

باسمه تعالی
درس: بازیابی اطلاعات



نام و نام خانوادگی: تینا توکلی - هادی امینی

شماره دانشجویی: 9912762370 - 9922762220

شماره تمرین گروهی: 04

توضیح و تحلیل بخش ها:

بخش اول:

پس از crawl کردن داده ها، خروجی در 2 فایل (newest و relevance) ذخیره شده است که برای استفاده از آنها در بخش بعدی، مقادیر عددی باید به صورت int باشد که این مورد را در فاز قبلی رعایت کرده ایم و همچنین تاریخ ها باید به صورت timestamp باشد که این مورد در بخش بعدی در فایل insert.py پوشش داده میشود(قبل از اضافه کردن داده ها).

خروجی:

```
ta > newestjson > [ ] papers > {} 5
1 {
2   "papers": [
3     {
4       "title": "Enhanced Perimeter Intrusion Detection System (PIDS) - Resilient to Environmental Variations for",
5       "Page(s)": null,
6       "Cites in Papers": null,
7       "Cites in Patent": null,
8       "Full Text Views": null,
9       "Publisher": "IEEE",
10      "DOI": "10.1109/SysCon61195.2024.10553451",
11      "Date of Publication": "15-18 April 2024",
12      "Abstract": "Perimeter intrusion detection systems (PIDS) often face challenges in accurately detecting bre",
13      "Published in": {
14        "name": "2024 IEEE International Systems Conference (SysCon)",
15        "link": "https://ieeexplore.ieee.org/xpl/conhome/10553374/proceeding"
16      },
17      "Authors": [
18        {
19          "name": "Ravindranath KV",
20          "from": "ComputerScience Department, Data Science, Texas A&M University, College Station, USA"
21        },
22        {
23          "name": "Irfan Ahmad Khan",
24          "from": "Marine Engineering Technology Department in a Joint Affiliate Appointment With Electrical",
25        }
26      ],
27      "IEEE Keywords": [
28        "Optical fibers",
29        "Optical fiber sensors",
30        "Transducers",
31        "Intrusion detection",
32        "Real-time systems",
33        "Sensors"
34      ]
35    }
36  ]
37 }
```

شکل 1 بخشی از خروجی crawl

بخش دوم:

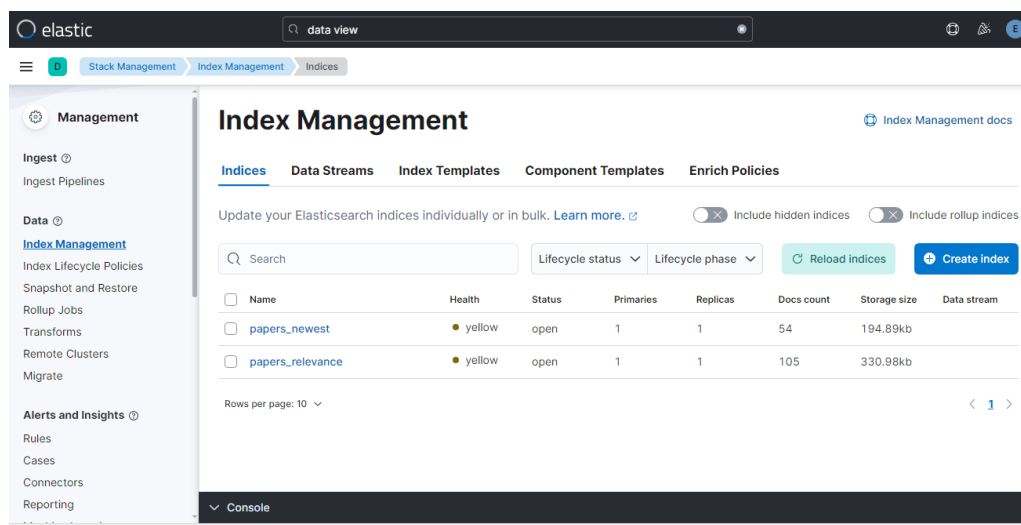
در این بخش از فایل `insert.py` استفاده می‌کنیم تا دو ایندکس برای دیتاهای `newest` و `relevance` ایجاد و داده‌ها را به `elastic` اضافه کنیم.

سپس چند `visualization` می‌سازیم تا با استفاده از آن‌ها اطلاعات آماری داده‌ها را در داشبورد نمایش دهیم:

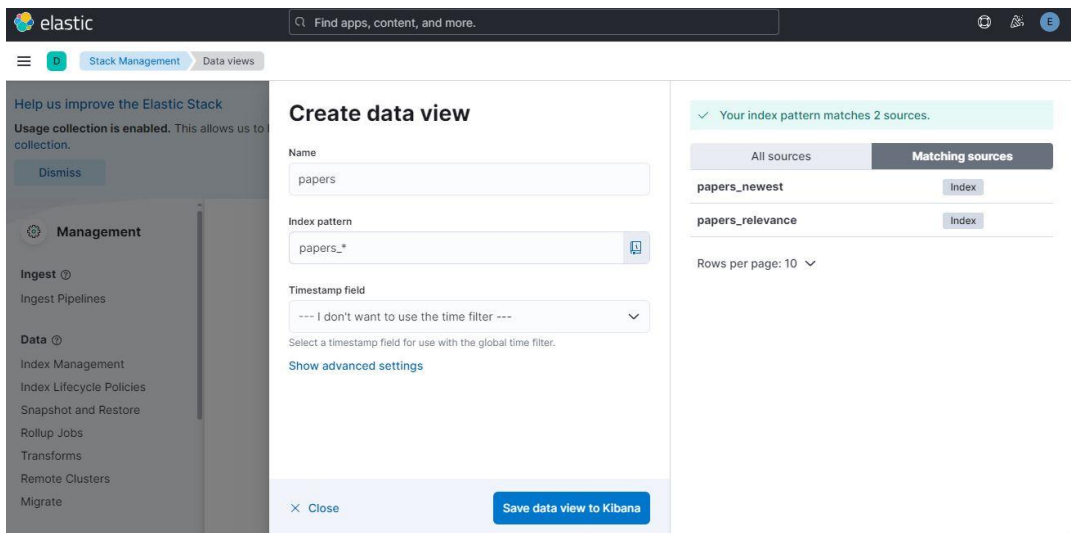
- ❖ نمودار `tag cloud` برای مشاهده کلمات کلیدی پرتکرار
- ❖ نمودار دایره‌ای برای مقایسه تعداد مقالات در کنفرانس‌های مختلف
- ❖ نمودار میله‌ای برای بررسی سال‌های انتشار مقالات
- ❖ نمودار میله‌ای افقی برای نمایش میزان `citation`‌ها
- ❖ تعداد کل رکورد‌ها

به وسیله این داشبورد می‌توانیم دید جامعی نسبت به نتایج جستجو به دست آوریم.

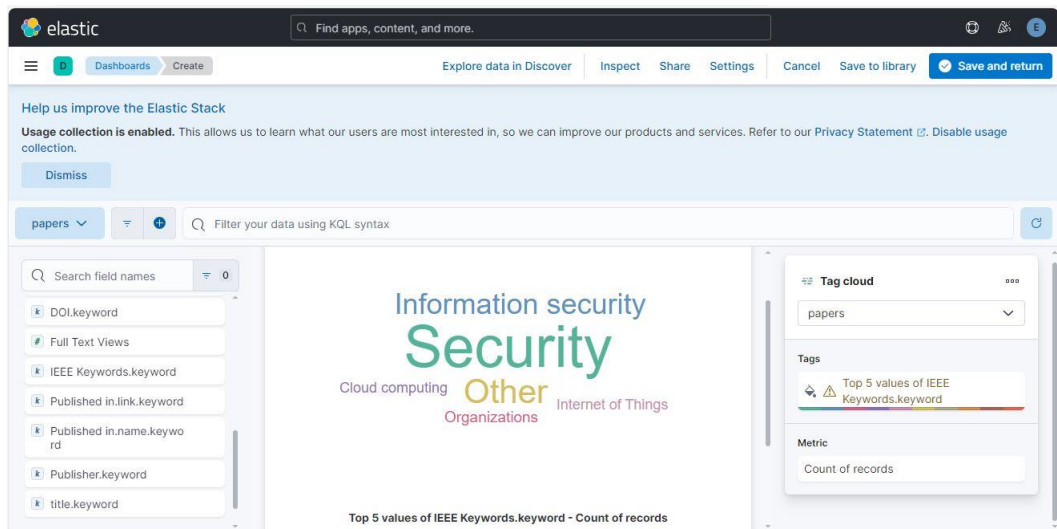
• خروجی :



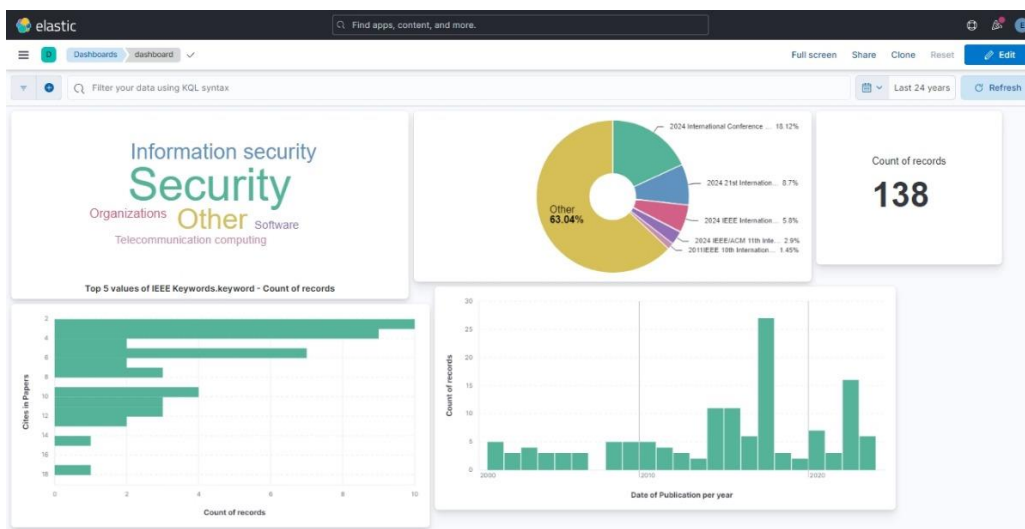
شکل 2 اضافه شدن ایندکس‌ها



شکل 3 ساخت data view از indexes



شکل 4- ایجاد یک داشبورد در ارتباط با keywords



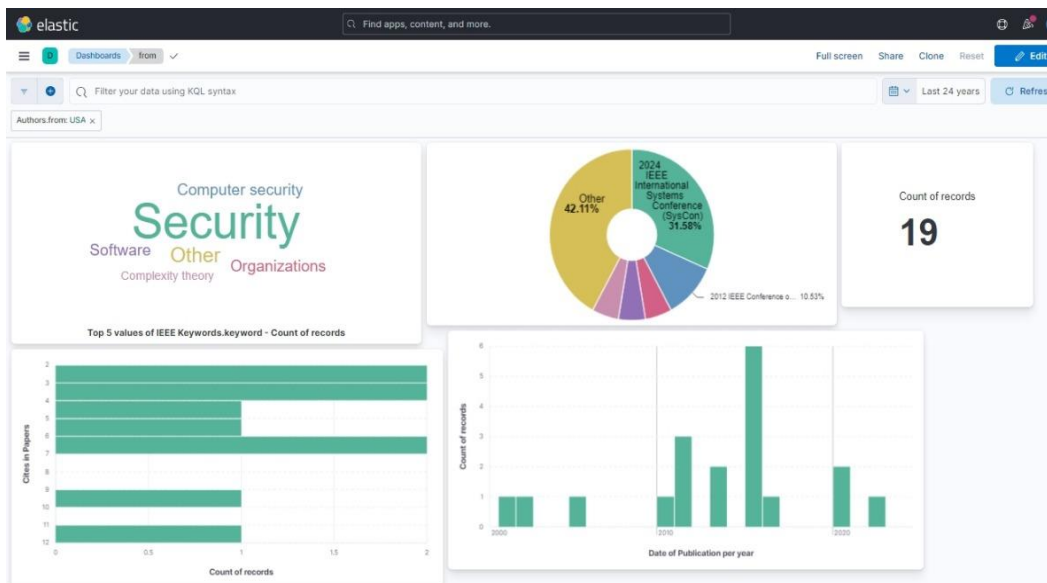
شکل 5- نمایش داشبورد نهایی

در 138 تا مقاله ی موجود بیشترین keywords مرتبط با امنیت وامنیت اطلاعات بوده وهمچنین اکثر مقالات در بازه 2015 تا 2018 منتشر شده و همچنین citation های کمی دارند و بیشتر آنها نیز در کفرانس یکسانی منتشر نشده اند و پراکندگی زیادی وجود دارد.

○ بخش اول ساخت پرس و جوها:

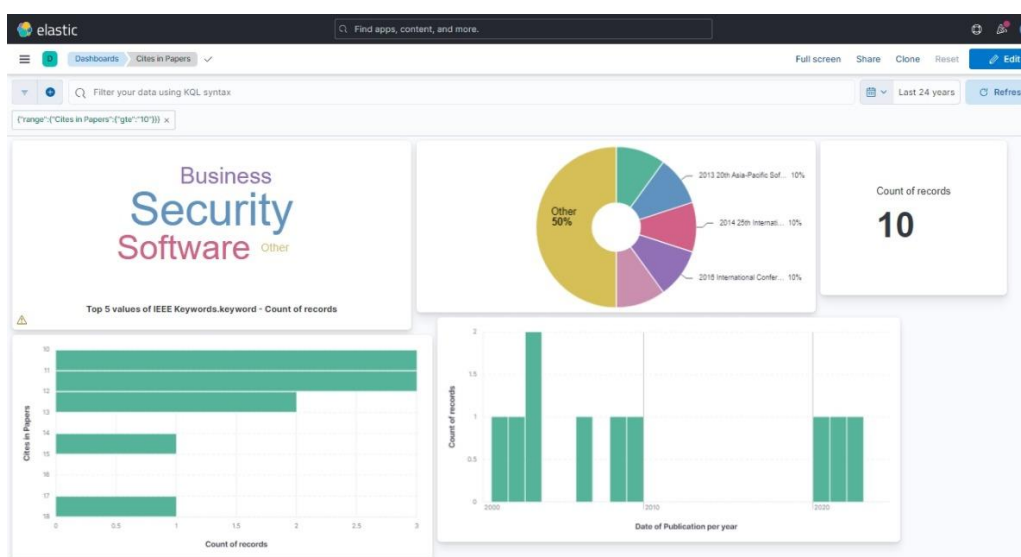
```

insert.py  outputjson X  command.txt  iK_HW1.ipynb  dashboard.html  search.html  query.py  main.py
output > outputjson > [ ] query0
1
2 {
3   "query0": [ ...
432 ],
433   "query1": [ ...
941 ],
942   "query2": [ ...
1398 ],
1399   "query3": [
1391 {
1392   "title": "A Novel Physical Layer Security Technique Using Master-Slave Full Duplex Communication",
1393   "Page(s)": null,
1394   "Cites in Papers": 6,
1395   "Cites in Patent": null,
1396   "Full Text Views": 688,
1397   "Publisher": "IEEE",
1398   "DOI": "10.1109/MMSYM.2019.8700776",
1399   "Date of Publication": "2002-06-07T20:19:00",
1400   "Abstract": "In this work we present a novel technique for physical layer security in the Internet-of-Thing",
1401   "Published in": {
1402     "name": "2019 IEEE MIT-S International Microwave Symposium (IMS)",
1403     "link": "https://ieeexplore.ieee.org/xpl/conhome/8697340/proceeding"
1404   },
1405   "Authors": [
1406     {
1407       "name": "Najme Ebrahimi",
1408       "from": "University of Michigan, Ann Arbor, USA"
1409     },
1410     {
1411       "name": "Behzad Yektakhah",
1412       "from": "University of Michigan, Ann Arbor, USA"
1413     },
1414     {
1415       "name": "Kamal Sanahandji"
1416     }
1417   ]
1418 }
1419 ]
1420 ]
1421 ]
1422 ]
1423 ]
1424 ]
1425 ]
1426 ]
1427 ]
1428 ]
1429 ]
1430 ]
1431 ]
1432 ]
1433 ]
1434 ]
1435 ]
1436 ]
1437 ]
1438 ]
1439 ]
1440 ]
1441 ]
1442 ]
1443 ]
1444 ]
1445 ]
1446 ]
1447 ]
1448 ]
1449 ]
1450 ]
1451 ]
1452 ]
1453 ]
1454 ]
1455 ]
1456 ]
1457 ]
1458 ]
1459 ]
1460 ]
1461 ]
1462 ]
1463 ]
1464 ]
1465 ]
1466 ]
1467 ]
1468 ]
1469 ]
1470 ]
1471 ]
1472 ]
1473 ]
1474 ]
1475 ]
1476 ]
1477 ]
1478 ]
1479 ]
1480 ]
1481 ]
1482 ]
1483 ]
1484 ]
1485 ]
1486 ]
1487 ]
1488 ]
1489 ]
1490 ]
1491 ]
1492 ]
1493 ]
1494 ]
1495 ]
1496 ]
1497 ]
1498 ]
1499 ]
1500 ]
1501 ]
1502 ]
1503 ]
1504 ]
1505 ]
1506 ]
1507 ]
1508 ]
1509 ]
1510 ]
1511 ]
1512 ]
1513 ]
1514 ]
1515 ]
1516 ]
1517 ]
1518 ]
1519 ]
1520 ]
1521 ]
1522 ]
1523 ]
1524 ]
1525 ]
1526 ]
1527 ]
1528 ]
1529 ]
1530 ]
1531 ]
1532 ]
1533 ]
1534 ]
1535 ]
1536 ]
1537 ]
1538 ]
1539 ]
1540 ]
1541 ]
1542 ]
1543 ]
1544 ]
1545 ]
1546 ]
1547 ]
1548 ]
1549 ]
1550 ]
1551 ]
1552 ]
1553 ]
1554 ]
1555 ]
1556 ]
1557 ]
1558 ]
1559 ]
1560 ]
1561 ]
1562 ]
1563 ]
1564 ]
1565 ]
1566 ]
1567 ]
1568 ]
1569 ]
1570 ]
1571 ]
1572 ]
1573 ]
1574 ]
1575 ]
1576 ]
1577 ]
1578 ]
1579 ]
1580 ]
1581 ]
1582 ]
1583 ]
1584 ]
1585 ]
1586 ]
1587 ]
1588 ]
1589 ]
1590 ]
1591 ]
1592 ]
1593 ]
1594 ]
1595 ]
1596 ]
1597 ]
1598 ]
1599 ]
1600 ]
1601 ]
1602 ]
1603 ]
1604 ]
1605 ]
1606 ]
1607 ]
1608 ]
1609 ]
1610 ]
1611 ]
1612 ]
1613 ]
1614 ]
1615 ]
1616 ]
1617 ]
1618 ]
1619 ]
1620 ]
1621 ]
1622 ]
1623 ]
1624 ]
1625 ]
1626 ]
1627 ]
1628 ]
1629 ]
1630 ]
1631 ]
1632 ]
1633 ]
1634 ]
1635 ]
1636 ]
1637 ]
1638 ]
1639 ]
1640 ]
1641 ]
1642 ]
1643 ]
1644 ]
1645 ]
1646 ]
1647 ]
1648 ]
1649 ]
1650 ]
1651 ]
1652 ]
1653 ]
1654 ]
1655 ]
1656 ]
1657 ]
1658 ]
1659 ]
1660 ]
1661 ]
1662 ]
1663 ]
1664 ]
1665 ]
1666 ]
1667 ]
1668 ]
1669 ]
1670 ]
1671 ]
1672 ]
1673 ]
1674 ]
1675 ]
1676 ]
1677 ]
1678 ]
1679 ]
1680 ]
1681 ]
1682 ]
1683 ]
1684 ]
1685 ]
1686 ]
1687 ]
1688 ]
1689 ]
1690 ]
1691 ]
1692 ]
1693 ]
1694 ]
1695 ]
1696 ]
1697 ]
1698 ]
1699 ]
1700 ]
1701 ]
1702 ]
1703 ]
1704 ]
1705 ]
1706 ]
1707 ]
1708 ]
1709 ]
1710 ]
1711 ]
1712 ]
1713 ]
1714 ]
1715 ]
1716 ]
1717 ]
1718 ]
1719 ]
1720 ]
1721 ]
1722 ]
1723 ]
1724 ]
1725 ]
1726 ]
1727 ]
1728 ]
1729 ]
1730 ]
1731 ]
1732 ]
1733 ]
1734 ]
1735 ]
1736 ]
1737 ]
1738 ]
1739 ]
1740 ]
1741 ]
1742 ]
1743 ]
1744 ]
1745 ]
1746 ]
1747 ]
1748 ]
1749 ]
1750 ]
1751 ]
1752 ]
1753 ]
1754 ]
1755 ]
1756 ]
1757 ]
1758 ]
1759 ]
1760 ]
1761 ]
1762 ]
1763 ]
1764 ]
1765 ]
1766 ]
1767 ]
1768 ]
1769 ]
1770 ]
1771 ]
1772 ]
1773 ]
1774 ]
1775 ]
1776 ]
1777 ]
1778 ]
1779 ]
1780 ]
1781 ]
1782 ]
1783 ]
1784 ]
1785 ]
1786 ]
1787 ]
1788 ]
1789 ]
1790 ]
1791 ]
1792 ]
1793 ]
1794 ]
1795 ]
1796 ]
1797 ]
1798 ]
1799 ]
1800 ]
1801 ]
1802 ]
1803 ]
1804 ]
1805 ]
1806 ]
1807 ]
1808 ]
1809 ]
1810 ]
1811 ]
1812 ]
1813 ]
1814 ]
1815 ]
1816 ]
1817 ]
1818 ]
1819 ]
1820 ]
1821 ]
1822 ]
1823 ]
1824 ]
1825 ]
1826 ]
1827 ]
1828 ]
1829 ]
1830 ]
1831 ]
1832 ]
1833 ]
1834 ]
1835 ]
1836 ]
1837 ]
1838 ]
1839 ]
1840 ]
1841 ]
1842 ]
1843 ]
1844 ]
1845 ]
1846 ]
1847 ]
1848 ]
1849 ]
1850 ]
1851 ]
1852 ]
1853 ]
1854 ]
1855 ]
1856 ]
1857 ]
1858 ]
1859 ]
1860 ]
1861 ]
1862 ]
1863 ]
1864 ]
1865 ]
1866 ]
1867 ]
1868 ]
1869 ]
1870 ]
1871 ]
1872 ]
1873 ]
1874 ]
1875 ]
1876 ]
1877 ]
1878 ]
1879 ]
1880 ]
1881 ]
1882 ]
1883 ]
1884 ]
1885 ]
1886 ]
1887 ]
1888 ]
1889 ]
1890 ]
1891 ]
1892 ]
1893 ]
1894 ]
1895 ]
1896 ]
1897 ]
1898 ]
1899 ]
1900 ]
1901 ]
1902 ]
1903 ]
1904 ]
1905 ]
1906 ]
1907 ]
1908 ]
1909 ]
1910 ]
1911 ]
1912 ]
1913 ]
1914 ]
1915 ]
1916 ]
1917 ]
1918 ]
1919 ]
1920 ]
1921 ]
1922 ]
1923 ]
1924 ]
1925 ]
1926 ]
1927 ]
1928 ]
1929 ]
1930 ]
1931 ]
1932 ]
1933 ]
1934 ]
1935 ]
1936 ]
1937 ]
1938 ]
1939 ]
1940 ]
1941 ]
1942 ]
1943 ]
1944 ]
1945 ]
1946 ]
1947 ]
1948 ]
1949 ]
1950 ]
1951 ]
1952 ]
1953 ]
1954 ]
1955 ]
1956 ]
1957 ]
1958 ]
1959 ]
1960 ]
1961 ]
1962 ]
1963 ]
1964 ]
1965 ]
1966 ]
1967 ]
1968 ]
1969 ]
1970 ]
1971 ]
1972 ]
1973 ]
1974 ]
1975 ]
1976 ]
1977 ]
1978 ]
1979 ]
1980 ]
1981 ]
1982 ]
1983 ]
1984 ]
1985 ]
1986 ]
1987 ]
1988 ]
1989 ]
1990 ]
1991 ]
1992 ]
1993 ]
1994 ]
1995 ]
1996 ]
1997 ]
1998 ]
1999 ]
2000 ]
2001 ]
2002 ]
2003 ]
2004 ]
2005 ]
2006 ]
2007 ]
2008 ]
2009 ]
2010 ]
2011 ]
2012 ]
2013 ]
2014 ]
2015 ]
2016 ]
2017 ]
2018 ]
2019 ]
2020 ]
2021 ]
2022 ]
2023 ]
2024 ]
2025 ]
2026 ]
2027 ]
2028 ]
2029 ]
2030 ]
2031 ]
2032 ]
2033 ]
2034 ]
2035 ]
2036 ]
2037 ]
2038 ]
2039 ]
2040 ]
2041 ]
2042 ]
2043 ]
2044 ]
2045 ]
2046 ]
2047 ]
2048 ]
2049 ]
2050 ]
2051 ]
2052 ]
2053 ]
2054 ]
2055 ]
2056 ]
2057 ]
2058 ]
2059 ]
2060 ]
2061 ]
2062 ]
2063 ]
2064 ]
2065 ]
2066 ]
2067 ]
2068 ]
2069 ]
2070 ]
2071 ]
2072 ]
2073 ]
2074 ]
2075 ]
2076 ]
2077 ]
2078 ]
2079 ]
2080 ]
2081 ]
2082 ]
2083 ]
2084 ]
2085 ]
2086 ]
2087 ]
2088 ]
2089 ]
2090 ]
2091 ]
2092 ]
2093 ]
2094 ]
2095 ]
2096 ]
2097 ]
2098 ]
2099 ]
2100 ]
2101 ]
2102 ]
2103 ]
2104 ]
2105 ]
2106 ]
2107 ]
2108 ]
2109 ]
2110 ]
2111 ]
2112 ]
2113 ]
2114 ]
2115 ]
2116 ]
2117 ]
2118 ]
2119 ]
2120 ]
2121 ]
2122 ]
2123 ]
2124 ]
2125 ]
2126 ]
2127 ]
2128 ]
2129 ]
2130 ]
2131 ]
2132 ]
2133 ]
2134 ]
2135 ]
2136 ]
2137 ]
2138 ]
2139 ]
2140 ]
2141 ]
2142 ]
2143 ]
2144 ]
2145 ]
2146 ]
2147 ]
2148 ]
2149 ]
2150 ]
2151 ]
2152 ]
2153 ]
2154 ]
2155 ]
2156 ]
2157 ]
2158 ]
2159 ]
2160 ]
2161 ]
2162 ]
2163 ]
2164 ]
2165 ]
2166 ]
2167 ]
2168 ]
2169 ]
2170 ]
2171 ]
2172 ]
2173 ]
2174 ]
2175 ]
2176 ]
2177 ]
2178 ]
2179 ]
2180 ]
2181 ]
2182 ]
2183 ]
2184 ]
2185 ]
2186 ]
2187 ]
2188 ]
2189 ]
2190 ]
2191 ]
2192 ]
2193 ]
2194 ]
2195 ]
2196 ]
2197 ]
2198 ]
2199 ]
2200 ]
2201 ]
2202 ]
2203 ]
2204 ]
2205 ]
2206 ]
2207 ]
2208 ]
2209 ]
2210 ]
2211 ]
2212 ]
2213 ]
2214 ]
2215 ]
2216 ]
2217 ]
2218 ]
2219 ]
2220 ]
2221 ]
2222 ]
2223 ]
2224 ]
2225 ]
2226 ]
2227 ]
2228 ]
2229 ]
2230 ]
2231 ]
2232 ]
2233 ]
2234 ]
2235 ]
2236 ]
2237 ]
2238 ]
2239 ]
2240 ]
2241 ]
2242 ]
2243 ]
2244 ]
2245 ]
2246 ]
2247 ]
2248 ]
2249 ]
2250 ]
2251 ]
2252 ]
2253 ]
2254 ]
2255 ]
2256 ]
2257 ]
2258 ]
2259 ]
2260 ]
2261 ]
2262 ]
2263 ]
2264 ]
2265 ]
2266 ]
2267 ]
2268 ]
2269 ]
2270 ]
2271 ]
2272 ]
2273 ]
2274 ]
2275 ]
2276 ]
2277 ]
2278 ]
2279 ]
2280 ]
2281 ]
2282 ]
2283 ]
2284 ]
2285 ]
2286 ]
2287 ]
2288 ]
2289 ]
2290 ]
2291 ]
2292 ]
2293 ]
2294 ]
2295 ]
2296 ]
2297 ]
2298 ]
2299 ]
2300 ]
2301 ]
2302 ]
2303 ]
2304 ]
2305 ]
2306 ]
2307 ]
2308 ]
2309 ]
2310 ]
2311 ]
2312 ]
2313 ]
2314 ]
2315 ]
2316 ]
2317 ]
2318 ]
2319 ]
2320 ]
2321 ]
2322 ]
2323 ]
2324 ]
2325 ]
2326 ]
2327 ]
2328 ]
2329 ]
2330 ]
2331 ]
2332 ]
2333 ]
2334 ]
2335 ]
2336 ]
2337 ]
2338 ]
2339 ]
2340 ]
2341 ]
2342 ]
2343 ]
2344 ]
2345 ]
2346 ]
2347 ]
2348 ]
2349 ]
2350 ]
2351 ]
2352 ]
2353 ]
2354 ]
2355 ]
2356 ]
2357 ]
2358 ]
2359 ]
2360 ]
2361 ]
2362 ]
2363 ]
2364 ]
2365 ]
2366 ]
2367 ]
2368 ]
2369 ]
2370 ]
2371 ]
2372 ]
2373 ]
2374 ]
2375 ]
2376 ]
2377 ]
2378 ]
2379 ]
2380 ]
2381 ]
2382 ]
2383 ]
2384 ]
2385 ]
2386 ]
2387 ]
2388 ]
2389 ]
2390 ]
2391 ]
2392 ]
2393 ]
2394 ]
2395 ]
2396 ]
2397 ]
2398 ]
2399 ]
2400 ]
2401 ]
2402 ]
2403 ]
2404 ]
2405 ]
2406 ]
2407 ]
2408 ]
2409 ]
2410 ]
2411 ]
2412 ]
2413 ]
2414 ]
2415 ]
2416 ]
2417 ]
2418 ]
2419 ]
2420 ]
2421 ]
2422 ]
2423 ]
2424 ]
2425 ]
2426 ]
2427 ]
2428 ]
2429 ]
2430 ]
2431 ]
2432 ]
2433 ]
2434 ]
2435 ]
2436 ]
2437 ]
2438 ]
2439 ]
2440 ]
2441 ]
2442 ]
2443 ]
2444 ]
2445 ]
2446 ]
2447 ]
2448 ]
2449 ]
2450 ]
2451 ]
2452 ]
2453 ]
2454 ]
2455 ]
2456 ]
2457 ]
2458 ]
2459 ]
2460 ]
2461 ]
2462 ]
2463 ]
2464 ]
2465 ]
2466 ]
2467 ]
2468 ]
2469 ]
2470 ]
2471 ]
2472 ]
2473 ]
2474 ]
2475 ]
2476 ]
2477 ]
2478 ]
2479 ]
2480 ]
2481 ]
2482 ]
2483 ]
2484 ]
2485 ]
2486 ]
2487 ]
2488 ]
2489 ]
2490 ]
2491 ]
2492 ]
2493 ]
2494 ]
2495 ]
2496 ]
2497 ]
2498 ]
2499 ]
2500 ]
2501 ]
2502 ]
2503 ]
2504 ]
2505 ]
2506 ]
2507 ]
2508 ]
2509 ]
2510 ]
2511 ]
2512 ]
2513 ]
2514 ]
2515 ]
2516 ]
2517 ]
2518 ]
2519 ]
2520 ]
2521 ]
2522 ]
2523 ]
2524 ]
2525 ]
2526 ]
2527 ]
2528 ]
2529 ]
2530 ]
2531 ]
2532 ]
2533 ]
2534 ]
2535 ]
2536 ]
2537 ]
2538 ]
2539 ]
2540 ]
2541 ]
2542 ]
2543 ]
2544 ]
2545 ]
2546 ]
2547 ]
2548 ]
2549 ]
2550 ]
2551 ]
2552 ]
2553 ]
2554 ]
2555 ]
2556 ]
2557 ]
2558 ]
2559 ]
2560 ]
2561 ]
2562 ]
2563 ]
2564 ]
2565 ]
2566 ]
2567 ]
2568 ]
2569 ]
2570 ]
2571 ]
2572 ]
2573 ]
2574 ]
2575 ]
2576 ]
2577 ]
2578 ]
2579 ]
2580 ]
2581 ]
2582 ]
2583 ]
2584 ]
2585 ]
2586 ]
2587 ]
2588 ]
2589 ]
2590 ]
2591 ]
2592 ]
2593 ]
2594 ]
2595 ]
2596 ]
2597 ]
2598 ]
2599 ]
2600 ]
2601 ]
2602 ]
2603 ]
2604 ]
2605 ]
2606 ]
2607 ]
2608 ]
2609 ]
2610 ]
2611 ]
2612 ]
2613 ]
2614 ]
2615 ]
2616 ]
2617 ]
2618 ]
2619 ]
2620 ]
2621 ]
2622 ]
2623 ]
2624 ]
2625 ]
2626 ]
2627 ]
2628 ]
2629 ]
2630 ]
2631 ]
2632 ]
2633 ]
2634 ]
2635 ]
2636 ]
2637 ]
2638 ]
2639 ]
2640 ]
2641 ]
2642 ]
2643 ]
2644 ]
2645 ]
2646 ]
2647 ]
2648 ]
2649 ]
2650 ]
2651 ]
2652 ]
2653 ]
2654 ]
2655 ]
2656 ]
2657 ]
2658 ]
2659 ]
2660 ]
2661 ]
2662 ]
2663 ]
2664 ]
2665 ]
2666 ]
2667 ]
2668 ]
2669 ]
2670 ]
2671 ]
2672 ]
2673 ]
2674 ]
2675 ]
2676 ]
2677 ]
2678 ]
2679 ]
2680 ]
2681 ]
2682 ]
2683 ]
2684 ]
2685 ]
2686 ]
2687 ]
2688 ]
2689 ]
2690 ]
2691 ]
2692 ]
2693 ]
2694 ]
2695 ]
2696 ]
2697 ]
2698 ]
2699 ]
2700 ]
2701 ]
2702 ]
2703 ]
2704 ]
2705 ]
2706 ]
2707 ]
2708 ]
2709 ]
2710 ]
2711 ]
2712 ]
2713 ]
2714 ]
2715 ]
2716 ]
2717 ]
2718 ]
2719 ]
2720 ]
2721 ]
2722 ]
2723 ]
2724 ]
2725 ]
2726 ]
2727 ]
2728 ]
2729 ]
2730 ]
2731 ]
2732 ]
2733 ]
2734 ]
2735 ]
2736 ]
2737 ]
2738 ]
2739 ]
2740 ]
2741 ]
2742 ]
2743 ]
2744 ]
2745 ]
2746 ]
2747 ]
2748 ]
2749 ]
2750 ]
2751 ]
2752 ]
2753 ]
2754 ]
2755 ]
2756 ]
2757 ]
2758 ]
2759 ]
2760 ]
2761 ]
2762 ]
2763 ]
2764 ]
2765 ]
2766 ]
2767 ]
2768 ]
2769 ]
2770 ]
2771 ]
2772 ]
2773 ]
2774 ]
2775 ]
2776 ]
2777 ]
2778 ]
2779 ]
2780 ]
2781 ]
2782 ]
2783 ]
2784 ]
2785 ]
2786 ]
2787 ]
2788 ]
2789 ]
2790 ]
2791 ]
2792 ]
2793 ]
2794 ]
2795 ]
2796 ]
2797 ]
2798 ]
2799 ]
2800 ]
2801 ]
2802 ]
2803 ]
2804 ]
2805 ]
2806 ]
2807 ]
2808 ]
2809 ]
2810 ]
2811 ]
2812 ]
2813 ]
2814 ]
2815 ]
2816 ]
2817 ]
2818 ]
2819 ]
2820 ]
2821 ]
2822 ]
2823 ]
2824 ]
2825 ]
2826 ]
2827 ]
2828 ]
2829 ]
2830 ]
2831 ]
2832 ]
2833 ]
2834 ]
2835 ]
2836 ]
2837 ]
2838 ]
2839 ]
2840 ]
2841 ]
2842 ]
2843 ]
2844 ]
2845 ]
2846 ]
2847 ]
2848 ]
2849 ]
2850 ]
2851 ]
2852 ]
2853 ]
2854 ]
2855 ]
2856 ]
2857 ]
2858 ]
2859 ]
2860 ]
2861 ]
2862 ]
2863 ]
2864 ]
2865 ]
2866 ]
2867 ]
2868 ]
2869 ]
2870 ]
2871 ]
2872 ]
2873 ]
2874 ]
2875 ]
2876 ]
2877 ]
2878 ]
2879 ]
2880 ]
2881 ]
2882 ]
2883 ]
2884 ]
2885 ]
2886 ]
2887 ]
2888 ]
2889 ]
2890 ]
2891 ]
2892 ]
2893 ]
2894 ]
2895 ]
2896 ]
2897 ]
2898 ]
2899 ]
2900 ]
2901 ]
2902 ]
2903 ]
2904 ]
2905 ]
2906 ]
2907 ]
2908 ]
2909 ]
2910 ]
2911 ]
2912 ]
2913 ]
2914 ]
2915 ]
2916 ]
2917 ]
2918 ]
2919 ]
2920 ]
2921 ]
2922 ]
2923 ]
2924 ]
2925 ]
2926 ]
2927 ]
2928 ]
2929 ]
2930 ]
2931 ]
2932 ]
2933 ]
2934 ]
2935 ]
2936 ]
2937 ]
2938 ]
2939 ]
2940 ]
2941 ]
2942 ]
2943 ]
2944 ]
2945 ]
2946 ]
2947 ]
2948 ]
2949 ]
2950 ]
2951 ]
2952 ]
2953 ]
2954 ]
2955 ]
2956 ]
2957 ]
2958 ]
2959 ]
2960 ]
2961 ]
2962 ]
2963 ]
2964 ]
2965 ]
2966 ]
2967 ]
2968 ]
2969 ]
2970 ]
2971 ]
2972 ]
2973 ]
2974 ]
2975 ]
2976 ]
2977 ]
2978 ]
2979 ]
2980 ]
2981 ]
2982 ]
2983 ]
2984 ]
2985 ]
2986 ]
2987 ]
2988 ]
2989 ]
2990 ]
2991 ]
2992 ]
2993 ]
2994 ]
2995 ]
2996 ]
2997 ]
2998 ]
2999 ]
3000 ]
3001 ]
3002 ]
3003 ]
3004 ]
3005 ]
3006 ]
3007 ]
3008 ]
3009 ]
3010 ]
3011 ]
3012 ]
3013 ]
3014 ]
3015 ]
3016 ]
3017 ]
3018 ]
3019 ]
3020 ]
3021 ]
3022 ]
3023 ]
3024 ]
3025 ]
3026 ]
3027 ]
3028 ]
3029 ]
3030 ]
3031 ]
3032 ]
3033 ]
3034 ]
3035 ]
3036 ]
3037 ]
3038 ]
3039 ]
3040 ]
3041 ]
3042 ]
3043 ]
3044 ]
3045 ]
3046 ]
3047 ]
3048 ]
3049 ]
3050 ]
3051 ]
3052 ]
3053 ]
3054 ]
3055 ]
3056 ]
3057 ]
3058 ]
3059 ]
3060 ]
3061 ]
3062 ]
3063 ]
3064 ]
3065 ]
3066 ]
3067 ]
3068 ]
3069 ]
3070 ]
3071 ]
3072 ]
3073 ]
3074 ]
3075 ]
3076 ]
3077 ]
3078 ]
3079 ]
3080 ]
3081 ]
3082 ]
3083 ]
3084 ]
3085 ]
3086 ]
3087 ]
3088 ]
3089 ]
3090 ]
3091 ]
3092 ]
3093 ]
3094 ]
3095 ]
3096 ]
3097 ]
3098 ]
3099 ]
3100 ]
3101 ]
3102 ]
3103 ]
3104 ]
3105 ]
3106 ]
3107 ]
3108 ]
3109 ]
3110 ]
3111 ]
3112 ]
3113 ]
3114 ]
3115 ]
3116 ]
3117 ]
3118 ]
3119 ]
3120 ]
3121 ]
3122 ]
3123 ]
3124 ]
3125 ]
3126 ]
3127 ]
3128 ]
3129 ]
3130 ]
3131 ]
3132 ]
3133 ]
3134 ]
3135 ]
3136 ]
3137 ]
3138 ]
3139 ]
3140 ]
3141 ]
3142 ]
3143 ]
3144 ]
3145 ]
3146 ]
3147 ]
3148 ]
3149 ]
3150 ]
3151 ]
3152 ]
3153 ]
3154 ]
3155 ]
3156 ]
3157 ]
3158 ]
3159 ]
3160 ]
3161 ]
3162 ]
3163 ]
3164 ]
3165 ]
3166 ]
3167 ]
3168 ]
3169 ]
3170 ]
3171 ]
3172 ]
3173 ]
3174 ]
3175 ]
3176 ]
3177 ]
3178 ]
3179 ]
3180 ]
3181 ]
3182 ]
3183 ]
3184 ]
3185 ]
3186 ]
3187 ]
3188 ]
3189 ]
3190 ]
3191 ]
3192 ]
3193 ]
3194 ]
3195 ]
3196 ]
3197 ]
3198 ]
3199 ]
3200 ]
3201 ]
3202 ]
3203 ]
3204 ]
3205 ]
3206 ]
3207 ]
3208 ]
3209 ]
3210 ]
3211 ]
3212 ]
3213 ]
3214 ]
3215 ]
3216 ]
3217 ]
3218 ]
3219 ]
3220 ]
3221 ]
3222 ]
3223 ]
3224 ]
3225 ]
3226 ]
3227 ]
3228 ]
3229 ]
3230 ]
3231 ]
3232 ]
3233 ]
3234 ]
3235 ]
3236 ]
3237 ]
3238 ]
3239 ]
3240 ]
3241 ]
3242 ]
3243 ]
3244 ]
3245 ]
3246 ]
3247 ]
3248 ]
3249 ]
3250 ]
3251 ]
3252 ]
3253 ]
3254 ]
3255 ]
3256 ]
3257 ]
3258 ]
3259 ]
3260 ]
3261 ]
3262 ]
3263 ]
3264 ]
3265 ]
3266 ]
3267 ]
3268 ]
3269 ]
3270 ]
3271 ]
3272 ]
3273 ]
3274 ]
3275 ]
3276 ]
3277 ]
3278 ]
3279 ]
3280 ]
3281 ]
3282 ]
3283 ]
3284 ]
3285 ]
3286 ]
3287 ]
3288 ]
3289 ]
3290 ]
3291 ]
3292 ]
3293 ]
3294 ]
3295 ]
3296 ]
3297 ]
3298 ]
3299 ]
3300 ]
3301 ]
3302 ]
3303 ]
3304 ]
3305 ]
3306 ]
3307 ]
3308 ]
3309 ]
3310 ]
3311 ]
3312 ]
3313 ]
3314 ]
3315 ]
3316 ]
3317 ]
3318 ]
3319 ]
3320 ]
3321 ]
3322 ]
3323 ]
3324 ]
3325 ]
3326 ]
3327 ]
3328 ]
3329 ]
3330 ]
3331 ]
3332 ]
3333 ]
3334 ]
3335 ]
3336 ]
3337 ]
3338 ]
3339 ]
3340 ]
3341 ]
3342 ]
3343 ]
3344 ]
3345 ]
3346 ]
3347 ]
3348 ]
3349 ]
3350 ]
3351 ]
3352 ]
3353 ]
3354 ]
3355 ]
3356 ]
3357 ]
3358 ]
3359 ]
3360 ]
3361 ]
3362 ]
3
```



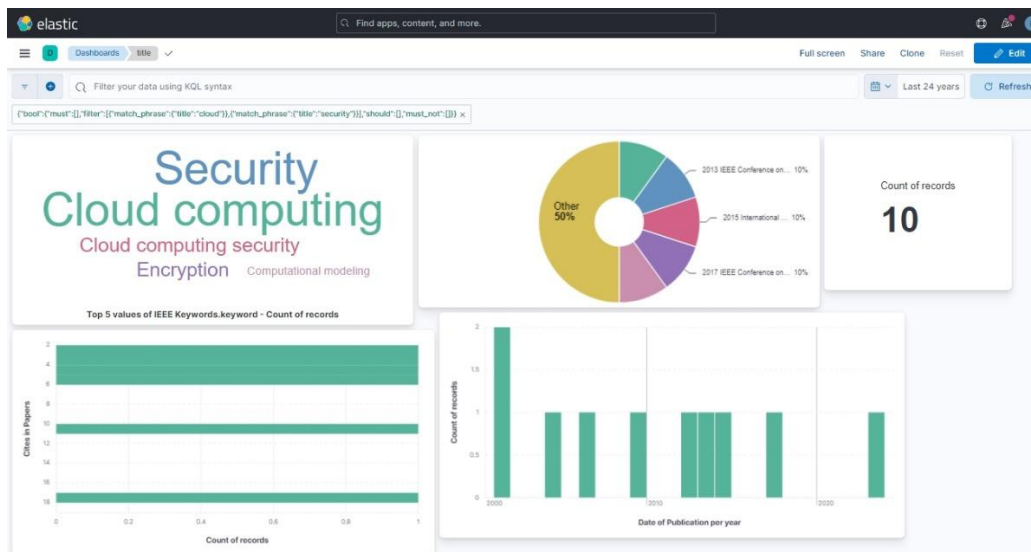
شکل 8 کوثری مربوط به `author.from`

این کوثری، مقاله های مربوط به نویسندگان آمریکایی را نشان میدهد. 30٪ این مقالات در کنفرانس IEEE International Systems 2024 منتشر شده اند. همچنین نمودار تاریخ انتشار نشان میدهد که در سال 2015 نسبت به سایر سال ها مقالات بیشتری منتشر شده است.



شکل 9 کوثری مربوط به `cites in paper`

این کوثری مقالات با حداقل 10 citation را بازیابی میکند. اکثر این مقالات در حوزه های امنیت و نرم افزار و بیزینس بوده اند.



شکل 10 کوئری مربوط به title

این کوئری مقالات شامل کلمه security و cloud در عنوان هستند. مطابق انتظار کلمات Cloud Computing و Security بیشتر در کلمات کلیدی حضور دارند و ترکیب این دو معادل Cloud computing security در رتبه سوم قرار دارد. همچنین رمزنگاری (encryption) نیز اهمیت داشته است.

بخش سوم:

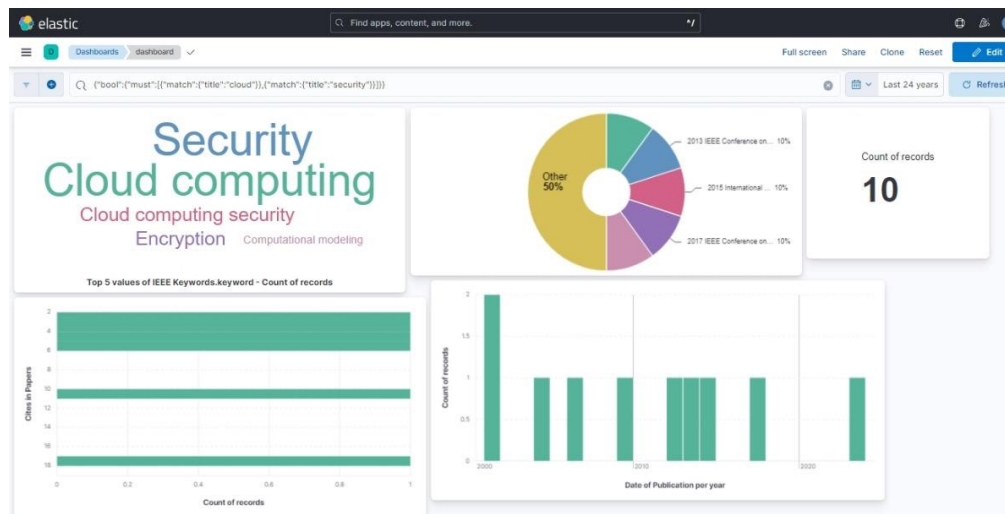
در این قسمت با استفاده از flask یک رابط وب برای جستجوی مقالات پیاده سازی میکنیم. با استفاده از ورودی کاربر، کوئری ساخته میشود و لینک داشبورد برای مشاهده نتایج جستجو نمایش داده میشود. (کاربر امکان ترکیب فیلدهای مختلف با استفاده از عملگرهای and, or دارد)

• خروجی:

The Advanced Search form shows the following configuration:

- Field:** Title
- Query:** cloud
- Operator:** AND
- Field:** Title
- Query:** security
- Buttons:** Add Condition, Search

شکل 11 – advanced search



شکل 12-نمایش داشبورد براساس کوئری وارد شده در صفحه advanced search

بخش چهارم:

در این بخش کاربر نام مقاله یا DOI آن را به همراه تعداد مقالات مشابه درخواستی اعلام میکند. مقاله مورد نظر از elastic دریافت میشود و شباهت کسینوسی آن با سایر مقالات محاسبه میشود. میزان شباهت از طریق محاسبه score به صورت زیر به دست می آید:

$$Score_i = 0.5 * Similarity_{title}(paper_{user}, paper_i) + Similarity_{author_keywords}(paper_{user}, paper_i) + Similarity_{IEEE_keywords}(paper_{user}, paper_i) + Similarity_{abstract}(paper_{user}, paper_i)$$

سپس امتیازها sort میشوند و بالاترین ها به تعداد درخواستی انتخاب میشوند.

• خروجی:

The screenshot shows the 'Advanced Search' interface with the following fields and options:

- Field:** DOI
- Title or DOI:** 10.1109/IC3IoT60841.2024.10550282
- Number of Similar Articles:** A dropdown menu with options: 5, 10, 15, 20, 30. The value 5 is currently selected.

شکل 13- براساس DOI

Advanced Search

[Advanced Search](#)
[RS](#)

Field

title
title
DOI
10

Title or DOI
Data Provenance P-Chain in Blockchain with Enhanced Security by Combin

Find Similar

شکل 14 براساس title

Recommended Papers

Rank	Title	DOI
1	Prov-IoT: A Security-Aware IoT Provenance Model	10.1109/TrustCom50675.2020.00183
2	The Research Study on Identification of Threats and Security Techniques in Cloud Environment	10.1109/IDICAIEI58380.2023.10407003
3	Review of Data Governance Approaches in the Field of Transportation Domain	10.1109/SCSP61506.2024.10552682
4	Cloud Security: Challenges and Strategies for Ensuring Data Protection	10.1109/ICTACSS9847.2023.10390302
5	Research of Security Situational Awareness and Visualization Approach in Cloud Computing	10.1109/NANA.2018.8648738
6	Cloud Storage Security Risks, Practices and Measures: A Review	10.1109/INOCOS50539.2020.9298281
7	Ethereum Powered E-Tendering System	10.1109/IC3IoT60841.2024.10550251
8	Security threat probability computation using Markov Chain and Common Vulnerability Scoring System	10.1109/ATNAC.2018.8615386
9	Public Key Infrastructure Approaches Based on Blockchain	10.1109/SSD61670.2024.10549485
10	IDPS based framework for security in green cloud computing and comprehensive review on existing frameworks and security issues	10.1109/CCCS.2015.7374153

شکل 15 خروجی مرتبط با لیست مقالات پیشنهاد داده شده براساس ورودی کاربر

مقاله مرجع در ارتباط با blockchain و security و data protection است که با توجه به امتیازی که به سایر مقالات تعلق گرفته براساس شباهت چکیده و کلمات کلیدی، این موضوعات در مقالات پیشنهاد شده نیز مشاهده میشود .

توضیح کدها:

- فایل insert.py:

برای ایندکس کردن داده های JSON از دو فایل جداگانه به Elasticsearch استفاده می شود.

- مراحل:

اتصال به Elasticsearch:

به Elasticsearch با استفاده از URL، احراز هویت API و غیرفعال کردن تأیید گواهی SSL متصل می شود.

- تابع load_json:

برای خواندن داده های JSON از یک فایل مشخص شده استفاده می شود.

فایل را با استفاده از open() در حالت خواندن ('r') و رمزگذاری UTF-8 باز می کند.

داده های JSON را با استفاده از load.json() بارگیری می کند و آنها را برمی گرداند.

- تابع `convert_to_date`:

برای تبدیل رشته تاریخ در هر مقاله به `datetime` استفاده می شود.

رشته تاریخ را با استفاده از `paper.get("Date of Publication")` بازیابی می کند.

اگر رشته تاریخ وجود داشته باشد:

رشته را با استفاده از `parser.dateutil` به `datetime` تبدیل کند.

در صورت موفقیت `datetime` را به عنوان `"Date of Publication"` در مقاله ذخیره می کند و تاریخ را چاپ می کند.

مقاله را با تاریخ تبدیل شده برمی گرداند.

- بارگذاری داده ها:

کد داده ها را از دو فایل JSON جداگانه با استفاده از `load_json` بارگیری می کند.

- تابع `insert`:

برای ایندکس کردن داده ها در `Elasticsearch` استفاده می شود.

داده ها را به عنوان ورودی دریافت می کند و نام شاخص را به عنوان `arg` می گیرد.

تاریخ های رشته را در داده ها با استفاده از `convert_to_date` به `datetime` تبدیل می کند.

لیستی از `actions` را ایجاد می کند که هر کدام شامل `index_`، `source_` (داده های مقاله) و `id_` (شناسه منحصر به فرد برای هر مقاله) است.

از `bulk.helpers` برای ایندکس کردن `batch` داده ها در `Elasticsearch` به طور همزمان استفاده می کند.

- فایل `app.py`:

`app = Flask(name)`: یک نمونه برنامه `Flask` ایجاد می کند.

- اتصال به `Elasticsearch`:

به `Elasticsearch` با استفاده از `URL`، احراز هویت `API` و غیرفعال کردن تأیید گواهی `SSL` متصل می شود.

- تابع `create_kibana_query`:

لیستی از نتایج جستجو (`results`) را از `Elasticsearch` دریافت می کند.

اگر نتایج وجود داشته باشد، یک پرس و جو `Kibana` را با استفاده از یک عبارت `"should"` بولین با یک پرس و جو `"match"` برای عنوان هر نتیجه (با فرض وجود `title` در اسناد `Elasticsearch`) ایجاد می کند.

دیکشنری پرس و جو `Kibana` یا `None` را در صورت عدم وجود نتیجه برمی گرداند.

- تابع `get_dashboard_url`:

رشته پرس و جو `Elasticsearch` رمزگذاری شده `JSON` (`query`) را دریافت می کند.

`URL` برای نمایش داشبورد `Kibana` را با درج رشته پرس و جو ارائه شده ایجاد می کند.

URL کامل نمایش داشبورد Kibana را برمی گرداند.

- home Route (GET Request):

قالب search.html را render می کند.

- search Route (POST Request):

پرس و جویهای جستجوی کاربر را مدیریت می کند.

داده های فرم را برای فیلدها، پرس و جویها و عملگرها بازیابی می کند.

پرس و جویهای Elasticsearch را با استفاده از یک عبارت "must" یا "should" بولین بسته به عملگرها ایجاد می کند.

جستجو را در شاخص Elasticsearch papers_relevance و papers_newest اجرا می کند.

بر اساس نتایج جستجو، یک پرس و جو Kibana و یک URL قابل جاسازی ایجاد می کند.

قالب dashboard.html را با URL جاسازی Kibana render می کند.

- rs_search Route (POST Request):

درخواست های کاربر برای مقالات مشابه را با استفاده از عملکرد سیستم توصیه (در rs.py پیاده سازی شده است) مدیریت می کند.

داده های فرم را برای فیلد، عنوان یا DOI و تعداد نتایج بازیابی می کند.

برای یافتن مقالات مشابه، تابع get_similar_papers را از rs.py فراخوانی می کند.

قالب rs.html را با لیست مقالات مشابه render می کند.

- Main:

- app.run(debug=True): برنامه Flask را در حالت debug اجرا می کند، که برای توسعه مفید است اما در محیط های

تولیدی توصیه نمی شود.

- rs.py (Recommender System Module):

- تابع get_similar_papers:

- ورودی ها:

es: شیء کلاینت Elasticsearch که برای تعامل با پایگاه داده Elasticsearch استفاده می شود.

target_field: نام فیلدی در اسناد Elasticsearch که برای جستجوی مقالات مشابه بر اساس آن استفاده می شود (مانند "title" یا "DOI").

target_value: مقداری که باید در فیلد target_field جستجو شود.

count: تعداد مقالات مشابهی که باید به عنوان نتایج برگردانده شوند.

- مراحل:

ایجاد پرس و جو Elasticsearch: تابع create_query را فراخوانی می کند تا یک پرس و جو جستجو ایجاد کند که مقالات مربوطه را بر اساس target_field و target_value بازیابی می کند.

جستجوی مقالات: از شیء `es` برای اجرای پرس و جوی جستجو در شاخص های `papers_relevance` و

`papers_newest` استفاده می کند. نتایج جستجو در متغیری به نام `res` ذخیره می شوند.

استخراج مقاله هدف: اولین مقاله بازیابی شده در `res['hits']['hits']` مقاله هدفی است که برای آن مقالات مشابه باید یافت شوند.

محاسبه شباهت: تابع `calculate_similarity` را با پارامترهای `data`، `target_paper` و `count` فراخوانی می کند تا مقالهها مشابه را بر اساس TF-IDF و شباهت کسینوس پیدا کند.

بازگشت نتایج: لیستی حاوی `title` و `DOI` های `count` مقاله مشابه برتر را برمی گرداند.

- تابع `calculate_similarity`:

- ورودی ها:

`data`: لیستی از تمام مقالات موجود در مجموعه داده، که به طور بالقوه از فایل های JSON بارگیری می شوند.

`target_paper`: مقاله ای که برای آن مقالات مشابه باید یافت شوند.

`count`: تعداد مقالات مشابهی که باید به عنوان نتایج برگردانده شوند.

- مراحل:

پیش پردازش کلمات کلیدی: کلمات کلیدی را از فیلدهای `"IEEE keywords"` و `"Author Keywords"` در `target_paper` استخراج و به یک رشته واحد تبدیل میکند.

تکرار در فیلدهای متنی: در فیلدهای متنی مرتبط (`title`، `IEEE Keywords`، `Author Keywords`، `Abstract`) در هر مقاله از لیست `data` تکرار می کند:

محتوای متنی را از فیلد فعلی برای هر مقاله استخراج می کند.

یک نمونه از `TfidfVectorizer` را برای یادگیری اهمیت کلمات در کل مجموعه داده ایجاد می کند.

`vectorizer` را بر روی محتوای متنی استخراج شده آموزش می دهد و یک نمایش TF-IDF از آن ایجاد می کند.

محتوای متنی `target_paper` را برای فیلد فعلی با استفاده از `vectorizer` آموزش دیده به یک بردار TF-IDF تبدیل می کند.

شباهت کسینوس بین بردار TF-IDF `target_paper` و بردارهای TF-IDF تمام مقالات دیگر را برای فیلد فعلی محاسبه می کند.

ترکیب شباهت ها: یک دیکشنری به نام `similarities` ایجاد می کند تا نمرات شباهت کسینوس را برای هر فیلد متنی ذخیره کند.

به شباهت عنوان (فیلد `title`) وزنی معادل 0.5 نسبت به سایر فیلدها اختصاص می دهد (3 مورد دیگر وزنی برابر با 1 دارند).

نمرات شباهت را از تمام فیلدها در یک لیست نمره واحد به نام `score_list` ترکیب می کند.

رتبه بندی مقالات: از `argsort` برای یافتن شاخص های مقالهها با بالاترین نمرات شباهت ترکیبی در `score_list` استفاده می کند. و همچنین شاخص `target_paper` را از لیست نهایی حذف می کند.

استخراج نتایج برتر: لیستی به نام `top_docs` حاوی `title` و `DOI` های مقاله ها مشابه برتر بر اساس شاخص های رتبه بندی شده در `top_indices` ایجاد می کند.

- تابع `create_query`:

- بر اساس نام فیلد (`target_field`) و مقدار (`target_value`) ارائه شده، یک پرس و جوی Elasticsearch ایجاد می کند.

- مراحل:

یک دیکشنری به نام `query` ایجاد می کند.

در داخل دیکشنری `query`، یک دیکشنری دیگر به نام `"bool"` تعریف می کند. این دیکشنری نحوه ترکیب چندین شرط در پرس و جو را مشخص می کند.

در داخل دیکشنری `"bool"`، یک لیست به نام `"must"` ایجاد می کند. این لیست شرایطی را مشخص می کند که همه اسناد باید برای تطابق با آن ها برآورده شوند.

در داخل لیست `"must"`، یک دیکشنری به نام `"match"` ایجاد می کند. این دیکشنری یک شرط تطابق فیلد و مقدار را تعریف می کند.

دیکشنری `"match"` دارای:

`target_field`: این کلید به طور پویا بر اساس آرگومان ورودی تابع (`target_field`) تنظیم می شود. فیلد را در اسناد Elasticsearch مشخص می کند که باید با آن مطابقت داشته باشد.

`target_value`: این `value` نیز به طور پویا بر اساس آرگومان ورودی تابع (`target_value`) تنظیم می شود. مقداری را که فیلد در اسناد باید با آن مطابقت داشته باشد مشخص می کند.

- مقدار برگشتی:

دیکشنری (`query`) را که بر اساس آرگومان های ورودی (`target_field, target_value`) ساخته شده است، برمی گرداند.

- بارگذاری داده ها:

تابع `load_json` و محتوای این فایل های JSON را می خواند.

لیست مقالات را از هر فایل JSON استخراج می کند و به ترتیب آنها را به متغیرهای `data_newest` و `data_relevance` اختصاص می دهد.

- ترکیب داده ها:

`data = data_newest + data_relevance`: لیست مقالات را از هر دو فایل JSON در یک لیست واحد به نام `data` ترکیب می کند. این یک مجموعه داده یکپارچه برای سیستم توصیه ارائه می دهد.

- `Query.py`:

برای جستجو در یک خوشه Elasticsearch و ذخیره نتایج در قالب JSON است.

○ اتصال به Elasticsearch:

به Elasticsearch با استفاده از URL، احراز هویت API و غیرفعال کردن تأیید گواهی SSL متصل می شود.

○ تابع save_json:

داده ها (نتایج جستجو) را در قالب JSON ذخیره می کند.

data: داده ای که باید ذخیره شود.

filename: مسیر و نام فایل خروجی JSON. به طور پیش فرض "json.output/output" است.

این تابع فایل را در حالت نوشتن ("w") باز می کند، داده ها را با استفاده از json.dump ذخیره می کند.

○ تابع search:

جستجوهای Elasticsearch را بر اساس لیستی از کوئری ها اجرا می کند.

queries: لیستی از دیکشنری ها، که هر کدام یک کوئری Elasticsearch را نشان می دهد.

یک دیکشنری خالی به نام json_result را برای ذخیره نتایج جستجو برای هر کوئری ایجاد می کند.

حلقه for هر کوئری را در لیست queries پیمایش می کند:

جستجوی Elasticsearch را با استفاده از es.search با query فعلی و شاخص های مشخص شده ('papers_relevance', 'papers_newest') انجام می دهد.

اسناد منبع را از نتایج جستجو با استفاده از لیست (hit["_source"] for hit in res['hits']['hits']) استخراج می کند و آنها را در یک لیست ذخیره می کند.

لیست اسناد منبع را برای کوئری فعلی به دیکشنری json_result با استفاده از فرمت 'f'query{i}' (به عنوان مثال query0، query1 و غیره) به عنوان کلید اضافه می کند.

این تابع دیکشنری json_result حاوی اسناد بازبایی شده را برای هر کوئری برمی گرداند.

○ تابع main:

قطعه کد اصلی که هر نتیجه جستجو را به طور مستقیم چاپ می کرد، نظر داده شده است. در عوض، از تابع search برای جمع آوری نتایج و ذخیره آنها در یک فایل JSON استفاده می شود.

کوئری های نمونه: لیستی از چهار query ایجاد می کند:

کوئری اول مقالاتی را با عبارت دقیق "cloud" و "security" در عنوان با استفاده از match_phrase با اپراتورهای bool جستجو می کند.

کوئری دوم مقالاتی را که بین 1 ژانویه 2000 و 1 ژانویه 2010 با استفاده از کوئری range در فیلد "Date of Publication" منتشر شده اند، جستجو می کند.

کوئری سوم مقالاتی با حداقل 10 استناد در فیلد "Cites in Papers" را با استفاده از کوئری range با gte (بزرگتر یا مساوی) جستجو می کند.

کوئری چهارم مقالاتی را که حداقل یک نویسنده از USA باشد.

مشاركت:

هادى امينى	تينى توكلى
10	10