Webowy system do automatycznego gromadzenia korpusów tekstu z Internetu i ich ręcznej anotacji meta danymi

Tytuł w języku angielskim: Web-based System for Gathering Text Corpora from Internet and Facilitating Manual Annotation of the Corpora

Spis treści

[Dział I 4](#_Toc452215406)

[Streszczenie 4](#_Toc452215407)

[Synopsis 4](#_Toc452215408)

[Wstęp: Cel, motywacje, ograniczenia 5](#_Toc452215409)

[Motywacje 5](#_Toc452215410)

[Cel pracy 5](#_Toc452215411)

[Ograniczenia pracy 6](#_Toc452215412)

[Kluczowe pojęcia i terminy używane w tej pracy 7](#_Toc452215413)

[Korpus językowy 7](#_Toc452215414)

[Anotacja 7](#_Toc452215415)

[Crawler 7](#_Toc452215416)

[Drzewo DOM 7](#_Toc452215417)

[Selektor 7](#_Toc452215418)

[Algorytm pozycyjny 7](#_Toc452215419)

[CCL 7](#_Toc452215420)

[Premorph 7](#_Toc452215421)

[Inforex 8](#_Toc452215422)

[Corpo-grabber 8](#_Toc452215423)

[WCRFT 9](#_Toc452215424)

[Blog Reader 9](#_Toc452215425)

[Przegląd literatury 10](#_Toc452215426)

[Internet Documents: A Rich Source for Spoken Language Modeling 10](#_Toc452215427)

[Efﬁcient Web Crawling for Large Text Corpora 10](#_Toc452215428)

[A DOM Tree Alignment Model for Mining Parallel Data from the Web 11](#_Toc452215429)

[Dział II 12](#_Toc452215430)

[Corpo-grabber 12](#_Toc452215431)

[Dlaczego skrypty PHP? 12](#_Toc452215432)

[Elementy 12](#_Toc452215433)

[Interfejs 13](#_Toc452215434)

[Działanie 13](#_Toc452215435)

[Badanie 13](#_Toc452215436)

[Teoria 13](#_Toc452215437)

[Praktyka 13](#_Toc452215438)

[Dział III 14](#_Toc452215439)

[Wyniki badań 14](#_Toc452215440)

[Wnioski z badań 14](#_Toc452215441)

[Podsumowanie 14](#_Toc452215442)

# Dział I

## Streszczenie

Celem tej pracy jest opisanie rozwiązania problemu, jakim jest skonstruowanie narzędzia do ekstrakcji korpusów lingwistycznych z zasobów internetowych, jak na przykład blogi, strony informacyjne, itp. Narzędzie to ma być przystępne dla użytkownika nieobytego w technologiach informatycznych. Przedstawię jakie rozwiązania do tej pory się pojawiły, moją propozycja rozwiązania problemu oraz analizę, czy moje rozwiązanie faktycznie może być aplikowane do tego problemu.

W ramach rozwiązania problemu postanowiłem podejść do niego z innej strony niż zwykle. Mając doświadczenie w tworzeniu stron internetowych wiem, że jeżeli rozpatrujemy strukturę drzewiatą wielu stron internetowych z tej samej domeny, to konkretne elementy, które nas interesują, są zazwyczaj tymi samymi gałęziami w drzewie strony. W związku z tym chciałem sprawdzić teorię, czy można dokonywać ektrakcji tekstu ze stron internetowych oznaczając konkretne fragmenty drzewa strony (DOM). Badanie potwierdzające moją teorię przeprowadziłem porównując moją metodę ekstrakcji tekstu z metodą konkurencyjną, która polega na ekstrakcji elementów ze strony internetowej przy pomocy selektorów CSS.

### Synopsis

In this paper I shall present a solution to the problem of constructing tool for extracting linguistic corpora from internet resources, like blogs, informational sites, etc. The tool must be handy for a user that isn’t common with informational technologies. I shall present what kind of solutions have been created, my solution to the problem, and analysis if my solution can actually be used to resolve this problem.

Regarding my problem, I decided to resolve it from different approach, compared to what was used before. Having experience in web design, I know that when we interpret many web pages from the same domain as tree structure, specific elements that we could be interested in are usually the same branches in these trees. Because of that I decided to test my theory, if it will be possible to extract text from web pages marking certain tree site parts (DOM). The research that confirmed my theory were done by comparison to another method, that involves in extracting elements of web page using CSS selectors.

## Wstęp: Cel, motywacje, ograniczenia

### Motywacje

Analiza syntaktyczna języka jest jedną z ważnych dziedzin informatyki. Przede wszystkim może nam służyć do analizy części mowy, zapis (może być różny, w zależności od akcentu), regularności, kolejności słów w zdaniu, znaczenia i problemów związanych z językiem. Tymi i innymi problemami zajmuje się dział nauki zwany lingwistyką, ale również lingwistyka informatyczna i inżynieria języka naturalnego.

Lingwistycy do analizy języka pisanego - czy mówionego - potrzebują mieć materiały do analizy. Często są to nagrania audio, ich transkrypcje, książki, czasopisma, nagrania video i inne. Do celów analizy przy pomocy narzędzia informatycznego potrzebna jest ich wcześniejsza cyfryzacja. Nagrania audio muszą być transkrybowane na medium cyfrowe, transkrypcje w formie analogowej muszą zostać przeanalizowane przy pomocy programu do optycznej interpretacji tekstu (Optical Character Recognition), tak samo czasopisma czy książki. Jednakże, cyfryzacja to nie wszystko: często typowe formaty transkrypcji danych są niewystarczające. Do tego celu teksty w formie cyfrowej muszą zostać przekształcone do formatu pliku zwanego korpusem.

Korpus językowy to skompilowany zbiór danych tekstowych dostępnych w formie elektronicznej, stanowiący materiał do badań. Jest zawsze w formie cyfrowej, czasem ten plik posiada również wstępną analizę syntaktyczną tekstu w nim zawartego. Politechnika Wrocławska opracowała własny format pliku obsługującego korpusy. Nazywa się on CCL i oprócz przechowywania korpusów, przechowują również anotacje istotne w procesie analizy lingwistycznej.

Lingwistyka informatyczna boryka się z wieloma problemami, jest to np. tworzenie modelu języka w oparciu o ciągle zmieniające się zasady. Jednym z nich jest pozyskiwanie materiałów do analizy, a przede wszystkim ich trudnodostępność. Jednakże jest miejsce, w którym są duże zasoby tekstu do analizy lingwistycznej, z której polscy naukowcy do tej pory nie czerpali bardzo. Jest nią internet.

Nie jest problemem pozyskiwanie korpusów z internetu. Każdy może wejść na dowolną stronę internetową i pobrać jej zawartość, wyszczególnić tekst i przekonwertować na format korpusu. Problemem jest natomiast pozyskanie dużych ilości tekstu w opraciu o pojedynczy, ogólny adres internetowy. Praca ta ma na celu przybliżenie algorytmów, które rozwiązują ten problem.

Praca ta również ma na celu przyjrzenie się konkurencyjnym algorytmom, które służą do gromadzenia korpusów językowych ze stron internetowych. Mam nadzieję, że dzięki tej pracy przyszłe pokolenia naukowców z dziedziny lingwistyki będą mogli korzystać z mojego narzędzia bez problemów, oraz że będą rozumieli jego działanie.

### Cel pracy

Celem pracy jest opracowanie systemu – narzędzia badawczego – do gromadzenia korpusów tekstowych z internetu, gdzie korpus tekstowy to duży zbiór tekstów gromadzonych w celach badawczych w ramach różnych dziedzin nauki. System powinien pracować automatycznie, ale pod kontrolą naukowca. System powinien umożliwić zdefiniowanie zakresu pobieranych danych, wydobycie tekstu wraz z jego semantyczną strukturą z dokumentów (np. z podziałem na akapity, nagłówki itd.), zapis do stosowanych formatów korpusowych, w tym formatu CCL opracowanego w Grupie Naukowej G4.19 PWr. oraz późniejszy opis tekstów meta-danymi na różnych poziomach, np. dokumentów, zdań, słów itd.

### Ograniczenia pracy

W związku z celem pracy opracowane jest narzędzie, które automatycznie pobiera stronę internetową, natomiast pod kontrolą naukowca jest element oznaczania danych, które go interesują na stronie.

Ograniczyłem się jedynie do zdefiniowania nowej metody analizowania strony internetowej (pozycyjnej). Porównuję to podejście z konkurencyjnym podejściem (według selektorów). W ramach badania analizuję oba podejścia na przykładzie wybranych blogów internetowych, oraz dwa podejścia mieszane. Badania ograniczyłem do ekstrakcji następujących elementów:

1. Notki – element opisujący wszystkie niżej wymienione elementy
2. Autor, tytuł, data wpisu i tekst wpisu

Do ekstrakcji tekstu uznałem, że element notki jest konieczny do wyznaczenia w celu wykrywania nieprawidłowości w strukturze.

Do badania wybrałem około 40 blogów i stron internetowych, które najpierw pobierałem, a potem ręcznie oznaczałem. Blogi wybierałem samodzielnie z listy stron, które były problematyczne przy algorytmie konkurencyjnym.

Narzędzie opracowywałem na platformie Windows, w związku z tym, część narzędzi skryptowych, jak skrypty linuxowe opracowane przez pracowników Politechniki Wrocławskiej, nie mogą zostać przeze mnie użyte i muszę wówczas opracować alternatywny algorytm.

## Kluczowe pojęcia i terminy używane w tej pracy

### Korpus językowy

Potrzebne fajne źródło słownikowe

### Anotacja

Potrzebne fajne źródło słownikowe

### Crawler

Potrzebne fajne źródło słownikowe

### Drzewo DOM

Potrzebne fajne źródło słownikowe

### Selektor

Potrzebne fajne źródło słownikowe

### Algorytm pozycyjny

Jest to opracowane przeze mnie podejście do analizy strony internetowej. Zamiast przedstawiać każdy element na stronie jako selektor o znanym typie, klasie bądź identyfikatorze, oznaczam każdy element jako ścieżka, jaką trzeba przebyć od korzenia do tego elementu. Pozwala to podejście definiowanie łatwych i jednoznacznych reguł w przypadku:

1. Nieistniejących nazw klas bądź identyfikatorów dla konkretnych elementów
2. Gdy element jest tej samej klasy lub (co jest błędne według W3C) ma ten sam identyfikator w drzewie
3. Gdy tekst jest oddzielony selektorami nowej linii („br”) a interesująca nas dana jest w jednym akapicie lub drugim

Ale o tych zaletach więcej opiszę we wnioskach z badań.

### CCL

CCL jest to format korpusowy opracowany na Politechnice Wrocławskiej. CCL jako format powstał w opraciu o format XCES (który w dużej mierze służy również do gromadzenia i anotacji korpusów lingwistycznych). Jest to plik tekstowy w formacie XML. Poza przechowywaniem tekstu źródłowego, format ten może przechowywać interesujące lingwistyków własności tekstu, takie jak:

1. Podziały na akapity i zdania
2. Podziały na tokeny i informacje bez-spacjowe
3. Anotacje morfosyntaktyczne
4. Anotacje do styli fragmentów (chunk-style) z możliwymi dyskontynuacjami
5. Nagłowki syntaktyczne anotacji
6. Właściwości tokenów – a co za tym idzie – anotacji

<http://nlp.pwr.wroc.pl/redmine/projects/corpus2/wiki/CCL_format>

### Premorph

Istnieje wiele form pośrednich pomiędzy dowolnym dokumentem zawierającym tekst a formatem CCL. Można zwykły dokument tekstowy podzielić na akapity i zapisać w formacie tekstowym. Można dokument podzielony na akapity zapisać w formacie html (z podziałem na akapity przy pomocy selektora „p”). Istnieje jeszcze format premorph, z którego korzystałem.

Nazwa premorph pochodzi z właściwości, jaką ma ten format: jest to plik XML sformatowany w podobny sposób, jak pliki CCL z jedną różnicą: nie ma podziału na słowa i nie ma anotacji morfo-syntaktycznej. Plik premorph różni się od pliku CCL następującymi elementami

1. Podział na akapity
2. Brak podziału na zdania, tokeny.
3. Nie zawiera jakichkolwiek anotacji morfo syntaktycznych.

A co je łączy?

1. Zapis w XML-u
2. Podział na listy chunków (akapity)
3. Nazwa akapitu (chunku)

### Inforex

Inforex jest narzędziem utworzonym na Politechnice Wrocławskiej. Służy ono do analizy morfo syntaktycznej korpusów w formacie CCL, oraz do ręcznej anotacji elementów mowy jak nazwy jednostkowe, anafory, nadawanie sensu słowom niejednoznacznym i wszelkiego rodzaju relacje pomiędzy jednostkami. System również wspiera ręczne czyszczenie tekstu oraz automatyczną analizę, w której się zawiera segmentacja tekstu, analiza morfo syntaktyczna, selekcja słów do późniejszego nadania znaczenia w przypadku słów niejednoznacznych.

Marcińczuk, Michał; Kocoń, Jan; Marcin, Ptak and Kaczmarek, Adam, 2010, *Inforex*, CLARIN-PL digital repository, <http://hdl.handle.net/11321/13>.

Nie wykluczone, że będę wspominał niejednokrotnie o tym systemie na przestrzeni tego dokumentu, ze względu na to, że główną inspiracją do napisania tej pracy był właśnie system Inforex. Z tego też powodu moje narzędzie będzie konwertowało dokumenty do formatu CCL.

### Corpo-grabber

Jest to narzędzie, podobnie jak moje, służące do automatycznego gromadzenia korpusów internetowych z Internetu. Korzysta ono z takich narzędzi jak:

1. Httrack – do pozyskiwania stron internetowych
2. Fdupes – Do deduplikacji tekstów identycznych
3. Onion – Do deduplikacji tekstów podobnych.
4. WCRFT – Do konwersji tekstu podzielonego na akapity do formatu CCL. Więcej o nim w podrozdziale.

Moje narzędzie opierałem na corpo-grabberze, nawet je podobnie nazywałem. W początkowej fazie projektu korzystałem zarówno z httracka i narzędzi deduplikacyjnych takich samych jak w tym projekcie, jednakże z czasem i doświadczeniem zacząłem korzystać z innych.

### WCRFT

Wrocław CRF Tagger – jest to narzędzie stworzone na Politechnice Wrocławskiej, tak jak powyższe narzędzia, do analizy morfo syntaktycznej tekstu. Korzysta z algorytmu CRF: Conditional Random Fields, który służy do rozpoznawania wzorców i w nauczaniu maszynowym. Ma szereg różnych zastosowań, jednym z nich jest łatwa konwersja tekstów w języku polskim na korpusy lingwistyczne, z której korzysta moje narzędzie w formie finalnej.

<https://en.wikipedia.org/wiki/Conditional_random_field>

<http://nlp.pwr.wroc.pl/redmine/projects/wcrft/wiki>

### Blog Reader

Jest to kolejne narzędzie opracowane na Politechnice Wrocławskiej, służące do ekstrakcji tekstu ze stron internetowych już pobranych. Korzysta z podejścia selektorowego do ekstrakcji tekstu. Podejście jest proste: definiujemy elementy, jakie chcemy pobrać ze strony i nadajemy im selektor CSS, według którego należy wyszukiwać tych elementów, które nas interesują. Informatycy są w stanie bez problemu wyznaczyć wystarczające reguły do opracowania selektora, jednakże osoby nieobyte w technologiach IT nie były w stanie ich zdefiniować, co było wielką wadą tego programu.

Moje narzędzie opierałem na Blog Readerze do celów badawczych. Była to najlepsza znana mi metoda ekstrakcji tekstu, gdzie chcemy pozyskać konkretną treść ze stron internetowych.

## Przegląd literatury

Literatura poruszająca problem pozyskiwania korpusów tekstu z internetu jest uboga, jednakże jest wiele prac naukowych dotykających problematyki zautomatyzowanego pobierania zasobów internetowych, oraz dotyczących wykorzystania korpusów tekstu w różnych dziedzinach wiedzy. W tym rozdziale postaram się przedstawić najistotniejsze źródła, z których korzystałem przy pracy, opiszę co zawierają oraz opiszę stosunek tych prac z moją pracą.

### Internet Documents: A Rich Source for Spoken Language Modeling

W tej pracy zaproponowano metodologię pozyskiwania dokumentów internetowych w celu pozyskania tekstów do modelowania języka mówionego. Aktualne modele językowe często są oparte na tekstach pisanych i/lub nadużywanym Czarnoksiężniku z krainy Oz, albo na eksperymentalnych dialogach. Przy pomocy tekstów pozyskanych z internetu, potem przeczytanych przez eksperymentatorów, a następnie przeanalizowanych przez algorytmy rozpoznawania mowy otrzymano 15% wzrost skuteczności w rozpoznawaniu słów.

Moja praca będzie w dużej mierze będzie opierała się na pozyskiwaniu tekstów z internetu. Moim zadaniem jednak będzie przekształcanie ich na korpusy tekstów w formacie CCL, gdzie praca ta opierała się głównie na pozyskaniu tekstów do celów późniejszego ich odczytania. Analiza morfo-syntaktyczna byłaby im niepotrzebna.

### Efﬁcient Web Crawling for Large Text Corpora

Praca ta skupia się na efektywnym pobieraniu zasobów internetowych. Autorzy proponują różne rozwiązania, które usprawniają proces pobierania, ale również umożliwiają przy pomocy różnych metod umożliwić łatwą ekstrakcję tekstu. Autorzy mieli dwie główne motywacje: pobieranie danych ze źródeł zawierających najwięcej treści oraz możliwe unikanie duplikatów danych.

**Ilość czy liczba? \/**

Pierwszy problem, czyli uzyskiwanie lepszych wyników pobierań, rozwiązali poprzez napisanie własnego crawlera, którego nazwali pajączek (spiderling). Zdefiniowali zależność pomiędzy ilością danych pobranych, ilością danych istotnych (czyli „gołego" tekstu) oraz ilością dokumentów pobranych. Wyznaczenie tej zależności umożliwiło im zredukowanie czasu pobierania danych oraz zwiększyło stosunek danych istotnych do danych pobranych (yield rate) poprzez odrzucanie domen, z których nie dało się wyekstraktować wystarczającej ilości danych.

Drugi problem, jakim jest unikanie duplikatów danych, uzyskali przez napisany przez nich program – jusText. Dokonywali deduplikacji w dwóch momentach: w momencie pobierania danych i po ekstrakcji tekstu. Nadawali dokumentom sumy kontrolne, które porównywali z pozostałymi dokumentami. W przypadku duplikatu, jeden z dokumentów usuwali a sumę zapisywali. Korzystali również z programu onion do deduplikacji tekstów zbliżonych.

W moim rozwiązaniu korzystam z gotowych rozwiązań do crawlowania. Nie potrzebuję wykrywać, czy strona zawiera treść. Zakładam, że naukowiec korzystający z mojego narzędzia wie jakiej treści oczekuje ze strony pobieranej, i że pobiera ją nie bez celu. Dlatego zależności wyznaczone przez autorów nie będą mnie interesowały.

Drugie rozwiązanie z ich pracy jest dla mnie przydatne. Duplikaty danych występują niemalże na każdej stronie internetowej. Choćby przykładem niech będą notki na blogach, które pojawiają się zarówno jako osobne strony, na stronach z komentarzami, oraz na stronach archiwum. Proces deduplikacji zatem jest potrzebny dla mnie po pobieraniu danych, ale również przed, jak i po analizie danych.

### A DOM Tree Alignment Model for Mining Parallel Data from the Web

Pracę tę znalazłem po sformułowaniu mojego problem. Opisuje ona inny problem niż mój na podstawie podobnych do moich założeń.

W tym dokumencie jest postawiona teza, że strony internetowe dwujęzykowe często posiadają taką samą strukturę strony i linków, niezależnie od języka, który się wybierze. Autorzy zbadali i zauważyli, że 10% spośród 150 tysięcy stron internetowych z domeny .de jest dwujęzykowa. W związku z tymi obserwacjami, powstało wiele crawlerów służących do pozyskiwania danych dwujęzycznych. Między innymi twórcy tej pracy, którzy zastosowali podobieństwo stron internetowych do ekstrakcji zdań w obu językach.

Ja również korzystam z tego podejścia. Jest ono bardzo wygodne i zastanawiające jest, że nikt nie zrobił tak prostego badania, jakim jest sprawdzenie jednolitości struktury stron internetowych.

# Dział II

## Corpo-grabber

Corpo-grabber jest moim podejściem do tematu, jakim jest rozwinięcie istniejącej już architektury na Politechnice Wrocławskiej. Moje narzędzie jest webowym skryptem w PHP. Ma ono za zadanie przy dyrektywach użytkownika pobrać wybraną stronę internetową, następnie po zakończonym procesie pobierania ma dokonać wstępnej deduplikacji stron internetowych, po czym wyświetla pobrany projekt. Następnie zadaniem użytkownika jest dla przykładowej pobranej strony internetowej zdefiniowanie interesujących go elementów strony. Po tym, mój program po głuższej analizie prezentuje wyniki pobierania.

### Dlaczego skrypty PHP?

Odpowiedź jest bardzo prosta. Podczas analizy strukturalnej strony internetowej potrzebna jest idealna wizualizacja strony internetowej. W związku z tym potrzebne jest narzędzie, które umie idealnie prezentować dane. Jest nią przeglądarka internetowa. W związku z tym warstwa prezentacyjna strony internetowej powinna być realizowana przy pomocy skryptów PHP.

Jednakże fragment części logicznej strony jest również zaprogramowana przy pomocy skryptów PHP. Wynika to z bardzo prostego faktu – do języka PHP napisanych zostało wiele interesujących skryptów, które są w stanie analizować drzewa DOM. Korzystam z nich i są bardzo pomocne. Jednakże do celów optymalizacyjnych powinno się te narzędzia napisać w zewnętrznym skrypcie, jak w języku Java (dla pracy na różnych systemach) czy Python (by konturować trendy na Politechnice Wrocławskiej).

### Elementy

Program dzieli się na parę elementów, które są wyznaczone przez interfejs:

1. Część dotycząca pobierania
2. Część dotycząca oznaczania stron

Z czego część druga ma jeszcze następujące podczęści:

1. Część dotyczącą wyboru podstrony
2. Podgląd podstrony
3. Widok drzewa podstrony
4. Analiza drzewa podstrony

Program korzysta z takich zewnętrznych programów jak:

1. WGET – program służący do pobierania stron internetowych
2. PHPHtmlParser – wtyczka PHP służąca do analizy drzewa DOM
3. Simple MVC - Prosta biblioteka MVC do PHP.

WGET jest wykorzystywany tylko w części pierwszej. Natomiast PHPHtmlParser jest wykorzystywany w części drugiej w podczęści 3 i 4. Sens takiego wykorzystania opiszę w podrozdziale „Działanie”.

### Interfejs

W ciągu całego procesu tworzenia oprogramowania korzystałem z wielu różnych podejść do tworzenia strony internetowej. Chciałem z początku podejść do tematu przy pomocy bardzo ubogiego interfejsu. Chciałem po prostu umożliwić wybieranie strony internetowej do pobrania i wyświetlić wynik. Jednakże wkrótce po tym okazało się, że nie da się spełnić wszystkich oczekiwań na prostym interfejsie. W związku z tym skorzystałem z biblioteki MVC jaką jest Simple MVC. Umożliwiła mi ona proste tworzenie interfejsu i prostej logiki.

W związku z tym, postanowiłem nagłówek przeznaczyć na łącza do podstawowych części strony:

1. Strona główna
2. Pobieranie pojedynczej strony
3. Pobieranie wielu stron z jednej domeny (Pobieranie zaawansowane)
4. Anotacja strony pobranej

#### Strona główna

Zawiera krótką adnotację dotyczącą przeznaczenia strony internetowej. Opisuje to, co możemy znaleźć na konkretnych podstronach.

#### Pobieranie pojedynczej strony

Z początku myślałem, że pobieranie pojedynczej strony internetowej będzie czymś istotnym na stronie internetowej. W związku z tym, po wprowadzeniu adresu strony algorytm automatycznie pobierał wybraną stronę internetową a następnie

### Działanie

## Badanie

### Teoria

### Praktyka

# Dział III

## Wyniki badań

### Wnioski z badań

## Podsumowanie

## Źródła

<http://dl.acm.org/citation.cfm?id=775182>

Możliwe źródła:

<http://www.teachingenglish.org.uk/article/analysing-language> - elementy języka

<http://korpusy.net/> - definicja korpusu

<http://nlp.pwr.wroc.pl/redmine/projects/corpus2/wiki/CCL_format> - o formacie CCL

<http://www.cs.utah.edu/nlp/readinglist/BaroniB04.pdf> - definicje dot. korpusów

<https://lindat.mff.cuni.cz/repository/xmlui/handle/11858/00-097C-0000-000D-F696-9> - jusText, Pomikalek Jan

Alternatywy do corpograbbera i inforex:

<http://www.textise.net/>

<https://www.diffbot.com/>

<https://github.com/kohlschutter/boilerpipe> // fulltext extraction

<https://code.google.com/p/boilerpipe/wiki/Benchmarks>

<http://www.crummy.com/software/BeautifulSoup/>

<http://rodp.me/2015/how-to-extract-data-from-the-web.html>

<http://www3.nd.edu/~tweninge/pubs/WH_TIR08.pdf>

<https://hal.archives-ouvertes.fr/inria-00326150/document>

<https://books.google.pl/books?id=vFHOx8wfSU0C&printsec=frontcover&dq=text+mining&hl=en&sa=X&ved=0ahUKEwj3usnaldjLAhVI7xQKHRmpDqwQ6AEIODAD#v=onepage&q=text%20mining&f=false>