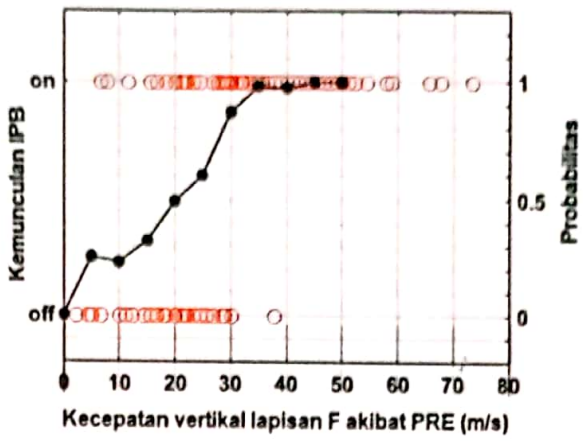
pengamatan IPB dan PRE menggunakan ionosonda di Chumphon, Thailand selama bulan Maret- April tahun 2011-2015. Pada Gambar 2, setiap lingkaran merah mengindikasikan hasil observasi di suatu malam dengan ataupun tanpa kemunculan IPB dan nilai kekuatan PRE pada malam tersebut. Lingkaran-lingkaran merah di posisi *on* di sumbu vertikal sebelah kiri menunjukkan kemunculan IPB, sedangkan posisi *off* menunjukkan tanpa kemunculan IPB. Nilai PRE setiap lingkaran merah ditunjukan pada skala horizontal. Malam-malam dengan kemunculan IPB banyak terjadi ketika nilai PRE lebih dari 30 m/s. Dari sampel yang diperoleh, hanya satu malam tanpa kemunculan IPB dengan nilai PRE > 30 m/s. Sebaliknya, pada malam-malam dengan nilai PRE < 30 m/s, IPB dapat terjadi ataupun tidak terjadi.



Gambar 2: Hubungan antara kekuatan PRE dan kemunculan IPB.

Lingkaran merah menunjukkan kecepatan vertikal lapisan F akibat PRE pada malam dengan kemunculan IPB (*on*) atau tanpa IPB (*off*). Pada tiap kelas kecepatan, dapat dihitung probabilitas kemunculan IPB yang direpresentasikan oleh titik hitam dengan nilai yang ditunjukkan pada sumbu di sebelah kanan.

Kurva hitam pada Gambar 2 menunjukkan hubungan kuantitatif kekuatan PRE dan kemunculan IPB. Nilai probabilitas kemunculan IPB sebagai fungsi dari PRE ditunjukkan oleh kurva tersebut dengan skala yang digambarkan oleh sumbu di sebelah kanan. Peluang kemunculan IPB lebih dari 0,8 saat nilai PRE > 30 m/s. Hal ini menunjukkan IPB hampir selalu terjadi ketika nilai PRE > 30 m/s. Bahkan, nilai probabilitas menjadi 1 (IPB pasti terjadi) saat nilai PRE lebih dari 40 m/s. Peluang kemunculan IPB berkurang seiring dengan menurunnya nilai PRE.

Rujukan

[1] Abadi, P., Otsuka, Y., Supriadi, S., and Olla, A., *Probability of ionospheric plasma bubble occurrence as a function of pre-reversal enhancement deduced from ionosondes in Southeast Asia*, disampaikan dalam ISAST VII/2019

ILMU DATA

Web Scraping

Mengikis data semi-terstruktur secara masif dan sistematis dari situs web

Oleh
R. Priyatikanto
Pussainsa LAPAN

Data seringkali dianggap sebagai *new oil* dari era modern karena data dianggap sebagai objek yang bisa diperjualbelikan dengan harga yang tidak murah. Namun, ada juga yang menekankan bahwa data tidak bisa dianalogikan secara persis dengan minyak. Data senantiasa diproduksi sementara minyak merupakan sumber daya alama yang terbatas kuantitasnya.

Dewasa ini, data dapat digali secara sistematis dari berbagai sumber. Piranti yang terkoneksi dengan internet menjadi corong data dan informasi yang dapat ditampung, dikelola, diolah, dan dianalisis dengan berbagai cara. Dunia maya adalah kolam untuk menjaring kumpulan informasi yang dapat dipandang sebagai data untuk analisis lebih lanjut. Dari kumpulan informasi yang tak terstruktur, ilmuwan data (*data scientist*) memanfaatkan metode statistik, *machine learning*, atau kecerdasan buatan untuk menyaringnya.

Pada artikel kali ini, akan dibahas satu metode ekstraksi data secara sistematis dari situs web. Metode ini sering disebut *web scraping* atau mengikis situs web.

Web scraping merupakan proses pengambilan data secara semi-terstruktur dari halaman web. Metode ini dapat mempermudah dan mempercepat proses akuisisi data dalam jumlah besar. Amat kontras bila dibandingkan pekerjaan yang dilakukan oleh asisten peneliti (mahasiswa atau fungsional litkayasa) beberapa tahun belakangan yang dengan sabarnya membuka halaman web dan mencatat data yang ditampilkan. Pekerjaan berbulan-bulan dapat dipersingkat menjadi proses *web scraping* yang hanya berlangsung

```
import requests, time, datetime as dtime
from bs4 import BeautifulSoup
dt = dtime.date(2016,1,1)
fo = open('chimera_scrape.dat', 'w')
for i in range(1100):
  tx = dt.strftime('%Y%m%d'); dt+= dtime.timedelta(1)
  url = 'https://solarmonitor.org/chimera.php?date=' + tx
  res = requests.get(url)
  sup = BeautifulSoup(res.text, 'html.parser')
  cen = sup.find_all(id='centroid')
  are = sup.find_all(id='area')
  bfi = sup.find_all(id='B_field')
  bfu = sup.find_all(id='B_flux')
  for j in range(len(cen)):
    te = tx + '_' + cen[j].get_text() + are[j].get_text()
       + bfi[j].get_text() + bfu[j].get_text()
    fo.write(te + '\n')
fo.close()
```

dalam hitungan menit.

Banyak perangkat yang telah dikembangkan untuk *web scraping*, diantaranya SCRAPEAPI, OCTOPARSE, AGENTY, PARSEHUB, dan SCARPY. Ada perangkat yang *handy* dan tidak menuntut kemampuan pemrograman komputer, ada pula yang memiliki banyak fitur yang dioperasikan melalui program komputer dengan bahasa tertentu. Pada artikel ini, akan diulas penggunaan REQUEST untuk melakukan *web scraping* parameter lubang korona dari situs `solarmonitor.org`.

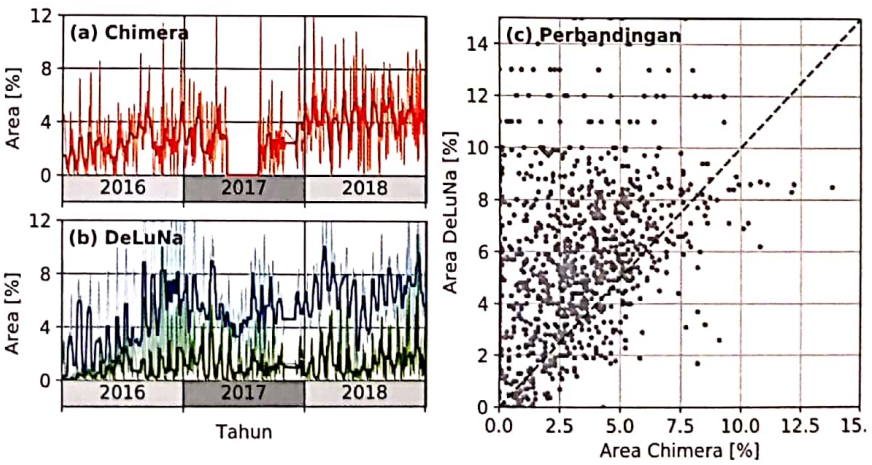
REQUESTS merupakan modul PYTHON untuk mengakses halaman web melalui *hypertext transfer protocol* (HTTP). Modul program ini relatif mudah untuk digunakan sepertihalnya SCRAPY. Sebuah program singkat dibuat untuk mengimplementasikannya. Selain REQUESTS, modul lain yang digunakan adalah modul TIME untuk pengaturan ritme kerja, modul DATETIME untuk perhitungan waktu dan kalender, serta modul BEAUTIFULSOUP untuk merapihkan hasil *scraping*.

Alur kerja dari program PYTHON yang dibuat adalah sebagai berikut. Pertama, direktori kerja dan lokasi penyimpanan *file* luaran didefinisikan. Kemudian, ekstraksi informasi dilakukan secara berulang, dari laman tertanggal 1 Januari 2016 hingga 1100 hari setelahnya. Pada setiap tanggal, REQUESTS mengakses informasi mentah *source code* dari laman terkait dan menyimpannya sebagai teks (tipe data *string*). BEAUTIFULSOUP melakukan *parsing* sehingga elemen dengan pola atau `id` tertentu dapat ditemukan dengan lebih mudah. Selanjutnya, parameter lubang korona dari `solarmonitor.org`, yakni koordinat pusat (`cen`), luas (`are`), medan magnet (`bfi`), dan fluks medan (`bfu`) dapat diekstrak dan disimpan dalam *file* luaran. Dengan prosedur ini, pekerjaan selesai dalam beberapa menit, bergantung pada kecepatan dan stabilitas jaringan internet saat eksekusi.

Setelah proses *scraping*, akan diperoleh daftar lubang korona yang terdeteksi selama 1100 hari sejak tahun 2016, beserta parameternya. Data ini akan dibandingkan dengan hasil identifikasi dan karakterisasi lubang korona yang dilakukan secara otomatis menggunakan algoritma DeLuNa (Deteksi Lubang Korona).

Sekilas, tampak DeLuNa cenderung menghasilkan area lubang korona yang lebih besar dibandingkan Chimera. Ada perbedaan pada kedua algoritma meski keduanya menggunakan metode dasar yang sama, yakni *intensity thresholding*. Nilai ambang (*threshold*) dan koreksi efek proyeksi merupakan faktor yang membedakan keduanya.

Terlepas dari perbandingan tersebut, pekerjaan ini mendemonstrasikan bagaimana *web scraping* membantu mengekstrak data dan informasi semi-terstruktur secara masif dalam tempo yang singkat.



Gambar 1: Plot luas/area lubang korona yang teramati tahun 2016-2018 dan dideteksi oleh algoritma Chimera dari `solarmonitor.org` (a) dan DeLuNa (b). Pada panel (b), plot warna biru menyatakan total area lubang korona sedangkan warna hijau menyatakan area lubang korona yang dianggap geoefektif. Perbandingan area yang dihasilkan Chimera dan DeLuNa disajikan pada panel (c).