiej o tej samej nazwie, która to z kolei gościła aż 11312 turystów, a zdjęć w niej zostało zrobionych tylko 95 — nie była więc nawet brana pod uwagę w analizie. Zsumowane wartości liczby turystów oraz liczby pochodzących z regionów zdjęć sprawiają, że gmina ta przestaje być w jakikolwiek sposób ciekawa z punktu widzenia tego typu

analizy. Mając do dyspozycji tego typu dane nie da się niestety z dużą dozą prawdopodobieństwa stwierdzić automatycznie, czy dany region jest miejscem naprawdę ciekawym, czy został wytypowany w wyniku błędu związanego z niedostateczną jakością danych, kluczowe są więc doświadczenie i intuicja badacza.

Trzeba uwzględnić także inny aspekt sytuacji — zdarza się, że mała liczba zdjęć pochodząca z danego regionu nie jest w rozpatrywanym kontekście czymś niepożądanym. Sprowadzając to do sytuacji turysty: istnieją różne preferowane formy wypoczynku. Miejsca, o których wiadomo, że posiadają rozbudowaną bazę hotelową, ale nie są popularne w sieciach społecznościowych, mogą oznaczać po prostu regiony bardzo spokojne, w którym można zwyczajnie zrelaksować się, zamiast zwiedzania okolicznych atrakcji. W przypadku takich założeń warto byłoby prawdopodobnie nie odrzucać od razu gmin, w których zdjęć jest mniej niż sto — trzeba by poważnie zastanowić się jaka minimalna liczba zdjęć pozwoli odfiltrować próbki zupełnie nieprzydatne, ale nie odrzucając przy tym miejsc, które umożliwiają spokojny wypoczynek. W analizowanym przypadku, gdy odrzucałem z góry tylko gminy, z których obszarów pochodziło mniej niż dziesięć zdjęć, analizując znalazłem gminę *Świeradów-Zdrój*, która jest bardzo popularną miejscowością typowo uzdrowiskową, a z której pochodziła niewielka liczba zdjęć. Metody i założenia analizy zależą więc od wymagań badacza.

Skorzystanie ze statystyk dotyczących turystycznych zachowań wyłącznie Polaków również wynikało ze sposobu udostępniania danych przez system *Głównego Urzędu Statystycznego*. Nie udało mi się znaleźć statystyk, które uwzględniałyby wszystkich turystów zagregowanych razem, niezależnie od ich pochodzenia i obywatelstwa. Mogło to wprowadzić przekłamania, zwłaszcza biorąc pod uwagę fakt, że większość haseł opisujących zebrane z serwisu społecznościowego dane była w języku angielskim. Myślę jednak, że tak powszechne korzystanie z języka angielskiego do opisu danych wynika głównie z międzynarodowej specyfiki serwisu *Flickr* — Polacy także opisują wysyłane przez siebie zdjęcia hasłami w języku angielskim, aby były zrozumiałe dla szerszego grona odbiorców.

Warto nadmienić, że w przypadku korzystania z *Banku Danych Lokalnych Głównego Urzędu Statystycznego* wiele statystyk nie jest dostępnych na aż takim poziomie szczegółowości jak gminy (czyli poziom *NTS 5*, w nomenklaturze *NTS*, opisanej w rozdziale 4.1.3). Ustalenie szczegółowości badanych obszarów na tym poziomie uniemożliwiło przeprowadzenie badań związanych ze statystykami dostępnymi tylko dla większych regionów. Z drugiej strony jednak — ustalenie innego poziomu szczegółowości zmieniłoby poważnie kontekst przeprowadzanych badań — w końcu wyniki dotyczyłyby na przykład całych województw, bądź regionów kraju, które to mają znacznie większą powierzchnię niż gminy. W zależności od wymagań badacza,

zmiana taka mogłaby być pożądana. Sposób implementacji stworzonych narzędzi pozwala małym nakładem pracy ją zmienić, aby program umożliwiał grupowanie zebranych danych do regionów innego rodzaju niż gminy.

Należy pamiętać, że przeprowadzona przeze mnie analiza, jak również i wszystkie wnioski, dotyczą danych, które udało mi się zebrać z terytorium Polski, wyłącznie z serwisu *Flickr*. Stworzone w ramach tej pracy narzędzia mogą być łatwo rozbudowane zarówno o obsługę dodatkowych regionów, jak i innych sieci społecznościowych, a sytuacja związana z dostępem do danych może diametralnie się zmienić. W szczególności, bardzo wiele zależy od popularności danego serwisu społecznościowego na rozpatrywanym terenie. Ze względu na problemy opisywane w rozdziale 4.6.4 związane z przypadkowym pobraniem dużej liczby zdjęć z fragmentów obszarów sąsiadów Polski, istnieją podstawy aby przypuszczać, że danych, które można pobrać z serwisu *Flickr*, pochodzących z obszarów Niemiec, Czech lub Austrii, może być znacznie więcej, niż tych pochodzących z obszaru Polski. Z całego obszaru Polski udało się zebrać około czterysta tysięcy zdjęć. Gdy przypadkiem pobrane zostały dane również pochodzące z fragmentów państw sąsiadujących z Polską, zdjęć w bazie danych było ponad milion i dwieście tysięcy. Należy pamiętać, że nie były to duże wycinki tych krajów. Sugeruje to, że efektywność prowadzenia tego typu badań może być ze względu na większą popularność sieci społecznościowych znacznie wyższa w krajach zachodniej Europy.

6. Wnioski i podsumowanie

Udało mi się zrealizować wszystkie założenia i cele stawiane na początku pracy, czyli stworzyć zestaw narzędzi, który umożliwiłby przeprowadzenie badań przy wykorzystaniu danych geolokalizacyjnych z sieci społecznościowej. W trakcie pracy napotkałem wiele nowych problemów, których pierwotnie się nie spodziewałem, a których samodzielne rozwiązanie było bardzo rozwijające. Efektem włożonej przeze mnie pracy jest łatwe do rozbudowy, działające narzędzie o dużym potencjale badawczym.

Pracując nad narzędziami będącymi bezpośrednim celem tej pracy przekonałem się jak ważne jest poświęcenie odpowiednio dużej ilości czasu i energii w odpowiednie zaplanowanie systemu komputerowego. Czas zainwestowany w rozważną analizę środowiska, a także w całościowe przemyślenie problemu który trzeba rozwiązać, potrafi się bardzo szybko zwrócić. Gdyby nie fakt, że tworzone przeze mnie narzędzia od początku miały być jak najłatwiejsze w rozbudowie i utrzymaniu, co automatycznie przełożyło się na poświęcenie znacznie większej ilości czasu na planowanie architektury rozwiązania, prawdopodobnie musiałbym w trakcie pracy wielokrotnie zaczynać pracę prawie od początku, przebudowując całkowicie aplikację. Ta sama uwaga dotyczy doboru technologii — warto zainwestować czas w poważną analizę środowiska, aby oszczędzić sobie powrotu do tego samego punktu, gdy okaże się, że pośpiesznie wybrana technologia nie nadaje się do realizacji założonego celu. Dzięki tego typu analizie poznałem możliwości skorzystania z przestrzennych baz danych, które okazały się być w moim przypadku idealnym rozwiązaniem wielu problemów.

Ważnym aspektem, o którym trzeba pamiętać przy prowadzeniu badań opartych na danych z sieci społecznościowych, jest fakt, że są to badania bardzo niekonwencjonalne i w dużej mierze zależą od czynników, na które nie ma się wpływu, takich jak popularność tego typu internetowych serwisów, czy ograniczone możliwości pobierania danych ze stron o bardziej zamkniętym dostępie. Z drugiej strony, takie niekonwencjonalne źródła informacji oferują potencjał, którego nie ma żadne inne źródło — to właśnie sprawia, że są takie atrakcyjne. Na pewno warto inwestować w rozwiązania nowe i jeszcze niezbadane, gdyż to właśnie one od zawsze były napędem wszelkiego rozwoju i postępu.

Wnioskiem płynącym z przeprowadzonej przeze mnie prostej analizy zebranych danych jest to, że bazując na danych pobranych z sieci społecznościowej, bardzo trudno jest zapewnić ich odpowiednią jakość i ilość. W przypadku, gdy dane są słabej jakości, bardzo trudno jest przeprowadzić jednoznaczne badania. Trzeba także pamiętać, że w zależności od celów, jakie ma przed sobą postawione badacz, drogi

postępowania podczas analizy mogą być zupełnie różne. Mimo wszystko — ostatecznie zawsze najważniejsze są umiejętności, doświadczenie i intuicja badacza. Żadne narzędzie nie jest w stanie tego zastąpić, w końcu przecież one wszystkie służą mu tylko pomocą. W dobrych rękach mogą rozwinąć swój prawdziwy potencjał, w nieodpowiednich natomiast — będą praktycznie bezużyteczne.

Literatura

[1] Dustin T. Duncan, Jared Aldstadt, John Whalen, Steven J. Melly, Steven L. Gortmaker. Validation of Walk Score® for Estimating Neighborhood Walkability: An Analysis of Four US Metropolitan Areas. *International Journal of Enviromental Research and Public Health*, 8(11), 2011.

[2] Universal McCann. Wave 7 - Cracking The Social Code, 2013. URL http://wave.umww.com/index.html.

[3] PewResearch Internet Project. Social Media Update 2014, Styczeń 2015. URL http://www.pewinternet.org/2015/01/09/social-media-update-2014/.

[4] International Telecommunication Union. The World in 2014: ICT Facts and Figures, 2014. URL https://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2014-e.pdf.

[5] *Standard of the Camera & Imaging Products Association, CIPA DC-008-Translation-2012, Exchangeable image file format for digital still cameras: Exif Version 2.3*. Camera & Imaging Products Association, Grudzień 2012.

Załączniki

A. Instrukcja użytkowania narzędzi

A.1. Przygotowanie środowiska uruchomieniowego

Kod źródłowy aplikacji, wraz ze skryptami przygotowującymi środowisko uruchomieniowe potrzebne do jej działania został umieszczony na płycie załączonej do pracy inżynierskiej. Aby skorzystać z przygotowanych narzędzi należy:

1. Zainstalować potrzebne do budowania aplikacji programy (między innymi *Apache Maven*), bazę danych *PostgreSQL* z nakładką *PostGIS*, oraz biblioteki potrzebne do kompilacji narzędzia *osm2pgsql* służącego do importowania danych z *OpenStreetMap* do bazy danych.
2. Skompilować program *osm2pgsql* — jest on udostępniony tylko w formie paczki kodów źródłowych, do użycia wymagana jest więc jego kompilacja. Jest załączony w oryginalnej formie na płycie dołączonej do pracy inżynierskiej.
3. Stworzyć w bazie danych schemat z tabelami i widokami wykorzystywanymi przez program.
4. Zaimportować dane przestrzenne dotyczące Polski z formatu *OpenStreetMap* do bazy danych przy użyciu narzędzia *osm2pgsql*.
5. Stworzyć dodatkowe indeksy i widoki korzystające z tabel powstałych po imporcie danych.
6. Zbudować program powstały w ramach tej pracy.

Procedura przedstawiona powyżej jest dość zawiła, dlatego w ramach uproszczenia powyższego procesu przygotowałem zestaw pomocniczych skryptów, które wykonują większość wymaganej pracy. Są one przygotowane dla systemu operacyjnego *Linux* i przetestowane na świeżej instalacji systemu *Ubuntu 14.04.1 LTS (64bit)*. Narzędzia mogą działać na każdym systemie, na którym da się uruchomić maszynę wirtualną *Java* oraz na którym może działać baza danych *PostgreSQL* z nakładką *PostGIS*, lecz skrypty automatyzujące proces przygotowane zostały tylko dla systemu *Linux*. Na innych systemach operacyjnych wymagane jest niestety ręczne przejście wszystkich kroków procedury.

Schemat przygotowania środowiska do działania programu na systemie operacyjnym *Linux* jest następujący:

1. Skopiowanie zawartości płyty do lokalnego systemu plików (wymagana będzie możliwość zapisu).
2. Przejście do katalogu „*scripts*”.

3. Uruchomienie skryptu *„prepare_enviroment.sh”*. Skrypt ten pobiera z publicznych repozytoriów i instaluje wszystkie programy i biblioteki pomocnicze, potrzebne do budowania i korzystania z programu stworzonego w ramach pracy. W tym kroku kompilowane jest także narzędzie *osm2pgsql*.

4. Uruchomienie skryptu *„prepare_database.sh”*. Skrypt ten tworzy bazę danych i uaktywnia w niej rozszerzenia przestrzenne udostępniane przez *PostGIS*. Nadaje uprawnienia do bazy użytkownikowi, który wykonuje skrypt.

5. *Uwaga — krok ten może trwać kilka godzin. Wymaga także około 20Gb wolnego miejsca na dysku (po procesie większość miejsca jest zwalniana).* Uruchomienie skryptu *„prepare-db-schema-and-import-osm.sh”*. Skrypt ten tworzy w bazie danych cały schemat tabel i widoków, wraz z indeksami i sekwencjami, a następnie importuje dane przestrzenne o terytorium Polski z załączonego do zestawu pliku w formacie *.osm* (są to dane wyeksportowane z *OpenStreetMap*). Po imporcie tworzy dodatkowe, pomocnicze indeksy na zaimportowanych tabelach, oraz widoki, które z nich korzystają.

6. Przejście do katalogu *„java”*. W katalogu tym znajdują się pliki źródłowe aplikacji.

7. Należy zbudować aplikację. W tym celu trzeba wykonać skrypt *„package.sh”*, który buduje aplikację za pomocą *Apache Maven*.

8. Aplikacja powinna zostać zbudowana i znajdować się w postaci pliku *.jar* w katalogu *„target”*.

9. Należy przenieść gotowy plik *.jar* do odrębnego katalogu. Do katalogu tego należy także skopiować katalog *„resources”* znajdujący się w głównym katalogu lokalnej kopii danych z płyty.

10. Należy w pliku *„resources/program.properties”* wypełnić dane dotyczące bazy danych. Domyślnie skrypty utworzyły w bazie danych konto bieżącemu użytkownikowi, z hasłem takim samym jak nazwa użytkownika.

11. Program jest gotowy do użytku.

A.2. Korzystanie z aplikacji

Korzystanie z przygotowanej w ramach tej pracy aplikacji jest bardzo proste. Ogólny schemat postępowania sprowadza się do:

1. Przygotowania pliku *„resources/tagi.txt”* z listą tagów, o które program będzie odpytywał sieć społecznościową w poszukiwaniu zdjęć. W pliku tym powinny znajdować się kolejne słowa, po jednym na linijkę.

2. Uruchomienia programu w trybie ściągania danych z sieci społecznościowej.
3. Uruchomienia programu w trybie synchronizacji danych przestrzennych. Proces ten może potrwać kilka minut, w jego wyniku pobrane z sieci społecznościowej dane będą odwzorowane w tabelach przestrzennych, które służą do analizy zdjęć.
4. Po ściągnięciu zadowalającej ilości danych można je wyświetlić na tle mapy uruchamiając program w trybie wyświetlania danych. Wymagana jest przed tym synchronizacja danych z kroku trzeciego.
5. Dane można eksportować do pliku *Shapefile* bądź *CSV* używając aplikacji w odpowiednim trybie.

Ważna uwaga: funkcje wyświetlania mapy, oraz eksportu danych wymagają wcześniejszej synchronizacji danych. Jest to zwłaszcza przydatne, ze względu na to, że podczas synchronizacji można określić listę tagów (oddzielonych spacjami), z którymi muszą być powiązane zdjęcia w bazie danych, aby być brane pod uwagę podczas analiz. Można dzięki temu ograniczyć analizy do danych powiązanych z hasłem „*ptaki*", lub „*architektura*". Kolejne tagi są oddzielone operatorem *OR* (czyli pod uwagę brane będą wszystkie dane powiązane z którymkolwiek z podanych na liście tagów).

Synchronizację można przeprowadzać wielokrotnie. Za każdym razem nadpisywane będą tylko dane w tabelach pomocniczych przechowujących dane przestrzenne. Oryginalne dane zostają w bazie danych nienaruszone.

Poniżej znajduje się schemat wywoływania programu (Listing 6).

B. Pełne dane *EXIF* dla zdjęcia z rozdziału 3.2.5

Pełne metadane *EXIF* dla przykładowego zdjęcia znajdują się na Listingu 7.

C. Lista pięćdziesięciu najpopularniejszych tagów dla zebranych w ramach testu danych

W tabelach C.1 oraz C.2 znajduje się pięćdziesiąt najpopularniejszych haseł, które udało się zebrać podczas testowania programu.

```
HELP_MESSAGE=Argumenty programu:

  -n
  --networks
    Wypisuje wszystkie dostepne serwisy spolecznosciowe, z ktorych mozna zbierac dane.

  -r
  --regions
    Wypisuje wszystkie dostepne regiony do przeszukiwania.

  -f <kod regionu> <kod serwisu spolecznosciowego>
  --fetch <kod regionu> <kod serwisu spolecznosciowego>
    Odpytpuje serwis spolecznosciowy o podanym kodzie na temat kolejnych tagow
    znajdujacych sie w pliku tagi.txt i zapisuje pobrane dane do bazy danych. Plik
    konfiguracyjny bazy danych znajduje sie w pliku program.properties.

  -s <kod regionu> [lista tagow]
  --synchronize-geometry-and-stats <kod regionu> [lista tagow]
    Synchronizuje dane geograficzne w pomocniczych tabelach w bazie danych dla danego
    regionu. Wymagane przed eksportem CSV oraz wyswietlaniem mapy. Te dwie funkcje
    korzystaja z synchronizacji zapewnionej przez synchronizacje i beda braly pod uwage
    tylko zdjecia polaczone z zadana lista tagow. Lista tagow jest opcjonalna. Jej
    niepodanie oznacza branie pod uwage wszystkich tagow dostepnych w bazie.

  -e <kod regionu> <plik wejsciowy> <plik wyjsciowy>
  --export <kod regionu> <plik wejsciowy> <plik wyjsciowy>
    Dopasowuje dane zebrane z sieci spolecznosciowych do danych wejsciowych z
    zewnetrznego zrodla w formacie CSV i eksportuje dopasowane dane jako plik CSV.
    Wymaga wczesniejszej synchronizacji tego samego regionu aby wyswietlic poprawne dane
    .

  -m [filtr zdjec]
  --map [filtr zdjec]
    Wyswietla mape, wraz z nalozonymi na nia lokalizacjami zsynchronizowanych zdjec.

  -S <sciezka do pliku wynikowego>
  --shp <sciezka do pliku wynikowego>
    Eksportuje wszystkie zdjecia z bazy danych do formatu Shapefile (.shp), tak aby dalo
     sie je wyswietlic na mapie w zewnetrznych programach do analizy.
```

Listing 6. Schemat wywoływania programu.

```
exif:ColorSpace: 1
exif:ComponentsConfiguration: 1, 2, 3, 0
exif:Compression: 6
exif:DateTime: 2013:08:10 21:20:13
exif:DateTimeDigitized: 2013:08:10 21:20:13
exif:DateTimeOriginal: 2013:08:10 21:20:13
exif:ExifImageLength: 1920
exif:ExifImageWidth: 2560
exif:ExifOffset: 238
exif:ExifVersion: 48, 50, 50, 48
exif:ExposureBiasValue: 0/2
exif:ExposureMode: 0
exif:ExposureProgram: 3
exif:ExposureTime: 125000/1000000
exif:Flash: 0
exif:FlashPixVersion: 48, 49, 48, 48
exif:FNumber: 260/100
exif:FocalLength: 354/100
exif:GPSAltitude: 0/1
exif:GPSAltitudeRef: 0
exif:GPSDateStamp: 2013:08:10
exif:GPSInfo: 750
exif:GPSLatitude: 50/1, 20/1, 42993/1000
exif:GPSLatitudeRef: N
exif:GPSLongitude: 15/1, 55/1, 44190/1000
exif:GPSLongitudeRef: E
exif:GPSMapDatum: WGS 84
exif:GPSProcessingMethod: NETWORK
exif:GPSStatus: V
exif:GPSTimeStamp: 19/1, 20/1, 4/1
exif:GPSVersionID: 2, 2, 0, 0
exif:ImageLength: 1920
exif:ImageWidth: 2560
exif:InteroperabilityIndex: R98
exif:InteroperabilityOffset: 720
exif:InteroperabilityVersion: 48, 49, 48, 48
exif:ISOSpeedRatings: 400
exif:Make: SAMSUNG
exif:MaxApertureValue: 276/100
exif:MeteringMode: 2
exif:Model: GT-I8190N
exif:Orientation: 1
exif:ResolutionUnit: 2
exif:SceneCaptureType: 0
exif:Software: I8190NXXAMB1
exif:WhiteBalance: 0
exif:XResolution: 72/1
exif:YCbCrPositioning: 2
exif:YResolution: 72/1
```

Listing 7. Pełne wyeksportowane metadane *EXIF* przykładowego zdjęcia.

Tab. C.1. Pierwsze dwadzieścia pięć najpopularniejszych tagów.

NR	*NAME*	*PHOTOS*
1	poland	209663
2	polska	120633
3	warsaw	48819
4	canon	36995
5	warszawa	32565
6	krakow	28393
7	europe	27672
8	canonefs18135mmf3556is	26864
9	canoneos550d	26855
10	polen	25488
11	railway	23596
12	kraków	23578
13	pkp	23520
14	rail	22092
15	railroad	21887
16	architecture	19388
17	square	18765
18	station	18748
19	wrocław	15981
20	iphoneography	15657
21	squareformat	15523
22	lowersilesia	15466
23	uploaded:by=instagram	15379
24	instagramapp	15375
25	dolnyśląsk	15080

Tab. C.2. Drugie dwadzieścia pięć najpopularniejszych tagów.

NR	*NAME*	*PHOTOS*
26	dolnośląskie	14761
27	cracow	14741
28	building	14189
29	polonia	14099
30	nature	12295
31	wroclaw	12118
32	city	11936
33	geotagged	11655
34	europa	10753
35	old	9917
36	winter	9448
37	nikon	9429
38	people	9257
39	street	9186
40	pologne	9088
41	train	8956
42	night	8400
43	łódź	8369
44	bw	8339
45	church	8332
46	gdansk	8299
47	tracks	8223
48	urban	7918
49	landscape	7890
50	timelapse	7742