Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - IS184853

INTEGRASI DATA PRODUK HALAL BERDASARKAN JARAK KESAMAAN KONSEPTUAL DAN TEKSTUAL KATA

INTEGRATION OF DATA HALAL PRODUCTS BASED ON SIMILARITY DISTANCE OF CONSEPTUAL AND TEXTUAL OF WORDS

MIFTAHUL JANNAH
NRP 05211540000153

Dosen Pembimbing
Nur Aini Rakhmawati, S.Kom, M.Sc.Eng, Ph.D

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

TUGAS AKHIR - IS184853

INTEGRASI DATA PRODUK HALAL BERDASARKAN JARAK KESAMAAN KONSEPTUAL DAN TEKSTUAL KATA

MIFTAHUL JANNAH
NRP 05211540000153

Dosen Pembimbing
Nur Aini Rakhmawati, S.Kom, M.Sc.Eng, Ph.D

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

UNDERGRADUATE THESIS - IS184853

INTEGRATION OF DATA HALAL PRODUCTS BASED ON SIMILARITY DISTANCE OF CONSEPTUAL AND TEXTUAL OF WORDS

MIFTAHUL JANNAH
05211540000153

Supervisor
Nur Aini Rakhmawati, S.Kom, M.Sc.Eng, Ph.D

INFORMATION SYSTEM DEPARTMENT
Information Technology and Communication Faculty
Sepuluh Nopember Institute of Technology
Surabaya 2019

LEMBAR PENGESAHAN

INTEGRASI DATA PRODUK HALAL BERDASARKAN JARAK KESAMAAN KONSEPTUAL DAN TEKSTUAL KATA

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi Institut
Teknologi Sepuluh Nopember

Oleh:

MIFTAHUL JANNAH
NRP. 0521 15 4000 0153

Surabaya, 10 Juli 2019

KEPALA DEPARTEMEN
SISTEM INFORMASI

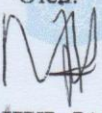
Mahendrawathi ER, S.T., M.Sc., Ph.D.
NIP 19761011 200604 2 001

LEMBAR PERSETUJUAN

INTEGRASI DATA PRODUK HALAL BERDASARKAN JARAK KESAMAAN KONSEPTUAL DAN TEKSTUAL KATA

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi Institut
Teknologi Sepuluh Nopember

Oleh:



MIFTAHUL JANNAH
NRP. 0521 15 4000 0153

Disetujui Tim Penguji: Tanggal Ujian: 10 Juli 2019
Periode Wisuda: September 2019

Nur Aini Rakhmawati, S.Kom, M.Sc.Eng, Ph.D **(Pembimbing)**

Faizal Johan Atletiko, S.Kom., M.T **(Penguji I)**

Irmasari Hafidz, S.Kom, M.Sc **(Penguji II)**

INTEGRASI DATA PRODUK HALAL BERDASARKAN JARAK KESAMAAN KONSEPTUAL DAN TEKSTUAL KATA

Nama Mahasiswa	**:**	**Miftahul Jannah**
NRP	**:**	**05211540000153**
Departemen	**:**	**Sistem Informasi FTIK-ITS**
Pembimbing I	**:**	**Nur Aini Rakhmawati, S.Kom, M.Sc.Eng, Ph.D**

ABSTRAK

Halal Nutrition Food merupakan produk riset berupa aplikasi yang memberikan pengguna informasi tentang produk makanan halal. Tidak hanya itu, Halal Nutrition Food juga menyediakan informasi nutrisi dan komposisi dari produk tersebut. Namun, data yang ada pada aplikasi ini masih perlu diperkaya dan diolah kembali. Open Food Facts merupakan situs yang menyediakan database mengenai produk makanan (seperti nama produk, komposisi, zat aditif) dimana semua orang di berbagai belahan dunia bisa berkontribusi untuk menambahkan data ke dalamnya maupun menggunakan kembali data yang ada. Pada tugas akhir ini dilakukan pengambilan data dari open food facts kemudian diintegrasikan dengan data pada aplikasi halal nutrition food. Data produk yang akan diintegrasikan diunduh dari situs web open food facts. Data yang ada sangat kotor dan perlu diolah sebelum dilakukan ke proses selanjutnya. Data tersebut diolah pada proses pre-processing data sedemikian rupa sehingga data yang ada dapat diolah ke proses pengukuran kemiripan data. Untuk mengurangi redundansi pada data bahan makanan, penulis melakukan pengukuran kemiripan bahan makanan menggunakan pengukuran kemiripan konseptual dengan Wordnet yang mengukur kemiripan antara dua dataset secara makna kata (sinonim). Selain itu, penulis juga mengukur kemiripan bahan makanan secara tekstual dengan fuzzy string matching, yaitu Levenshtein distance, Jaro-Winkler distance,

dan Jaccard distance. Setelah pengukuran kemiripan bahan makanan dilakukan, hasil pengukuran kemiripan diuji untuk mengetahui akurasinya. Dari pengujian tersebut, ditemukan bahwa kombinasi pengukuran kemiripan menggunakan Levenshtein distance (tekstual) dan Wordnet similarity (konseptual) adalah yang paling optimal mengukur kemiripan data bahan makanan sehingga bahan makanan yang diintegrasikan bebas dari redundansi.

Kata kunci: halal, open food facts, wordnet, levenshtein distance, jaro-winkler distance, jaccard distance, integrasi.

INTEGRATION OF DATA HALAL PRODUCTS BASED ON SIMILARITY DISTANCE OF CONSEPTUAL AND TEXTUAL OF WORDS

Name : Miftahul Jannah
NRP : 05211540000153
Department : Information System FTIK-ITS
Supervisor : Nur Aini Rakhmawati, S.Kom, M.Sc.Eng, Ph.D

ABSTRACT

Halal Nutrition Food is a research product in the form of an application that gives users information about halal food products. Not only that, Halal Nutrition Food also provides nutritional information and composition of these products. However, the data contained in this application still needs to be enriched and reprocessed. Open Food Facts is a site that provides a database of food products (such as product names, compositions, additives) where everyone in various parts of the world can contribute to adding data to it or reusing existing data. In this final project, data retrieval from open food facts is then integrated with the data on the halal nutrition food application. Product data to be integrated is downloaded from the open food facts website. The data is very dirty and needs to be processed before going to the next process. The data is processed in the process of pre-processing data in such a way that existing data can be processed into the process of measuring data similarity. To reduce redundancy in food data, measurements of the similarity of food ingredients were carried out using measurement of conceptual similarity with Wordnet which measures the similarity between two datasets in word meaning (synonym). In addition, the authors also measured textual similarity of foodstuffs with fuzzy string matching, namely Levenshtein distance, Jaro-Winkler distance, and Jaccard distance. After measuring the similarity of food ingredients, the results of the similarity measurements were

tested to determine their accuracy. From the test, it was found that the combination of measurements of similarity using Levenshtein distance (textual) and Wordnet similarity (conceptual) is the most optimal measure of the similarity of food data so that food ingredients are integrated free from redundancy.

Keywords: halal, open food facts, wordnet, levenshtein distance, jaro-winkler distance, jaccard distance, integration.

KATA PENGANTAR

Segala puji bagi Allah yang hanya kepadaNya kami memuji, memohon pertolongan, dan memohon ampunan. Puji syukur kehadirat Allah, Tuhan Yang Maha Esa, karena dengan limpahan rahmat-Nya penulis dapat menyelesaikan laporan tugas akhir dengan judul Integrasi Data Produk Halal Berdasarkan Jarak Kesamaan Konseptual dan Tekstual Kata.

Terima kasih penulis sampaikan kepada pihak-pihak yang telah mendukung, memberikan saran, motivasi, semangat, dan bantuan baik berupa materiil maupun moril demi tercapainya tujuan pembuatan tugas akhir ini. Tugas akhir ini tidak akan pernah terwujud tanpa bantuan dan dukungan dari berbagai pihak yang sudah meluangkan waktu, tenaga dan pikirannya. Secara khusus penulis akan menyampaikan ucapan terima kasih yang sebanyak-banyaknya kepada:

1. Allah SWT, yang senantiasa melimpahkan berkah dan rahmat-Nya selama penulis mengerjakan Tugas Akhir ini.
2. Orangtua dan keluarga penulis, yang memberikan dukungan penuh, motivasi dan do'a yang senantiasa tercurah untuk kesuksesan anaknya, terutama dalam perkuliahan.
3. Ibu Nur Aini Rakhmawati, S.Kom., M.Sc.Eng., Ph.D. selaku dosen pembimbing yang telah meluangkan waktu, tenaga dan pikiran, memberikan ilmu, petunjuk, dan motivasi selama pengerjaan Tugas Akhir ini.
4. Bapak dan ibu dosen penguji, Pak Faizal Johan Atletiko, S.Kom., M.T, dan Ibu Irmasari Hafidz, S.Kom, M.Sc, yang telah memberikan saran dan kritik yang membangun untuk suksesnya pengerjaan tugas akhir ini.
5. JMMI dan Mutiara, yang telah memberikan dukungan dan motivasi kepada penulis.
6. Teman-teman Lannister, yang telah membersamai dan memberikan bantuan dan semangat kepada penulis.

7. Jajaran pengurus Lab Studio Aplikasi Terapan, dan Lab ADDI, yang memberikan sarana prasarana untuk suksesnya pengerjaan Tugas Akhir ini.
8. Segenap dosen dan karyawan Departemen Sistem Informasi, yang secara langsung maupun tidak langsung membantu menyukseskan pengerjaan Tugas Akhir ini.
9. Dan pihak lainnya yang berkontribusi dalam tugas akhir yang belum dapat penulis sebutkan satu per satu.

Penyusunan tugas akhir ini masih jauh dari kata sempurna, untuk itu penulis menerima segala kritik dan saran yang membangun sebagai upaya menjadi lebih baik lagi ke depannya. Semoga buku tugas akhir ini dapat memberikan manfaat untuk pembaca.

Surabaya, 10 Juli 2019

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

Pada bab ini penulis akan menjelaskan mengenai hal-hal yang mendorong atau melatarbelakangi dilakukannya tugas akhir ini. Bab ini akan diuraikan menjadi beberapa komponen, yaitu latar belakang masalah, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan relevansi yang dimiliki tugas akhir ini.

1.1 Latar Belakang

Indonesia merupakan negara yang menganut pancasila dimana sila pertamanya adalah ketuhanan Yang Maha Esa. Masyarakatnya menganut beragam agama yang berbeda, namun masyoritas menganut agama Islam. Umat muslim di Indonesia berdasarkan sensus penduduk tahun 2010 mencapai 87,18% (tidak kurang dari 207 juta orang) [1]. Dalam kesehariannya sebagai muslim, seseorang harus melakukan salah satu syari'at agamanya, yaitu tidak boleh memakan makanan kecuali yang halal. Telah banyak produk-produk makanan di Indonesia ini yang sudah berlabel halal. Baik itu mengklaim halal pada produknya sendiri atau bahkan mengantongi sertifikat halal dari lembaga sertifikasi halal di negara produksinya.

Di Indonesia, lembaga sertifikasi halal yang diakui adalah Lembaga Pengkajian Pangan Obat obatan dan Kosmetika Majelis Ulama Indonesia (LPPOM MUI) yang didirikan sejak 1989 [2]. Pada situs web LPPOM MUI, kita bisa melakukan pencarian produk halal serta informasi sertifikatnya. Selain itu, LPPOM MUI juga mengakui 45 lembaga sertifikasi halal di berbagai belahan dunia (Asia, Australia & New Zealand, Eropa, Amerika, dan Afrika Selatan) per Januari 2018 [3].

Tercatat ada 727.617 jumlah produk yang tersertifikasi halal pada data LPPOM MUI pada periode 2011-2018 [4]. Meski demikian, informasi yang disediakan pada sistem pencarian produk halal yang telah tersertifikasi oleh MUI di situs LPPOM MUI (http://www.halalmui.org) hanya mencakup nama produk, nomor sertifikat halal, produsen, dan tanggal berlaku sertifikat.

Pada sistem pencarian ini tidak disediaka akses informasi mengenai komposisi dan nutrisi dari suatu produk tersebut.

Halal Nutrition Food merupakan produk riset berupa aplikasi yang memberikan pengguna informasi tentang produk makanan halal. Tidak hanya itu, Halal Nutrition Food juga menyediakan informasi nutrisi dan komposisi dari produk tersebut [5], dimana tidak didapakan dari situs pencarian produk halal LPPOM MUI. Halal Nurition Food menyediakan fitur pencarian dimana pengguna dapat mengakses informasi-informasi diatas tersebut dengan mudah.

Namun, data yang ada pada aplikasi ini masih perlu pengolahan dan penambahan. Data produk yang ada saat ini berjumlah 389.145 produk entri. Diantara data tersebut, masih terdapat data yang kotor, seperti redundansi bahan makanan. Data yang ada pada Halal Nutrition Food bisa lebih diperlengkap dengan data dari open food facts yang hampir setiap harinya mempublikasikan tambahan data baru [6].

Open Food Facts merupakan situs yang menyediakan database yang bersifat terbuka tentang informasi suatu produk makanan dimana semua orang di berbagai belahan dunia bisa berkontribusi untuk menambahkan data produk ke dalamnya. Database ini di publish dalam sebagai *open-data* sehingga bisa digunakan kembali oleh siapapun [6].

Untuk mengurangi redundansi dari data bahan makanan pada database Halal Nutrition Food, perlu dilakukan pengukuran kesamaan bahan makanan dan *replacement* sehingga bahan makanan yang sama tetapi tertulis dengan istilah berbeda bisa distandarisasi menjadi istilah yang sama. Terdapat dua jenis metode yang akan dimanfaatkan penulis untuk melakukan hal ini, yaitu WordNet::Similarity::path dan Fuzzy String Matching. Dimana Wordnet::Similarity::path mengukur jarak kesamaan antara dua kata secara konseptual (sinonim), sedangkan fuzzy string matching mengukur jarak kesamaan karakter diantara keduanya secara tekstual. Bahan makanan yang dikomparasi disini adalah antara bahan makanan pada

Halal Nutrition Food dengan bahan makanan pada Open Food Facts.

1.2 Rumusan Masalah

Rumusan masalah yang akan diselesaikan dalam Tugas Akhir ini adalah:

1. Bagaimana mengurangi rendundansi data bahan makanan pada database Halal Nutrition Food ketika diintegrasikan dengan database Open Food Facts?
2. Bagaimana melakukan pengukuran kesamaan bahan makanan antara data pada database Halal Nutrition Food dengan Open Food Facts dengan menggunakan pengukuran jarak konseptual antar kata dan jarak tekstual antar kata?
3. Bagaimana mengintegrasikan data Open Food Facts ke data Halal Nutrition Food?

1.3 Batasan Permasalahan

1. Hasil dari Tugas Akhir ini adalah database Halal Nutrition Food yang telah terintegrasi dengan data pada Open Food Facts.
2. Data produk yang diambil dari database Open Food Fact [6] merupakan produk yang mengandung bahan makanan berbahasa inggris dan perancis.
3. Data produk yang diambil dari Open Food Facts hanya produk yang memiliki data bahan makanan saja.

1.4 Tujuan

1. Mengurangi rendundansi bahan makanan pada database Halal Nutrition Food ketika diintegrasikan dengan database Open Food Facts.
2. Melakukan pengukuran kesamaan bahan makanan antara data pada database Halal Nutrition Food dengan Open Food Facts dengan menggunakan pengukuran jarak

sinonim antar kata dan jarak kesamaan karakter antar kata, serta mengetahui metode yang paling cocok.
3. Mengintegrasikan data Open Food Facts ke data Halal Nutrition Food.

1.5 Manfaat

1. Bagi pengguna: Pengguna dapat mengakses informasi pada aplikasi Halal Nutrition Food yang disajikan secara lebih rapih, akurat, dan kaya informasi. Sehingga aplikasi ini menjadi lebih kredibel dan dapat digunakan untuk kebermanfaatan yang lebih luas baik di bidang teknologi, sosial, budaya, dan sebagainya.
2. Bagi peneliti: Peneliti dapat memahami dan menghasilkan penelitian terkait pengembangan perangkat lunak linked data mengenai produk halal khususnya pada pengolahan data beserta teknologinya. Dimana hasil penelitian ini bisa digunakan kembali dan atau dikembangkan menjadi lebih baik lagi.

1.6 Relevansi

Tugas akhir ini memiliki relevansi pada beberapa mata kuliah di Departemen Sistem Informasi ITS, yaitu pengantar basis data, teknologi web, dan manajemen dan administrasi basis data.

Tugas akhir ini juga memiliki relevansi pada salah satu laboratorium di departemen sistem informasi, yaitu Laboratorium Ananlisis Data dan Diseminasi Informasi (ADDI), dimana terdapat proses yang menggunakan keilmuan berkaitan dengan pembersihan data, serta diseminasi informasi. Tugas akhir ini juga sejalan dengan *research roadmap* yang ada pada lab ADDI dimana terdapat salah satu topik yaitu integrasi data halal nutrition food.

BAB II
TINJAUAN PUSTAKA

Bab tinjauan pustaka terdiri dari landasan-landasan yang akan digunakan dalam penelitian tugas akhir ini, mencakup penelitian-penelitian sebelumnya, kajian pustaka, dan metode yang digunakan selama pengerjaan.

2.1 Penelitian Terkait

2.1.1. Integrasi Data dan Visualisasi Graf pada Aplikasi Halal Nutrition Food

Penelitian yang dilakukan oleh Berlian Fajar Yusuf [7] ini mengintegrasikan data pada Aplikasi Halal Nutrition Food, serta membuat visualisasinya dalam bentuk graf. Sebelum pengintegrasian, data pada Open Food Facts dilakukan data cleansing terlebih dahulu yang terdiri dari beberapa tahapan, salah satunya pengukuran kesamaan pada bahan makanan. Kemudian data diintegrasikan dengan melakukan indeks menggunakan Apache Lucene. Kemudian visualisasi dilakukan menggunakan METIS graph partitioning. Pengukuran kesamaan bahan makanan (pada data cleansing) pada penelitian ini menggunakan Levenshtein dan Jaccard distance untuk memperbaki kesalahan penulisan. Penelitian kami melanjutkan penelitian ini dengan mencoba melakukan pengukuran kesamaan bahan makanan mengunakan menggunakan metode lain, yang akan dibahas lebih lanjut pada bab 3.

Berdasarkan eksperimen yang dilakukan, pengukuran kesamaan oleh Leacock dan Chodorow adalah yang paling cocok untuk kasus ini, dibandingkan metode lainnya.

2.1.2. Combining local context and WordNet Similarity For Word Sense Identification

Penelitian ini dilakukan oleh Leacock dan Chodorow [8] untuk mengukukur kesamaan semantik menggunakan path-length

similarity pada WordNet. Pada tugas akhir ini, kami akan menggunakan metode yang diajukan Leacock dan Chodorow ini untuk melakukan pengukuran kesamaan bahan makanan.

2.1.3. Using WordNet and Semantic Similarity to Disambiguate an Ontology

Penelitian yang dilakukan oleh Martin Warin dari Stockholms Universitet [9] ini membandingkan beberapa metode pengukuran kesamaan semantik untuk memperkaya suatu ontologi yang disebut Common Procurement Vocabulary (CPV) menggunakan ukuran untuk *semantic similarity* dan WordNet.

Penelitian ini membandingkan kesamaan semantik antara kata yang berada pada leaf-node dengan kata yang ada pada parent-node dalam sebuah ontologi. Leaf-node yang mendapatkan skor kesamaan tertinggi lebih mirip secara semantik dengan parent-node, akan dipilih untuk memperkaya leaf-node.

2.1.4. Assessing Sentence Similarity Using WordNet based Word Similarity

Paper ini menyajikan cara menghitung kesamaan antara teks dan kalimat yang sangat singkat tanpa menggunakan korpus literatur eksternal. Metode ini menggunakan WordNet. Hasil perhitungan diverifikasi melalui percobaan. Eksperimen ini dilakukan pada dua set pasangan kalimat terpilih. Penulisnya menunjukkan bahwa teknik ini lebih baik dibandingkan dengan pengukuran kesamaan berdasarkan kata lainnya dan cukup fleksibel untuk memungkinkan pengguna membuat perbandingan tanpa kamus tambahan atau informasi corpus [10].

2.2 Dasar Teori

2.2.1. Halal Nutrition Food

Halal Nutrition Food merupakan produk riset berupa aplikasi yang memberikan pengguna informasi tentang halal serta nutrisi dari produk makanan dan minuman [5]. Aplikasi ini pertama dikembangkan oleh Irfan Rizki Ananda pada tahun 2015 dengan mengembangkan aplikasi open linked data produk makanan halal.

Selanjutnya pada tahun 2016 penelitian yang berjudul "Rancang Bangun Perangkat Lunak Linked Open Data Halal Dan Gizi Pada Produk Makanan Dan Minuman" dilakukan oleh Jauhar Fatawi [11]. Beliau mengembangkan aplikasi Halal Nutrition Food dengan menambahkan informasi komposisi dan zat aditif pada produk halal.

Lalu aplikasi ini dikembangkan lagi oleh Adnan Mauludin Fajriyadi dengan judul "Peningkatan Relevansi Pencarian Produk Halal dalam Aplikasi Halal Nutrition Menggunakan Algoritma OKAPI BM25F" [12]. Beliau melakukan penelitian ini untuk meningkatkan akurasi dan relevansi pencarian yang berbasis pada produk.

Pada pengembangan selanjutnya, Azmi Adi Firmansyah mengembangkan sistem pencarian produk terkait dengan mengukur kemiripan produk menggunakan Euclidean Distance dan Cosine Similarity [13]

Kemudian penelitian berjudul "Rancang Bangun Aplikasi Android Halal Nutrition Food Menggunakan Kombinasi Query-Independent dan Query-Dependent Ranking" oleh Ahmad Choirun Najib [14] dilakukan untuk mengembangkan pencarian tidak hanya berdasarkan dataset produk, tetapi informasi pada dataset entitas lain seperti komposisi produk. Hal tersebut diwujudkan dengan memberi skor pada entitas dalam dataset dengan batasan-batasan tertentu, serta memberi bobot pada kueri pengguna setelah pencarian dilakukan.

Untuk memperkaya informasi yang ada pada database Halal Nutrition Food, pengembangan dilakukan dengan mengintegrasikannya dengan data pada Open Food Facts. Pengembangan ini dilakukan oleh Berlian Fajar Yusuf dalam penelitiannya yang berjudul "Integrasi Data dan Visualisasi Graf pada Aplikasi Halal Nutrition Food" [7]. Selain itu, beliau juga membuat visualisasinya berupa graf.

2.2.2. Open Food Facts

Open-Food-Facts adalah database produk makanan dengan bahan, alergen, fakta nutrisi dan berbagai informasi yang dapat ditemukan pada label produk.

Database ini bersifat terbuka dimana semua orang di berbagai belahan dunia bisa berkontribusi untuk menambahkan data produk ke dalamnya [6]. Database ini di publish dalam sebagai *open-data* sehingga bisa digunakan kembali oleh siapapun. Dilaunching pada 19 Mei 2012, Open Food Facts sekarang memiliki lebih dari 320.000 produk dengan hampir 6.000 kontributor dari berbagai negara [15].

Open Food Facts sangat berguna bagi pengembang teknologi tentang makanan seperti Halal Nutrition Food, karena database Open Food Facts bisa bisa kami untuk memperkaya database produk halal. Database makanan yang ada pada Open Food Facts tidak hanya menyediakan nama produk dan produsennya, tetapi juga komposisi, fakta nutrisi, dan bahan aditif dari produk.

Data lengkap dari semua produk Open Food Facts bisa kita unduh dalam bentuk ekspor MonggoDB dump, CSV (comma-separated values), maupun RDF.

2.2.3. Resource Description Framework (RDF)

RDF adalah model standar untuk pertukaran data di Web. RDF memiliki fitur yang memfasilitasi *data merging* bahkan antar skema yang berbeda [16]. RDF merupakan kerangka kerja untuk mengekpresikan informasi tentang *resources*, yang bisa berupa apapun termasuk orang, benda, konsep, dokumen. RDF

memungkinkan kita untuk membuat pernyataan dengan format yang sederhana [17]:

<subjek> <predikat> <objek>

Subjek dan objek merupakan dua *resources* yang saling berhubungan. Hubungan antara kedua dijelaskan dengan predikat. Bentuk dari 3 elemen ini biasa disebut dengan *triples*. Contoh:

<aisyah> <adalah> <manusia>
<aisyah> <lahir pada> <25 Januari>
<aisyah> <adalah temannya> <fatimah>
<fatimah> <tinggal di> <Surabaya>

Untuk menuliskan graf RDF terdapat sejumlah format serialisasi yang berbeda. Namun, ada banyak cara untuk menuliskan grafik yang sama. Diantaranya kelompok Turtle (N-Triples, Turtle, TriG and N-Quads), berbasis JSON, dan XML syntax [17].

2.2.4. Apache Lucene

Apache Lucene adalah *library* mesin pencari teks berperforma tinggi dan berfitur lengkap yang seluruhnya ditulis dalam bahasa Java. Teknologi ini *open-source* dan bisa dipakai hampir di semua aplikasi yang membutuhkan pencarian teks lengkap, terutama lintas platform [18].

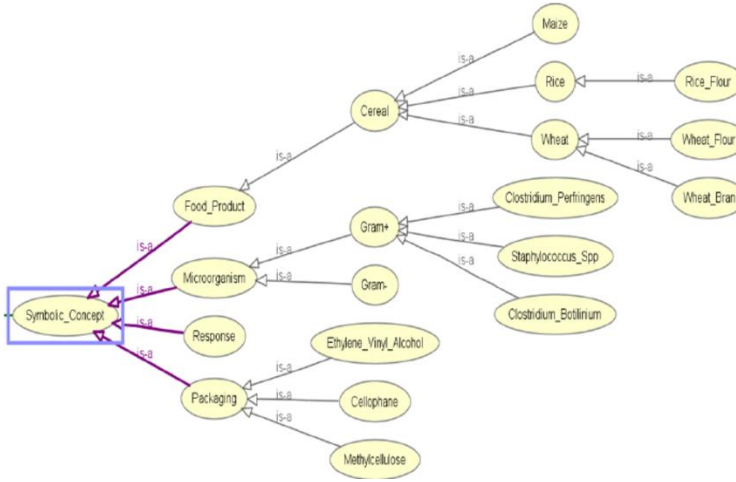
2.2.5. WordNet::Similarity::path

WordNet adalah database leksikal berbahasa Inggris yang dikembangkan oleh Princenton University. Kata benda, kata kerja, kata sifat dan kata keterangan dikelompokkan ke dalam set sinonim (synsets), masing-masing mengekspresikan konsep yang berbeda. Synsets saling berhubungan dengan hubungan leksikal. *Network* yang dihasilkan dari kata-kata dan konsep yang berhubungan secara makna dapat dinavigasi dengan browser. WordNet juga tersedia secara publik untuk diunduh.

Struktur WordNet sangat berguna untuk komputasi linguistik dan pemrosesan bahasa alami [19].

WordNet::Similarity::path merupakan sebuah modul untuk menghitung keterkaitan semantik antar kata dengan menghitung node kata dalam hierarki ‘is-a’ dari WordNet. Panjang sebuah *path* terdiri dari nodes. Semakin panjang *path* maka semakin tidak berelasi kedua kata/konsep. Maka dari itu, nilai kemiripan antara dua konsep berbanding terbalik dengan *path length (distance)* [20]

$$relatedness = \frac{1}{distance}$$



Gambar 2.1 contoh hierarki *is-a* pada WordNet

Gambar di atas adalah contoh ilustrasi hierarki *is-a* yang ada pada WordNet. Dimana *distance* yang dimaksud dalam perhitungan *path-length* adalah menghitung node pada jalur terpendek antara dua konsep.

Contoh:

Relatedness (*wheat_bran*, *wheat_flour*) $= \frac{1}{2}$

$= 0.5$

Pada tugas akhir ini, wordnet akan digunakan sebagai metode pengukuran kemiripan kata secara konseptual, dimana wordnet kesamaan kata berdasarkan sinonim/arti kata.

2.2.6. Semantic Similarity Measure by Leacock-Chodorow

Perhitungan kemiripan semantik oleh Leacock-Chodorow [8] merupakan pengembangan perhitungan *path-length* pada WordNet. Kesamaan antara dua konsep *a* dan *b* sama dengan jumlah node pada jalur terpendek di antara mereka, dibagi dua kali lipat kedalaman maksimum (dari node terendah ke atas) dalam taksonomi di mana *a* dan *b* berada. Jumlah node antara dua *siblings* (dua node dengan *parent* yang sama) adalah tiga.

$$sim_{LCH}(a,b) = -log\frac{length\ (a,b)}{2D}$$

Ukuran Leacock-Chodorow mengasumsikan virtual top node mendominasi semua node, dan karenanya akan selalu mengembalikan nilai yang lebih besar dari nol, selama dua konsep yang dibandingkan dapat ditemukan di WordNet karena akan selalu ada jalan di antara mereka [9].

Berikut contoh perhitungan kemiripan antara 2 bahan yang ada pada database halal yaitu young coconut flesh dan young coconut meat.

id ▲ 1	iName	iType	eNumber	created_at	updated_at
222202	young coconut drink	0	NULL	2018-06-16 00:00:00	2018-06-16 00:00:00
222203	young coconut flesh	0	NULL	2018-06-16 00:00:00	2018-06-16 00:00:00
222204	young coconut juice	0	NULL	2018-06-16 00:00:00	2018-06-16 00:00:00
222205	young coconut juicesugar	0	NULL	2018-06-16 00:00:00	2018-06-16 00:00:00
222206	young coconut kernel	0	NULL	2018-06-16 00:00:00	2018-06-16 00:00:00
222207	young coconut meat	0	NULL	2018-06-16 00:00:00	2018-06-16 00:00:00

Gambar 2.2 contoh data bahan makanan pada Halal Nutrition Food

Maka ketika dimasukkan ke dalam perhitungan wordnet similarity menggunakan metode Leacock-Chodorow (LCH) skor kemiripannya akan menjadi

LCH

	young /JJ	coconut /NN	flesh /NN
young/JJ	-	-	-
coconut/NN	-	3.6889	1.3863
meat/NN	-	2.5903	2.3026

```
[Description]
LCH relies on the length of the shortest path between two
synsets for their measure of similarity.
LCH(s1, s2) = -Math.log_e( LCS(s1, s2).length / ( 2 *
max_depth(pos) ) ).

[Parameters]
 - min score = 0.0
 - max score = Infinity
 - error score = -1.0
 - acceptable pos pairs = [['n', 'n'], ['v', 'v']]
 - use all senses = true
 - use root node = true
 - max depth N = 20
 - max depth V = 14
```

Gambar 2.3 hasil perhitungan LCH dengan http://ws4jdemo.appspot.com/

2.2.7. Fuzzy String Matching

Fuzzy string matching atau bisa juga disebut approximate string matching adalah metode untuk menemukan suatu string yang spesifik di dalam suatu ruang pencarian, dimana mentoleransi terjadinya eror sampai pada tingkat tertentu [21]. Fuzzy string matching pada tugas akhir ini akan digunakan sebagai metode pengukuran kemiripan kata secara tekstual karena metode ini mengukur kesamaan karakter antar kata.

Secara singkat, fuzzy string matching merupakan metode untuk mengukur kesamaan antara dua string dengan menghasilkan angka yang menunjukkan tingkat kemiripan string. Ada

berbagai macam jenis dari fuzzy string matching yang telah dikembangkan, diantaranya adalah Levenshtein Distance, Jaro-Winkler Distance, dan Jaccard Distance

2.2.8. Levenshtein Distance

Levenshtein distance menggunakan edit distance dalam operasinya. Edit distance menghitung jarak minimum pengeditan antara string yang akan di komparasi dengan string target. Levenshtein distance meghitung jumlah penambahan, penghapusan, atau subtitusi karakter yang dibutuhkan untuk membuat dua string menjadi sama [22] [23].

Contoh:

Levenshtein distance diantara “kitten” dan “sitting” adalah 3, karena terdapat tiga pengeditan untuk mengubah string satu ke yang lain, dan tidak ada cara lain untuk melakukannya kurang dari tiga pengeditan:

1. **k**itten → **s**itten (subtitusi karakter "s" terhadap "k")
2. sitt**e**n → sitt**i**n (subtitusi karakter "i" terhadap "e ")
3. sittin → sittin**g** (penambahan karakter "g" di akhir).

2.2.9. Jaro-Winkler Distance

Algoritma Jaro-Winkler adalah algoritma pengukur kemiripan antara dua string. Algoritma ini merupakan algoritma yang ditemukan oleh Matthew A. Jaro pada tahun 1989 dan 1995. Algoritma ini kemudian dikembangkan oleh William E. Winkler dengan memodifikasi Jaro Distance untuk memberikan bobot yang lebih tinggi untuk prefix kemiripan [24] [25]. Algoritma ini melakukan diantara lain [26]:

1. Menghitung panjang string,
2. Menemukan jumlah karakter yang sama di dalam dua string, dan
3. Menemukan jumlah transposisi

Pada algoritma Jaro (d_j) digunakan rumus untuk menghitung jarak antara dua string yaitu s1 dan s2 adalah sebagai berikut:

$$d_j = \begin{cases} 0 & if\ m = 0 \\ \frac{1}{3}\left(\frac{m}{|s1|} + \frac{m}{|s2|} + \frac{m-t}{m}\right) & otherwise \end{cases}$$

dimana :

m adalah jumlah karakter yang sama

|s1| adalah panjang string 1

|s2| adalah panjang string 2

t adalah jumlah transposisi

Dan pada Algoritma Jaro-Winkler (d_w) digunakan skala prefix (p) yang memberi awalan pada set string. Dengan rumus sebagai berikut :

$$d_w = d_j + (lp(1 - d_j))$$

dimana :

d_j adalah hasil perhitungan kemiripan string s1 dan s2 dengan algoritma Jaro Distance

l adalah panjang karakter yang sama pada awalan string sebelum ditemukan adanya ketidaksamaan dengan batas maksimum sampai 4 karakter.

p adalah nilai standar untuk konstanta dalam karya Winkler adalah p = 0,1

Nilai perhitungan yang didapat dari 0 hingga 1. Dengan nilai 0 setara dengan tidak ada kemiripan dan 1 adalah sama persis.

Contoh:
Jika string 1 = TOMATO, string 2 = TOMAOT, maka
m = 6
s_1 = 6
s_2 = 6
t = 1, karakter yang tertukar hanya T dengan O
maka nilai jaro distance (d_j) adalah

$d_j = \frac{1}{3}\left(\frac{6}{6} + \frac{6}{6} + \frac{6-1}{6}\right) = 0.944$
$d_w = 0.944 + (3 \times 0.1\ (1-0.944)) = 0.961$

2.2.10. Jaccard Distance

Koefisien jaccard similarity merupakan metode statistik yang digunakan untuk membandingkan kesamaan dan keragaman set sampel. Koefisien jaccard mengukur kesamaan diantara set sample yang terbatas dan didefinisikan sebagai jumlah irisan dibagi jumlah gabungan dari set sample. [27].

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Keterangan:
J (A,B) = $Jaccard\ Similarity\ Coeficient$
A = $Kata$ A
B = $Kata$ B

Sedangkan jaccard distance mengukur ketidaksamaan antara set sample.

$$d_j\ (A,B) = 1 - J\ (A,B)$$

Keterangan:
d_j (A,B) = $Jaccard\ Distance$

Contoh:
A ={abcde} | B={abdcde}

	ab	**bc**	**cd**	**de**	**dc**	**bd**
A	1	1	1	1	0	0
B	1	0	1	1	1	1

$d_j = 1 - \frac{|A \cap B|}{|A \cup B|}$

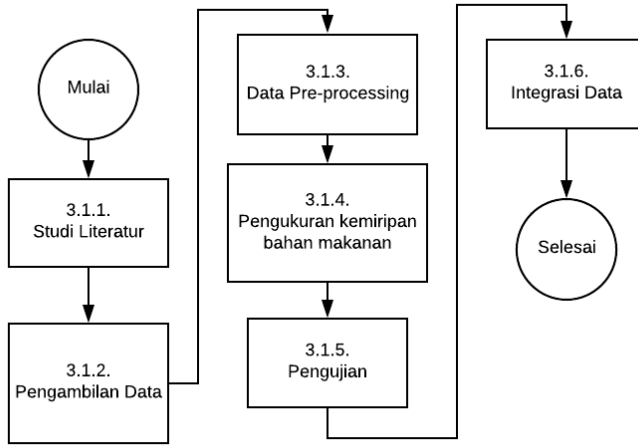
$= 1 - \frac{3}{6}$
$= 0.5$

BAB III
METODOLOGI

Pada bab ini akan dijelaskan bagaimana lalngkah pengerjaaan tugas akhir dengan disertakan deskripsi dari setiap penjelasan untuk masing-masing tahapan beserta jadwal kegiatan pengerjaan tugas akhir.

3.1. Tahapan Pelaksanaan Tugas Akhir

Pada sub-bab ini akan dijelaskan mengenai tahapan-tahapan yang dilakukan untuk melaksanakan tugas akhir. Secara umum, metodologi tersebut dapat dilihat pada Gambar 3.1 berikut



Gambar 3.1 Tahapan pelaksanaan tugas akhir

3.1.1. Studi Literatur

Dalam studi literatur, penulis mengumpulkan beberapa literatur terkait dengan data pre-processing, wordnet::similarity::path oleh Leacock Chodorow, fuzzy string similarity, integrasi data, serta penelitian terkait Halal Nutrition Food sebelumnya.

Kumpulan literatur tersebut dipahami dan dipelajari oleh penulis terlebih dahulu sebagai acuan untuk melakukan tahapan pelaksanaan tugas akhir selanjutnya.

3.1.2. Pengambilan Data

Sumber data yang digunakan pada tugas akhir ini adalah data informasi produk makanan pada situs open food facts [6]. Format data yang diambil adalah format csv. Data ini berisi informasi mengenai produk makanan dari berbagai sumber yang disatukan dalam satu platform open food facts. Informasi yang tersedia di dalam data csv tersebut dintaranya, nama produk, merek produk, komposisi produk, dan fakta nutrisi produk.

3.1.3. Data Pre-processing

Data yang berasal dari Open Food Facts merupakan data yang terhimpun dari berbagai sumber inputan, dimana data tersebut terbilang kotor dan tidak siap diproses. Maka dari itu, perlu adanya data pre-processing untuk membersihkan dan merapihkan data sehingga siap diproses pada tahapan selanjutnya.

1. Drop duplicates,

Menghapus baris data produk makanan yang sama persis.

2. Menghilangkan baris dengan null value,

Menghapus baris data produk makanan yang tidak terisi kolom '*ingredients*'nya.

3. Casefolding

Mengubah semua kata menjadi lower case (huruf kecil). Hal ini dilakukan agar tidak terjadi kesalahan pada proses pengukuran kemiripan bahan makanan.

4. Menghilangkan dan mengganti karakter yang tidak diperlukan

Beberapa karakter yang tidak dibutuhkan maupun karakter yang ada karena kesalahan penulisan, akan dihapus atau diganti dengan karakter terkait.

5. Menerjemahkan data bahan makanan

Pada tahap ini, data bahan makanan masih berupa data yang kotor dimana ada beberapa data bahan makanan masih berbahas bukan inggris. Maka data bahan makanan dideteksi bahasanya, kemudian jika bahasa yang terdeteksi bukan bahasa inggris, maka bahan makanan diterjemahkan ke dalam bahasa inggris.

6. Menghilangkan stopword

Menghapus kata-kata yang tidak diperlukan dan ingin disaring dalam data sebelum data di proses lebih lanjut. Kata-kata ini didefinisikan terlebih dahulu dalam *stop list*. Dalam kasus ini, stopword yang dimaksud adalah semua kata selain bahan makanan.

7. Tokenisasi

Proses ini memisahkan sebuah unit dokumen (kalimat) menjadi beberapa bagian yang disebut *token*. Pada data yang masih kotor, bahan-bahan makanan sebuah produk dituliskan dalam bentuk kalimat yang masing-masing bahan makanan dipisahkan dengan koma (,). Maka dari itu, perlu dilakukan tokenization agar bisa mendapatkan token yang berupa bahan makanan unik sehingga memungkinkan untuk melakukan pengolahan data selanjutnya.

3.1.4. Pengukuran kemiripan bahan makanan secara konseptual dan tekstual

Untuk mengurangi rendundansi pada data bahan makanan dari open food facts setelah dilakukan data pre-processing, maka perlu menghilangkan dua bahan makanan yang mirip menjadi satu bahan makanan terkait. Pada tugas akhir ini, pengukuran kemiripan kata akan dilakukan dengan beberapa metode yang berbeda. Metode tersebut dikelompokkan secara umum menjadi dua jenis, yaitu konseptual dan tekstual.

Pengukuran kemiripan kata secara konseptual mengukur kemiripan dua kata dalam hal makna/sinonim kata, terlepas dari kesamaan tiap karakter pada masing-masing kata. Dalam tugas akhir ini, metode Wordnet akan digunakan sebagai pengukuran kemiripan kata secara konseptual.

Pengukuran kemiripan kata secara tekstual mengukur kemiripan dua kata secara tekstual, dimana mengacu pada tiap karakter antar kata untuk mengukur kemiripannya. Dalam tugas akhir ini, metode fuzzy string matching digunakan sebagai pengukuran kemiripan secara tekstual.

Contoh kata bahan makanan yang diukut secara konseptual adalah ‘baking_soda’ dibandingkan dengan ‘sodium_bicarbonate’ merupakan bahan yang sama [28]. ‘salt’ dan ‘sodium_chloride’ merupakan dua bahan yang sama [29]. ‘kernel’ dan ‘seed’ merupakan istilah yang sama yaitu merepresentasikan biji. Atau ‘flesh’ dan ‘meat’ dalam suatu konteks tertentu merupakan hal yang sama. Untuk itu, penelitian ini menggunakan pengukuran Wordnet::similarity::path dengan metode Leacock dan Chodorow [8] seperti yang telah diuraikan pada 2.2.6.

Contoh perhitungan kemiripan dengan Wordnet antara 2 bahan makanan yang ada pada database halal yaitu young coconut flesh dan young coconut meat.

id ▲ 1	iName	iType	eNumber	created_at	updated_at
222202	young coconut drink	0	*NULL*	2018-06-16 00:00:00	2018-06-16 00:00:00
222203	young coconut flesh	0	*NULL*	2018-06-16 00:00:00	2018-06-16 00:00:00
222204	young coconut juice	0	*NULL*	2018-06-16 00:00:00	2018-06-16 00:00:00
222205	young coconut juicesugar	0	*NULL*	2018-06-16 00:00:00	2018-06-16 00:00:00
222206	young coconut kernel	0	*NULL*	2018-06-16 00:00:00	2018-06-16 00:00:00
222207	young coconut meat	0	*NULL*	2018-06-16 00:00:00	2018-06-16 00:00:00

Gambar 3.2 contoh data bahan makanan pada Halal Nutrition Food

Maka ketika dimasukkan ke dalam perhitungan wordnet similarity menggunakan metode Leacock-Chodorow (LCH) skor kemiripannya akan menjadi

LCH

	young /JJ	coconut /NN	flesh /NN
young/JJ	-	-	-
coconut/NN	-	3.6889	1.3863
meat/NN	-	2.5903	2.3026

```
[Description]
LCH relies on the length of the shortest path between two
synsets for their measure of similarity.
LCH(s1, s2) = -Math.log_e( LCS(s1, s2).length / ( 2 *
max_depth(pos) ) ).

[Parameters]
 - min score = 0.0
 - max score = Infinity
 - error score = -1.0
 - acceptable pos pairs = [['n', 'n'], ['v', 'v']]
 - use all senses = true
 - use root node = true
 - max depth N = 20
 - max depth V = 14
```

Gambar 3.3 hasil perhitungan LCH dengan http://ws4jdemo.appspot.com/

Selain pengukuran kemiripan bahan makanan secara konseptual, dilakukan juga pengukuran kemiripan secara tekstual yang yaitu menggunakan fuzzy string matching. Fuzzy string matching lebih berfokus pada mengukur perkiraan kesamaan antar karakter kata. Fuzzy string matching yang digunakan adalah levenshtein distance (2.2.8), jaro-winkler distance (2.2.9), dan jaccard distance (2.2.10).

3.1.5. Pengujian

Pengujian yang dimaksud adalah melakukan perbandingan metode yang paling efektif untuk mengukur kesamaan bahan makanan. Metode yang akan dibandingkan adalah: fuzzy string matching, Wordnet::similarity::path oleh Leacock-Chodorow dan gabungan antara keduanya.

Aspek-aspek yang akan dibandingkan antara lain, berapa banyak jumlah data yang terintegrasi, dan akurasinya. Hal ini dilihat dari data produk maupun data bahan makanan. Pengukuran akurasi akan dilakukan dengan supervisi manual. Supervisi manual yang dilakukan menggunakan metode sampling, sehingga tidak semua dataset disupervisi satu-persatu. Setelah data diintegrasikan, dilihat apakah replacement-replacement maupun penambahan produk yang dilakukan benar.

3.1.6. Integrasi Data Bahan Makanan Beserta Produknya ke Database Halal Nutrition Food

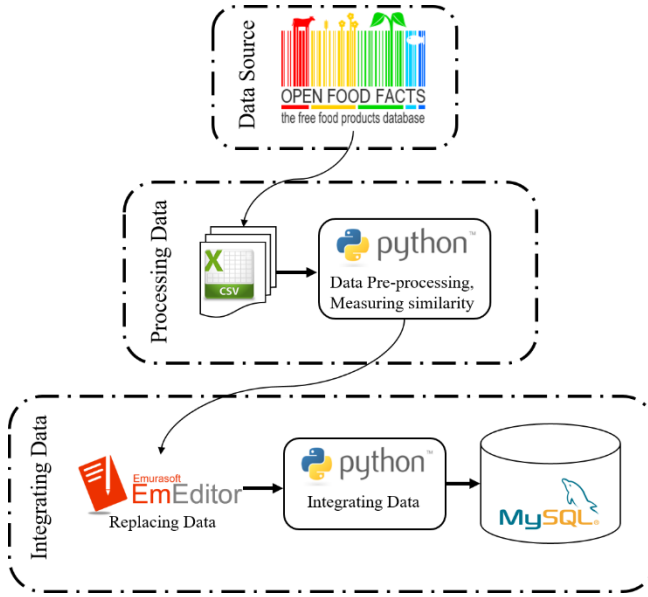
Pada tahap ini, data bahan makanan yang mencapai threshold kemiripan akan di-*replace*. Penggantian bahan makanan yang mirip dilakukan dengan salah satu metode paling optimal, dimana metode ini dipilih setelah melalui tahap pengujian.

Setelah bahan makanan yang mirip di replace dengan bahan makanan terkait, data bahan makanan beserta produknya diintegrasikan ke database halal nutrition food. Proses integrasi data dilakukan sedemikian rupa sehingga data yang dimasukkan tidak terjadi redundansi dengan data yang sudah ada.

BAB IV
PERANCANGAN

Bab ini berisi rancangan penelitian untuk pengerjaan tugas akhir. Proses yang terdapat pada bab ini antara lain pre-processing data, pengukuran kemiripan bahan makanan, integrasi data, dan pengujian.

4.1. Arsitektur sistem



Gambar 4.1 arsitektur sistem

4.1.1 Sumber Data

Sumber data diambil dari ditus web open food facts. Situs ini menyediakan open data tentang produk makanan yang bisa digunakan oleh siapapun, dan bisa ditambahkan oleh siapapun. Data yang didapatkan dari open food facts berupa file csv.

4.1.2 Pemrosesan Data

Data bahan makanan yang didapatkan dari situs open food facts kemudian diolah dengan menggunakan bahasa pemrograman python. Pertama bahan makanan melalui pre-processing untuk dibersihkan sedemikian rupa sehingga dapat dilakukan processing data selanjutnya Kemudian data bahan makanan diukur kemiripannya menggunakan bahasa pemrograman python dengan llibrary yang mendukung pengukuran kemiripan tersebut.

4.1.3 Integrasi Data

Setelah dilakukan pengukuran kemiripan menggunakan bahasa python, dihasilkan mapping penggantian bahan makanan yang mirp dengan bahan makanan terkait. Lalu bahan makanan yag mirip tersebut direplace menggunakan EmEditor. Setelah di replace, data diolah kembali di python untuk menyesuaikan data kepada struktur tabel yang ada pada database myssql. Lalu setelah data siap, data diintegrasikan ke sistem database MySQL.

4.2. Data Pre-processing

Data yang diunduh dari website open food facts berupa file csv. File csv tersebut memiliki 804.017 baris dan 175 kolom dengan ukuran file 2 giga pada awal diunduh.

Data yang didapatkan dari open food facts masih bersifat sangat kotor. Berikut cuplikan data pada file open food facts:

Tabel 4.1 Cuplikan data open food facts

code	product_name	brands	ingredients_text
1199	solène céréales poulet	crous	antioxydant : érythorbate de sodium, colorant ...
1663	crème dessert chocolat	ferme de la frémondière	lait entier, sucre, amidon de maïs, cacao, aga...
2264	baguette poitevin	crous resto	baguette poite vin pain baguette 50,6%: farine...
3827	suedois saumon	crous	paln suédois 42,6%: farine de blé, eau, farine...
4510	salade shaker taboulé	crous	taboulé 76,2%, légumes 12%, huile de colza, se...
4530	banana chips sweetened (whole)	NaN	bananas, vegetable oil (coconut oil, corn oil ...

Pada contoh data di atas, data bahan makanan (kolom ‘ingredients_text’) belum terpisah-pisah perbaris, masih ada karakter-karakter yang tidak diperlukan, serta masih banyak kata-kata yang bukan merupakan bahan makanan. Pun pada kolom tersebut, beberapa bahan makanan masih ada yang berbahasa Perancis, sedangkan pada tugas akhir ini, bahan makanan yang akan diukur kemiripan dan diintegrasikan ke dalam aplikasi halal nutrition food adalah yang berbahasa Inggris saja.

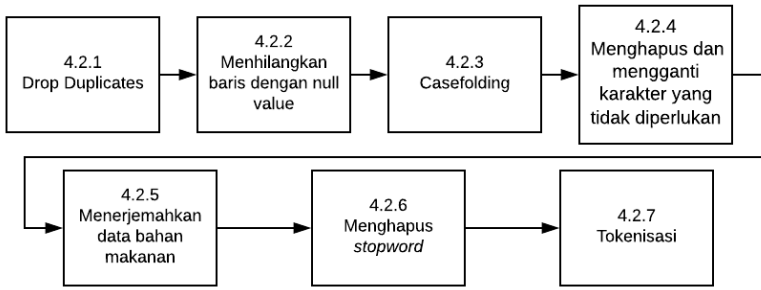
Contoh bentuk data yang diharapkan setelah dilakukan pre-processing adalah seperti berikut,

Tabel 4.2 Contoh data yang diharapkan setelah pre-processing

ingredient
sodium
dye
caramel
tomato
mayonnaise
rapeseed oil
water
egg yolks
vinegar
mustard

Maka dari itu perlu adanya beberapa langkah pre-processing data sehingga siap diproses untuk pengukuran kemiripan.

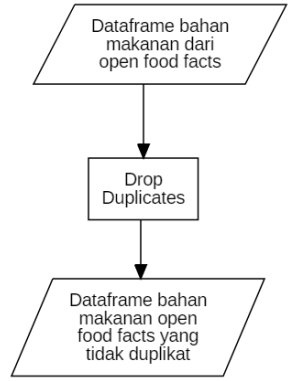
Pertama-tama, file csv dibaca dan dijadikan data frame menggunakan python dengan library pandas dengan menggunakan aplikasi berbasis web, jupyter notebook. Data frame yang dibentuk hanya terdiri dari 4 kolom yaitu 'code', 'product_name', 'brands', dan 'ingredients_text' dengan 804.017 baris. Data frame tersebut lalu dilakukan pre-processing kemudian ditulis kembali dalam file csv. Berikut alur pre-processing data:



Gambar 4.2 Alur data pre-processing

4.2.1. Drop Duplicates

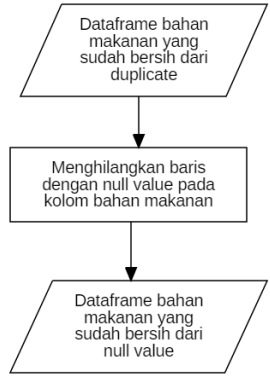
Data yang sudah dimuat ke dalam dataframe pertama-tama dibersihkan dengan membuang row yang identik dengan hanya menyisakan salah satunya. Proses ini dapat dengan mudah dilakukan nantinya dengan menggunakan library pandas.



Gambar 4.3 Alur drop duplicates

4.2.2. Menghilangkan baris dengan null value

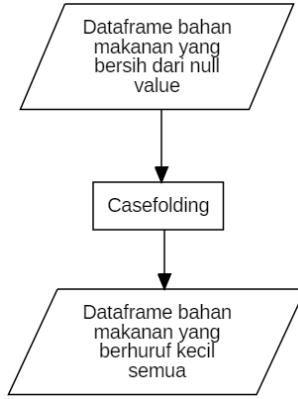
Selanjutnya, dari dataframe yang ada, akan diambil baris dengan nilai not null pada kolom ‘ingredients_text’nya. Karena inti dari tugas akhir ini adalah mengolah data bahan makanan, maka data produk makanan yang tidak terdapat keterangan ingredients pada barisnya tidak akan diproses lebih lanjut. Proses ini dapat dengan mudah dilakukan nantinya dengan menggunakan library pandas.



Gambar 4.4 Alur menghilangkan null value

4.2.3. Casefolding

Pada tahap ini, teks pada dataframe dirubah ke bentuk huruf kecil semua, agar ketika proses pengukuran kemiripan, data lebih akurat dan mudah diolah. Proses casefolding dilakukan menggunakan bantuan library python untuk merubah string menjadi lower case.



Gambar 4.5 Alur casefolding

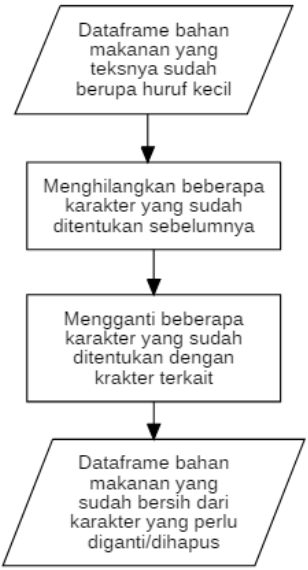
4.2.4. Menghapus dan mengganti karakter yang tidak diperlukan

Langkah selanjutnya yaitu menghilangkan dan mengganti karakter yag tidak diperlukan, berikut daftarnya:

Tabel 4.3 Karakter yang dihapus/diganti

Karakter yang dihapus		Karakter yang diganti dengan koma (,)	
1	[]	1	\d\d[,]\d[%]
2	{}	2	[%]\d\d[.]\d
3	()	3	\d\d[%]
4	.	4	\d\d mg
5	:	5	\d\d\d kcal
6	&	6	\d\d\d kj
7	·	7	\d\d\d ml
8	"	8	[\d]+[az]*
9	<	9	[\d]+[az]*+[\d]+
10	@	10	[\d]+[az]*+[\d]+[a-z]+
11	?		
12	=		
13	-		
14	,,		
15	“		
16	”		
17	*		
18	#		

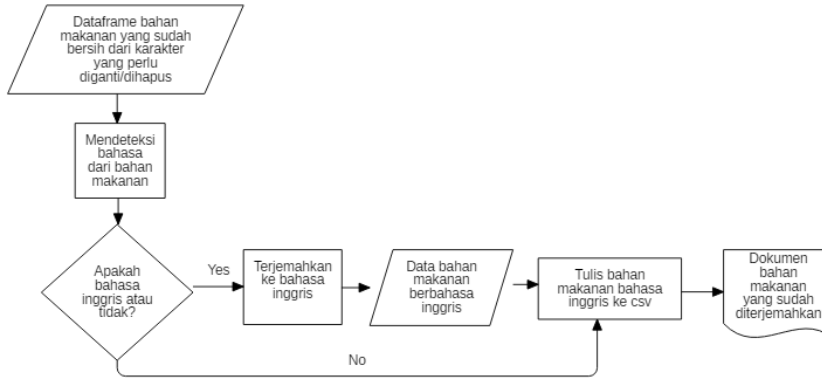
Pada proses ini terdapat dua tahap utama, aitu menghilangkan/menghapus karakter yang ada pada tabel di atas untuk dihapus, dan mengganti karakter yang ada pada tabel di atas dengan karakter terkait. Proses penghilangan dan penggantian karakter ini dapat dengan mudah dilakukan nantinya dengan library pandas.



Gambar 4.6 Alur menghapus/mengganti karakter

4.2.5. Menerjemahkan data bahan makanan

Beberapa bahan makanan yang ada didapati masih berbahasa perancis, maka perlu adanya proses penerjemahan. Proses penerjemahan menggunakan API google translate yaitu googletrans.



Gambar 4.7 Alur menerjemahkan bahan makanan

4.2.6. Menghapus *Stopword*

Pada tahap ini, daftar stopword dibuat terlebih dahulu. Kemudian daftar ini digunakan sebagai inputan pada proses stopword removal sehingga didapatkan data yang bersih dari stopword. Stopword yang digunakan pada proses ini berasal dari 2 sumber. Pertama, dari daftar stopword yang sudah dibuat sebelumnya, kedua dari library nltk yang menyediakan list stopword.

Gambar 4.8 Alur menghapus stopword

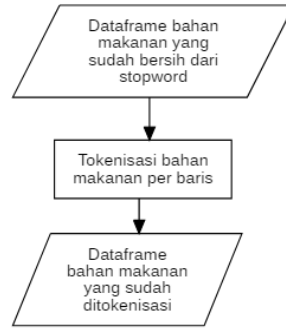
Berikut cuplikan stopword yang dibuat.

Tabel 4.4 Stopwords

No.	Stopwords
1	**Kandungan:** Ingredient, contain, including, traces of, may contain, made from, that contain, possible, presence of, not specified
2	**Peringatan:** to keep at, unopened, must be, to consume, preferably before, stored in, refrigerator, in a, non-metallic container, consume within, protected from, warm and dry, warning, after opening, store, and, preferably, consume, before, best before, shake, well, before opening, after opening, eat, quickly, keep dry, not suitable, children, keep between, away, direct, deciatatlon, store in, cool and dry place, once opened, container, avoiding, sharp edges, keep it, in the, freezer
3	**Informasi tambahan:** net weight, this is, regularly, checked, during, minimum, durability, period, properly, from organic farming, no ingredients indicated, from biological agriculture, percentage of , fat lower, collagen ratio, less than, made in, serving, suggestion, printed on top, produced by, see, distributed by, nutritional, in varying, proportions, dosage, weight, best, additional, information, vegetarians, no preservatives, unspecified, superior, quality, proud of, our simple, our family, recipe, for allergens, nutritional values, averages

4.2.7. Tokenisasi

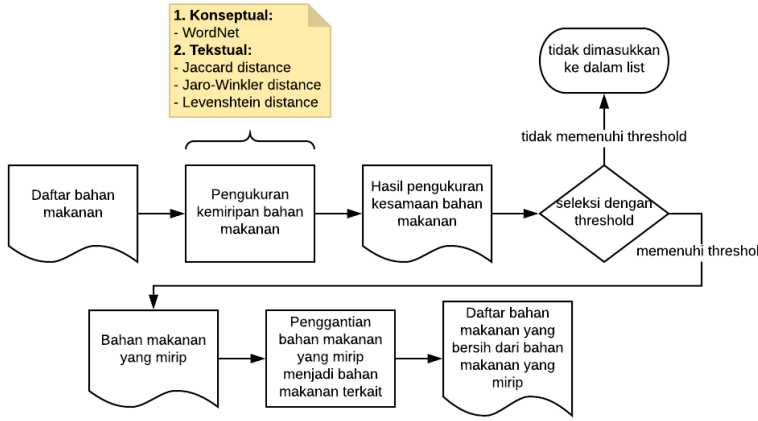
Tokenisasi dilakukan pada daftar bahan makanan sehingga dapat menghasilkan bentuk data yang diinginkan. Proses tokenisasi disini memisahkan kata per row dengan tanda pemisah koma (,).



Gambar 4.9 Alur tokenisasi

4.3. Pengukuran Kemiripan Bahan Makanan

Berikut alur pengukuran kemiripan bahan makanan:



Gambar 4.10 Alur pengukuran kemiripan

Setelah bahan data bahan makanan diakukan pre-processing, bahan makanan diukur kemiripannya menggunakan dua jenis metode pengukuran kemiripan, yaitu konseptual dan tekstual. Pengukuran konseptual menggunakan wordnet dengan metode Leacock Chodorow (LCH).

Pengukuran kemiripan secara tekstual menggunakan metode levenshtein distance, jaro-winkler distance, dan jaccard distance.
Selain itu juga akan dilakukan penggabungan dua metode konseptual-tekstual, yaitu wordnet & jaccard, wordnet & jaro-winkler, dan wordnet & levenshtein.
Data bahan makanan yang diukur kemiripannya yaitu data bahan makanan hasil preprocessing dari data open food facts, dan juga data bahan makanan dari database aplikasi halal nutrition food yang sudah ada.
Jenis pengukuran yang kemiripan yang akan dilakukan adalah sebagai berikut.

Pengukuran kemiripan secara konseptual:
K.1. Pengukuran kemiripan menggunakan wordnet LCH
Mengukur kemiripan bahan makananan menggunakan library yang disediakan oleh Wordnet dengan variasi pengukuran oleh Leacock-Chodorow. Pengukuran yang dilakukan menghasilkan nilai *similarity* atau tingkat kesamaan antar bahan makanan yang dibandingkan. Hasil pengukuran yang didapatkan berupa normalisasi (1%-100%), threshold yang digunakan adalah 80% atau 0.8.

Pengukuran kemiripan secara tekstual:
T.1. Pengukuran kemiripan menggunakan levenshtein distance
Mengukur kemiripan bahan makanan menggunakan library textdistance.levenshtein degan menggunakan metode normalized_similarity. Pengukuran yang dilakukan menghasilkan nilai *similarity* atau tingkat kesamaan antar bahan makanan yang dibandingkan. Hasil pengukuran yang didapatkan berupa normalisasi (1%-100%), threshold yang digunakan adalah 80% atau 0.8.
T.2. Pengukuran kemiripan menggunakan jaro-winkler distance
Mengukur kemiripan bahan makanan menggunakan library textdistance.jaro_winkler degan menggunakan metode

normalized_similarity. Pengukuran yang dilakukan menghasilkan nilai *similarity* atau tingkat kesamaan antar bahan makanan yang dibandingkan. Hasil pengukuran yang didapatkan berupa normalisasi (1%-100%), threshold yang digunakan adalah 80% atau 0.8.

T.3. Pengukuran kemiripan menggunakan jaccard distance

Mengukur kemiripan bahan makanan menggunakan library textdistance.jaccard degan menggunakan metode normalized_similarity. Pengukuran yang dilakukan menghasilkan nilai *similarity* atau tingkat kesamaan antar bahan makanan yang dibandingkan. Hasil pengukuran yang didapatkan berupa normalisasi (1%-100%), threshold yang digunakan adalah 80% atau 0.8.

Pengukuran kemiripan gabungan konseptual-tekstual:

G.1. Pengukuran kemiripan gabungan menggunakan wordnet & levenshtein distance

Mengkombinasikan hasil pengukuran kemiripan bahan makanan dengan metode wordnet LCH dan levenshtein distance.

G.2. Pengukuran kemiripan gabungan menggunakan wordnet & jaro-winkler distance

Mengkombinasikan hasil pengukuran kemiripan bahan makanan dengan metode wordnet LCH dan jaro-winkler distance.

G.3. Pengukuran kemiripan gabungan menggunakan wordnet & jaccard distance

Mengkombinasikan hasil pengukuran kemiripan bahan makanan dengan metode wordnet LCH dan jaccard distance.

Setelah didapatkan daftar data bahan makanan yang mirip, maka data bahan makanan yang ada sebelumnya akan diganti dengan data terkait. Sehingga, didapatkan data produk serta bahan makanannya yang sudah bersih dari data yang mirip.

4.4. Pengujian

4.4.1. Penghitungan akurasi berdasarkan pengujian manual

Pengujian dilakukan dengan cara supervisi manual pada data dengan menggunakan pengukuran sensitivitas dan spesifisitas dengan menggunakan sejumlah sampel. Pengujian dilakukan juga dengan mengacu pada referensi literatur mengenai bahan makanan yang berjudul "Dictionary of Food Ingredients" [30] untuk memeriksa kebenaran dari hasil pengukuran kemiripan.

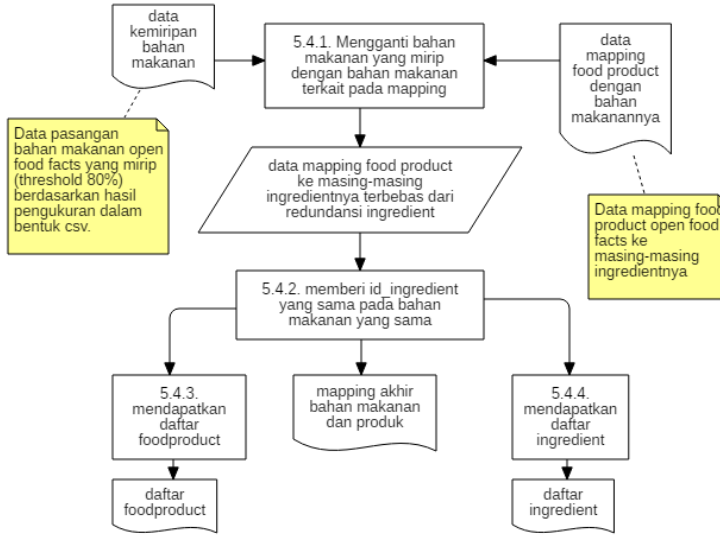
Pengujian menggunakan pengukuran precision dan recall dengan menghitung nilai *true positive, true negative, false positive,* dan *false negative* terhadap sampel yang diuji.

4.4.2. Membandingkan metode pengukuran kemiripan

Setelah pengujian dilakukan dan mendapatkan hasil akurasi, maka pengukuran kemiripan dengan keempat metode diatas dibandingkan sehingga terlihat mana metode pengukuran kemiripan yang memiliki precision dan recall terbaik.

4.5. Integrasi Data Bahan Makanan Beserta Produknya ke Database Halal Nutrition Food

Setelah dihasilkan data bahan makanan yang bersih, data akan diintegrasikan ke dalam basis data halal nutrition food. Sistem basis data halal nutrition food menggunakan MySQL. Berikut alur integrasi data yang akan dilakukan:



Gambar 4.11 alur integrasi data open food facts ke database halal

Data baru yang telah diolah merupakan data yang bersumber dari open food facts. Perlu dilakukan pemetaan terhadap data baru ke database yang telah ada pada aplikasi halal nutrition food. Data baru ini akan diintegrasikan ke dalam tiga tabel: ‘ingredients’, ‘foodproducts’, dan ‘foodproduct_ingredient’. Berikut struktur ketiga tabel tersebut:

Tabel 4.5 database tabel ingredients

Tabel: Ingredients				
No	**Kolom**	**Tipe**	**Keterangan**	**Mapping data baru**
1	Id	int(10)	Menyimpan id unik dari tiap ingredient.	melanjutkan dari id sebelumnya (222626)
2	iName	varchar(255)	Menyimpan nama bahan makanan	ingredient

3	iType	int(11)	Menandai aditive/non-aditive (0/1)/	-
4	eNumber	varchar(255)	Menyimpan kode aditif standar European Economic Community (EEC).	-
5	created_at	timestamp	Menyimpan waktu saat data ditambahkan	Auto generate
6	updated_at	timestamp	Menyimpan waktu saat data diupdate.	Auto generate

Tabel 4.6 database tabel foodproducts

Tabel: foodproducts				
No.	**Kolom**	**Tipe**	**Keterangan**	**Mapping data baru**
1.	Id	int(10)	Menyimpan id unik dari tiap produk makanan	melanjutkan dari id sebelumnya (393075)
2.	fCode	varchar(255)	Menyimpan nilai pada kolom code dari data baru hasil cleansing	code
3.	fName	varchar(255)	Menyimpan nama produk	Product_name
4.	fManufacture	varchar(255)	Menyimpan nama perusahaan produsen produk.	-
5.	fVerify	int(11)	Menyimpan status verifikasi data oleh admin	0
6.	fView	int(11)	Menyimpan jumlah berapa klai produk dilihat	-
7.	Weight	int(11)	nutrition facts	-
8.	Calories	int(11)	nutrition facts	-

Tabel: foodproducts				
No.	**Kolom**	**Tipe**	**Keterangan**	**Mapping data baru**
9.	totalFat	double(8.2)	nutrition facts	-
10.	saturatedFat	double(8.2)	nutrition facts	-
11.	transFat	double(8.2)	nutrition facts	-
12.	cholesterol	double(8.2)	nutrition facts	-
13.	sodium	double(8.2)	nutrition facts	-
14.	totalCarbohydrates	double(8.2)	nutrition facts	-
15.	dietaryFiber	double(8.2)	nutrition facts	-
16.	Sugar	double(8.2)	nutrition facts	-
17.	Protein	double(8.2)	nutrition facts	-
18.	vitaminA	int(11)	nutrition facts	-
19.	vitaminC	int(11)	nutrition facts	-
20.	Calcium	int(11)	nutrition facts	-
21.	iron	int(11)	nutrition facts	-
22.	user_id	int(10)	Menyimpan id user	-
23.	created_at	timestamp	Menyimpan waktu saat data ditambahkan	-
24.	updated_at	timestamp	Menyimpan waktu saat data diupdate.	-

Tabel 4.7 database tabel foodproduct_ingredient

Tabel: foodproduct_ingredient				
No.	**Kolom**	**Tipe**	**Isi**	**Mapping data baru**
1	foodproduct_id	int(10)	Menyimpan id yang mereferensi ke kolom id pada tabel foodproducts.	id yang mereferensi ke kolom id pada tabel foodproducts.
2	ingredient_id	int(10)	Menyimpan id yang mereferensi ke kolom id pada tabel ingredients.	id yang mereferensi ke kolom id pada tabel ingredients

Tabel: foodproduct_ingredient				
No.	**Kolom**	**Tipe**	**Isi**	**Mapping data baru**
3	created_at	timestamp	Menyimpan timestamp saat data ditambahkan	timestamp saat data ditambahkan
4	updated_at	timestamp	Menyimpan timestamp saat data diupdate.	timestamp saat data diupdate.

Berikut cuplikan data csv yang terdapat dalam proses integrasi data produk makanan dan bahan makanan ke database halal.

Tabel 4.8 data mapping food product dengan bahan makanannya

Food_code	ingredient
1199	potassium sorbate
1199	carotene
1199	aroma
1663	whole milk
1663	sugar
1663	maize starch
1663	cocoa
1663	agar agar
2264	baguette poite wine bread baguette
2264	wheat flour
2264	yeast
2264	gluten
2264	braised wheat flour
2264	yeast inactivated
2264	ascorbic acid

Tabel 4.9 data kemiripan bahan makanan

No.	Find	Replace
1	cellulose gums	cellulose gum
2	ascorbic acid e	ascorbic acid
3	l ascorbic acid	ascorbic acid
4	e ascorbic acid	ascorbic acid
5	baking powder '	baking powder
6	baking powder r	baking powder
7	altantic salmon	atlantic salmon
8	veqetable oils	vegetable oil
9	vegetable oils	vegetable oil
10	vegetables oils	vegetable oil

BAB V
IMPLEMENTASI

Pada bab ini akan dijelaskan mengenai proses pelaksanaan tugas akhir berdasarkan perancangan yang telah dibahas pada bab sebelumnya.

5.1. Lingkungan Penelitian

Dalam proses implementasi pada pengerjaan tugas akhir ini memerlukan beberapa perangkat keras dan perangkat lunak untuk menunjang keberhasilan pengerjaannya. Perangkat keras dan perangkat lunak tersebut akan dijelaskan pada tabel berikut.

Tabel 5.1 Lingkungan perangkat keras

Perangkat keras	
Laptop	
Jumlah	20 buah
Merek	LG
CPU	Intel Core i5-2400
RAM	4 GB
Disk	160 GB HDD

Tabel 5.2 Lingkungan perangkat lunak

Perangkat lunak	
Jenis	Perangkat
Operation System	Windows 10 Education 64-bit
Python IDE & editor	Jupyter Notebook
Database Administrator	phpMyAdmin
Text Editor	EmEditor

Tabel 5.3 Library python yang digunakan

Library Python	
Library	Kegunaan
pandas	Banyak digunakan dalam keseluruhan proses pengerjaan, terutama pada pre-processing data.
textdistance	Menyediakan library untuk semua jenis pengukuran kemiripan pada penelitian ini.
googletrans	Menerjemahkan bahan makanan pada pre-processing data.
langdetect	Mendeteksi bahasa dari kata/kalimat. Membantu saat proses penerjemahan
nltk.corpus stopword	Menjadi salah satu sumber stopword
glob	Membantu mencari path name dari banyak file dengan pola tertentu, berfungsi pada proses penggabungan banyak file.

5.2. Data Pre-processing

Data masukan pada tahap pre-processing didapatkan dengan mengeksport dan mengunduh data pada situs open food facts. Data yang diunduh berupa data csv dengan separator koma (,). Kolom yang digunakan yaitu kolom 'code', 'product_name', 'brands', dan 'ingredients_text'.

Data makanan dari open food facts di load menjadi DataFrame ke IDE python jupyter notebook menggunakan library pandas agar mudah untuk pengolahan selanjutnya. Dataframe ini memiliki 804.017 baris. Dataframe inilah yang kemudian dijadikan inputan pre-processing. Berikut kodenya.

```
import pandas as pd
import glob

df = pd.read_csv("E:\@Final Project\!NEXT STAGE\data\en-
openfoodfacts.csv", sep="\t", skip_blank_lines=False,
skiprows=0, usecols=['code', 'product_name', 'brands'
, 'ingredients_text'])
df.info()
```

Output dari baris kode di atas adalah:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 804017 entries, 0 to 804016
Data columns (total 4 columns):
code                 804017 non-null object
product_name         760725 non-null object
brands               512283 non-null object
ingredients_text     416506 non-null object
dtypes: object(4)
memory usage: 24.5+ MB
```

Output di atas menunjukkan informasi bahwa dataframe df memiliki 804.017 baris dan 4 kolom, dengan rincian di bawahnya menunjukkan jumlah baris yang tidak null per kolomnya.
Berikut implementasi tahapan pre-processing data yang dilakukan pada data open food facts:

5.2.1. Drop duplicates

Pada tahap ini, baris yang terdeteksi duplikat di hapus salah satunya. Proses menghapusan data duplikat ini menggunakan salah satu function dari library pandas yang bisa diimplementasikan ke dataframe pandas, yaitu drop_duplicates(), kemudian untuk mengecek jumlah baris pada dataframe menggunakan atribut '.shape'. Berikut kodenya

```
#drop duplicates
df.drop_duplicates(inplace=True)
df.shape
```

```
out: (803903, 4)
```

Output dari kode tersebut menunjukkan baris dataframe setelah dihapus data duplikatnya sebanyak 803.903 baris dengan jumlah kolom yang tetap, yaitu 4 . Dengan kata lain, baris yang terkurangi pada tahap ini adalah sebanyak 114 baris (didapat dari banyak baris sebelumnya yaitu 804.017 dikurangi 803.903).

5.2.2. Menghilangkan null value

Pada tahap ini, baris yang didapati kosong pada kolom ‘ingredients’ akan dihapus. Proses ini menggunakan suatu function dari library pandas yaitu .notnull(). Function tersebut diaplikasikan ke kolom ‘ingredients_text’ pada dataframe df. Kemudian untuk mengecek jumlah baris pada dataframe, digunakan atribut ‘.shape’. Berikut kodenya.

```
1. df = df[df.ingredients_text.notnull()]
2. df.shape

out: (416479, 4)
```

Hasil outputan dari kode tersebut dapat diketahui bahwa baris yang tersisa setelah pre-processing pada tahap ini adalah 416.479 baris. Dengan kata lain, baris yang tereliminasi pada tahap ini sejumlah 387.424 baris.

5.2.3. Casefolding

Pada tahap ini, tiap teks pada dataframe diubah menjadi huruf kecil. Proses ini menggunakan fungsi yang disedikan oleh pandas yaitu .str.lower(). Fungsi ini hanya bisa diberlakukan pada series (istilah library pandas untuk data dengan satu kolom). Maka yang dilakukan untuk menggunakan fungsi tersebut adalah dengan mengimplementasikannya satu persatu ke setiap kolom yang ada. Berikut kodenya.

```
2.  df['product_name']=df['product_name'].str.lower()
3.  df['brands']=df['brands'].str.lower()
```

```
4.  df['ingredients_text']=df['ingredients_text'].str.lo
    wer()
```

Setelah baris kode di atas di jalankan, dataframe disimpan ke dalam csv agar tedokumentasikan.

Berikut cuplikan data setelah proses drop duplicates, menghilangkan null value, dan casefolding.

Tabel 5.4 Contoh data setelah casefolding

code	ingredients_text
417011002	sugar, cocoa butter, cream powder (16%), corn (11%), cocoa mass, lactose, emulsifier: lecithin (soy), salt barley malt, natural flavor.
417012009	sugar, cocoa butter, raisins, cocoa mass, whole milk powder, hazelnuts, lactose, skimmed milk powder, butterfat, jamaican rum, emulsifier: lecithin (soy), natural flavor.
417296003	sugar, cocoa butter, cocoa mass, whole milk powder, almonds (7.7%), lactose, skim milk powder, butterfat, emulsifier: lecithin (soy), natural flavor.
433070274	sugar, almonds, water, sorbitol, invert sugar, food coloring, turmeric, carmine, fd&c blue @1: egg white
433821494	only the best: un-bleached flour, organic steel cut whole oats, organic california raisins, creamery butter, whole egg, un-refined sugar, madagascar vanilla.

5.2.4. Menghapus dan mengganti karakter yang tidak diperlukan

Sebelum masuk kedalam proses penghapusan dan penggantian karakter, data di load terlebih dahulu dari file csv yang telah dihasilkan pada tahapan sebelumnya. Load data pada kali ini hanya diambil kolom yang diperlukan saja, yaitu kolom ‘code’ dan ‘ingredients_text’.

Pada tahap ini, inputan data setelah berubah menjadi huruf kecil semua pada proses sebelumnya, diproses kembali dengan

menghapus beberapa karakter yang perlu dihilangkan, dan subtitusi karakter yang perlu diganti dengan isian yang seharusnya. Daftar karakter yang akan diganti maupun dihilangkan dapat dilihat pada tabel Tabel 4.3.

Berikut kode program untuk menghapus dan mengganti karakter yang tidak diperlukan.

```
# menghilangkan character(s) dengan regular expression
df_ing['ingredients_text'] = df_ing['ingredients_text'].replace(to_replace=[r'\d\d[,]\d[%]', r'\d\d[%]', r'[%]\d\d[.]\d', r'\d\d[,]\d[%]', r'\d\d mg', r'\d\d g', r'\d\d\d kcal', r'\d\d\d kj', r'\d\d\d ml', r'[\d]+[az]*', r'[\d]+[az]+[\d]+'], value='', regex=True)

# mengganti character(s)
dengan koma, dan menghilangkan character(s)
def change_string(x):
    return x.replace('(', ',').replace(')',',').replace('[',',').replace(']',',').replace('{',',').replace('}',',').replace(':',',').replace('&',',').replace('"'"'",',').replace('<',',').replace('@',',').replace('?',',').replace('=',',').replace('-',',').replace(',,',',').replace('"','').replace('_',',').replace('*','').replace('#','').replace('%','')

df_ing['ingredients_text'] = df_ing['ingredients_text'].map(lambda x: change_string(x))
```

Pertama, data pada kolom ‘ingredients_text’ beberapa karakter tertentu dihilangkan (contoh: ‘(’) dengan cara di replace dengan blank (‘’).

Lalu selanjutnya didefinisikan fungsi, dimana pada python, untuk membuat fungsi baru ditandai dengan ‘def’. Fungsi change_string() berisi metode untuk me-replace beberapa karakter (contoh: ‘#’) dengan karakter lainnya (contoh: ‘,’).

Kemudian barulah fungsi change_string() diimplementasikan pada kolom ‘ingredients_text’ milik dataframe ‘df_ing’.

Berikut cuplikan data setelah melalui proses penghapusan dan penggantian karakter.

Tabel 5.5 Contoh data setelah penggantian karakter

code	ingredients_text
417011002	sugar, cocoa butter, cream powder , corn , cocoa mass, lactose, emulsifier, lecithin ,soy, salt barley malt, natural flavor.
417012009	sugar, cocoa butter, raisins, cocoa mass, whole milk powder, hazelnuts, lactose, skimmed milk powder, butterfat, jamaican rum, emulsifier, lecithin ,soy, natural flavor.
417296003	sugar, cocoa butter, cocoa mass, whole milk powder, almonds ,., lactose, skim milk powder, butterfat, emulsifier, lecithin ,soy, natural flavor.
433070274	sugar, almonds, water, sorbitol, invert sugar, food coloring, turmeric, carmine, fd,c blue , egg white
433821494	only the best, un,bleached flour, organic steel cut whole oats, organic california raisins, creamery butter, whole egg, un,refined sugar, madagascar vanilla.

5.2.5. Penerjemahan bahan makanan

Pada tahap ini, data bahan makanan yang sebelumnya telah dibersihkan dari beberapa karatkter tertentu di load ke dalam dataframe untuk kemudian dilakukan proses penerjemahan.

Data frame yang telah diload tersebut kemudian tidak langsung kesemuanya dilakukan proses penerjemahan menggunakan library google translate, melainkan dipotong-potong menjadi sub dataframe.

Pertama, library yang dibutuhkan diload ke dalam python .

```
1. from googletrans import Translator  # Import Transla
   tor module from googletrans package
2. translator = Translator() # Create object of Transla
   tor.
3. from langdetect import detect
4. proxy = {
```

```
        'http': 'http://username:password@1.1.1.1:12
    34',
        'https': 'http://username:password@1.1.1.1:1
    234',
    }
```

Kemudian, data berupa file csv hasil dari proses sebelumnya diload ke dalam dataframe python.

```
# Loading Data: off_replacedchar.csv

df_replaced = pd.read_csv("E:\@Final Project\!NEXT ST
AGE\data_preprocessing\off_replacedchar.csv", encodin
g="ISO-8859-
1", delimiter=',', sep="\t", skip_blank_lines=False,
usecols=['code', 'ingredients_text'], dtype={'code':s
tr})
```

Setelah data sudah diload ke dalam dataframe, per barisnya pada dataframe dilakukan deteksi bahasa baris tersebut. Jika bahasa terdeteksi inggris, maka baris tersebut langsung dituliskan ke dalam file csv outputan. Jika tidak terdeteksi bahasa inggris, maka bahan makanan pada baris tersebut di terjemahkan ke dalam bahasa inggris, baru kemudian data hasil terjemahan tersebut dituliskan menjadi baris baru pada file csv outputan.

```
  #Translating1

  with open('E:\@Final Project\!NEXT STAGE\data_prepro
  cessing\off_trans1.csv', 'w', encoding='utf-
  8', newline='\n', ) as tulis:
      for index, row in df_replaced.iterrows():
          translator = Translator()
          try:
              tr = translator.translate(row['ingredien
  ts_text'], dest='en')
          except Exception as e:
              print(e)
              continue
          tulis.write(str(row['code'])+',"'+tr.text+'"
  \n')
```

```
12.         print(str(row['code'])+',"'+tr.text+'" \n')
13.
14. # ned to fix : separator using #
```

Menggunakan library googletrans untuk menerjemahkan kata pada python memiliki limitasi. Maka dari itu, proses penerjemahan dilakukan secara bertahap. Ketika mencapai limit, maka proses penerjemahan dilanjutkan dengan jarak beberapa waktu. Sehingga, baris kode untuk menerjemahkan ada berulang kali, disesuaikan dengan baris terakhir yang sempat diproses.

Untuk meminimalisir terjadinya kesalahan, tiap ulangan baris kode untuk proses penerjemahan hasil file outputnya dipisah dengan file output sebelumnya.

Setelah proses penerjemahan, kemudian file outputan dari hasil proses tersebut dilakukan penggabungan. Berikut baris kodenya.

```
1.  from glob import glob
2.  filenames = glob('E:\@Final Project\!NEXT STAGE\data
    _preprocessing\off_trans*.csv')
3.  df = pd.read_csv('E:\@Final Project\!NEXT STAGE\data
    _preprocessing\off_1trans.csv', quotechar='"'"', er
    ror_bad_lines=False, index_col=None, encoding="ISO-
    8859-
    1", delimiter=',', sep="\t", skip_blank_lines=False,
    header=None, dtype={'code':str}, engine='python').dr
    opna(axis=1)
4.
5.  for f in filenames:
6.      df_temp = pd.read_csv(f, quotechar='"', error_ba
    d_lines=False, index_col=None, encoding="ISO-8859-
    1", delimiter=',', sep="\t", skip_blank_lines=False,
    header=None, dtype={'code':str}, engine='python').dr
    opna(axis=1)
7.      print(f)
8.      df = pd.concat([df, df_temp], ignore_index=True)
```

Kemudian setelah semua data hasil terjemahan diletakkan dalam satu dataframe, dataframe dituliskan ke dalam csv. Sebelum ditulis ke dalam bentuk csv, dataframe pada kolom bahan makanan kemudian dilakukan proses lowering case. Hal ini dikarenakan tidak menutup kemungkinan bahwa hasil dari proses penerjemahan memiliki huruf kapital. Berikut baris kodenya.

```
1. # write dataframe to csv
2. df[1] = df[1].str.lower() # lowering case
3. df.to_csv("trans_all.csv")
4. df
```

5.2.6. Menghapus *stopword*

Pada proses ini, data bahan makanan yang terdeteksi mengandung stopword dihilangkan. Sumber dari stopwordnya sendiri ada dua yang digunakan pada proses ini yaitu library nltk dan dari stopword list yang dibuat sendiri oleh tugas akhir acuan sebelumnya. Ada beberapa langkah yang dilakukan pada proses menghapus stopword ini.

Berikut baris kodenya.

a. Meload list stopword
   Pertama, list stopword diload ke dalam variabel list. Stopword ini sendiri terdiri dari dua macam, yaitu stopword yang sudah disiapkan sebelumnya (pada tabel Tabel 4.4) dalam file csv, dan yang kedua adalah stopword yang diambil dari library nltk.

```
1.  # load the stopwordlists
2.  from nltk.corpus import stopwords
3.  stop1 = stopwords.words('english')
4.
5.  stop2 = pd.read_csv("E:\@Final Project\!NEXT STAG
    E\data_preprocessing\stopword\stopwords.tsv", hea
    der=None)
6.  stop2 = stop2.values.tolist()
```

b. Meload data bahan makanan
Setelah stopword list diload, maka data bahan makanan yang berbahasa inggris di load ke dalam dataframe.

```
1. #loadthedata
2. df_removedstop = pd.read_csv("trans_all.csv", del
   imiter=',', sep="\n", index_col=0)
3. df_removedstop.columns = ['code', 'ingredients_te
   xt']
```

c. Memisah kata bahan makanan menjadi list dan sekaligus menghapus stopword.
Agar masing-masing kata pada kolom bahan makanan bisa dibandingkan dengan masing-masing stopword pada stopword list, maka hal selanjutnya yang harus dilakukan adalah memisah kata pada kolom bahan makanan menjadi list bahan makanan.
Kemudian setelah itu dilakukan penghapusan stopword pada kolom bahan makanan. Proses penghilangan stopwrod itu sendiri dengan cara membandingkan kata perkata yang sudah ditokenisasi dengan list stopword yang ada, lalu jika tidak ada di dalam list stopword, maka kata tersebut disimpan. Untuk mengetahui perbedaan kata sebelum dan sesudah proses penghapusan ini, maka length dari list bahan makanan pada kolom ‘ingredients_text’ tiap row nya dijumlah dan ditampilkan ke output, baik itu sebelum dan juga sesudah proses penghapusan.

```
1. import datetime
2. print("start: ")
3. print(datetime.datetime.now())
4. print("\n")
5.
6. # split ingredients into list of word
7. df_removedstop['ingredients_text'] = df_removedst
   op['ingredients_text'].str.split()
8.
9. # count words before stopword removal
```

```
df_removedstop['ing_len'] = df_removedstop['ingredients_text'].apply(len)
print("len string sebelum stopword di remove:")
print("{:,}".format(df_removedstop['ing_len'].sum()))
print("\n")

# removing stopword
df_removedstop['ingredients_rem'] = df_removedstop['ingredients_text'].apply(lambda x: [item for item in x if (item not in stop1 and item not in stop2)])

# count words after stopword removal
df_removedstop['ing_len'] = df_removedstop['ingredients_rem'].apply(len)
print("len string setelah stopword di remove:")
print("{:,}".format(df_removedstop['ing_len'].sum()))
print("\n")

print("finish: ")
print(datetime.datetime.now())

[Out:]

start:
2019-06-11 10:24:51.488883


len string sebelum stopword di remove:
6,616,718


len string setelah stopword di remove:
6,259,542


finish:
2019-06-11 10:30:22.007030
```

d. Menggabungkan kembali kata bahan makanan yang sudah bersih dari stopword

Setelah semua bahan makanan non-stopword disimpan pada barisnya masing-masing, maka list bahan makanan yang terpisah pisah tiap rownya di gabungkan kembali menjadi text string seperti pada mulanya.

```
# rejoin words in df_removedstop

def rejoin_words(row):
    my_list = row['ingredients_rem']
    joined_words = ( " ".join(my_list))
    return joined_words

df_removedstop['ingredients_rem'] = df_removedstop.apply(rejoin_words, axis=1)
df_removedstop.drop(['ing_len','ingredients_text'],axis=1, inplace=True) # dropping unnecessary column
df_removedstop.columns = ['code', 'ingredients_text'] # rename the column
```

e. Menuliskan dataframe yang baru ke dalam csv

Setelah dataframe bahan makanan bersih dari stopword, maka dataframe dibersihkan dari row yang pada kolom ‘ingredients_text’nya berupa null value, lalu menuliskan dataframe ke dalam berntuk csv.

```
df_removedstop = df_removedstop[pd.notnull(df_removedstop['ingredients_text'])]
# write to csv
df_removedstop.to_csv("E:\@Final Project\!NEXT STAGE\data_preprocessing\stopword_removed2.csv")
```

5.2.7. Tokenisasi

Pada tahap ini, data yang ada dari proses sebelumnya diload ke dalam dataframe. Kemudian dibuatlah fungsi custom_tokenize() untuk melakukan tokenisasi pada tiap baris dataframe. Fungsi tersebut kemudian diimplementasikan ke dalam kolom bahan makanan pada dataframe.

Untuk menyaring baris-baris yang tidak memiliki makna pada kolom bahan makanannya, maka pada fungsi custom_tokenize() tidak hanya melakukan tokenisasi, tetapi juga dilakukan seleksi untuk memastikan bahwa baris terkait memiliki konten pada kolom bahan makanannya. Hasil dari filtering ini dituliskan ke dalam file csv baru. Berikut baris kodenya.

```
8.  # load the data
9.  df_tokenized = pd.read_csv("E:\@Final Project\!NEXT
    STAGE\data_preprocessing\stopword_removed.csv", deli
    miter=',', sep="\t", index_col=0)
10. df_tokenized
11.
12. import nltk
13.
14. #tokenizing
15.
16. def custom_tokenize(text):
17.     if not text:
18.         print('The text to be tokenized is a None ty
    pe. Defaulting to blank string.')
19.         text = ''
20.     return nltk.word_tokenize(text)
21. df_tokenized['tokenized'] = df_tokenized['ingredient
    s_text'].apply(custom_tokenize)
22.
23.
24. # write each tokenized word per row
25. err=0
26.
27. with open('tokenized.csv', 'w', encoding='utf-
    8', newline='\n', ) as tulis:
28.     for index, row in df_tokenized.iterrows():
29.         try:
```

```
            for token in row['tokenized']:
                if(len(token)<=1 or token=="'s"):
                    continue
                else:
                    tulis.write(str(row['code']).low
er()+',"'+token+'"\n')
        except:
            err+=1
            print('error '+str(row['code'])+'"\n')
            continue
print(err)
```

Setelah itu, sebelum masuk ke proses pengukuran kemiripan, data bahan makanan diseleksi kembali dengan menghilangkan data yang duplikat mengacu pada kolom bahan makanan.

```
df_tokenized_nodup = pd.read_csv("tokenized.csv", del
imiter=',', sep="\n", header=None)
df_tokenized_nodup.columns = ['code', 'ingredient']
df_tokenized_nodup = df_tokenized_nodup.drop_duplicat
es(subset='ingredient')
df_tokenized_nodup.dropna(subset=['ingredient'], inpl
ace=True)
df_tokenized_nodup = df_tokenized_nodup[(type(df_toke
nized_nodup.ingredient) == object) == True]
df_tokenized_nodup.to_csv("tokenized_nodup3.csv", ind
ex=False)
df_tokenized_nodup
```

Total hasil dari tokenisasi bahan makanan setelah dihilangkan dari baris-baris duplikat adalah sebanyak 103.100 baris bahan makanan.

5.3. Pengukuran Kemiripan Bahan Makanan

Pengukuran kemiripan bahan makanan dilakukan dengan empat metode utama, yaitu dengan menggunakan Jaccard distance, JaroWinkler distance, Levenshtein distance, dan Wordnet Leacock-Chodorow.

Threshold yang diberlakukan pada pengukuran kemiripan ini adalah 80%. Agar pengukuran kemiripan dapat diukur hasilnya

terhadap threshold, maka dilakukan normalisasi terhadap hasil pengukuran kemiripan. Pada metode pengukuran jaccard distance, JaroWinkler, dan Levenshtein distance, menormalisasi hasil kemiripiannya bisa langsung dengan salah satu fitur pada library textdistance, yaitu normalized_similarity().

Sebelum masuk ke pengukuran kemiripan, data bahan makanan yang tiap katanya sudah dipisah per baris pertama-tama dilakukan seleksi dengan mengecek apakah kata tersebut bertipe string/tidak. Karena bahan makanan yang bukan merupakan string tidak bisa diukur kemiripannya menggunakan library pengukuran kemiripan terkait. Berikut baris kode untuk menyeleksi daftar bahan makanan.

```
# filtering hanya menggunakan apakah type string/tidak

def filtering_str(in_f, out_f, out_f_dirty, size):
    print(datetime.datetime.now())
    reader = pd.read_csv(in_f, delimiter=',', sep="\n", chunksize=size)
    ch =0
    for chunk in reader:
        df_filtered = pd.DataFrame({'code': [],'ingredient': [],})
        df_dirty = pd.DataFrame({'code': [],'ingredient': [],})
            for index, row in chunk.iterrows():
                if type(row['ingredient']) is str: # memfilter apakah word berupa string
                    df_filtered = df_filtered.append({'code': row['code'],'ingredient': row['ingredient'],}, ignore_index=True)
                else:
                    df_dirty = df_dirty.append({'code': row['code'],'ingredient': row['ingredient'],}, ignore_index=True)
            df_filtered.to_csv(out_f, index=False, header=False, mode='a')
            df_dirty.to_csv(out_f_dirty, index=False, header=False, mode='a')
            print(datetime.datetime.now())
            ch+=1
```

```
19.             print(ch)
20.
21.     filtering_str('tokenized_nodup.csv', 'inputan_c
   lean_fuzzystr.csv', 'inputan_dirty_fuzzystr.csv', 100
   0)
```

Pengukuran kemiripan bahan makanan dengan metode fuzzy string matching (jaccard distance, jaro-winkler distance, dan levenshtein distance) dilakukan dengan menggunakan library python yaitu textdistance. Pertama, semua library yang dibutuhkan pada proses ini dipanggil.

Kemudian dibuat sebuah fungsi untuk mengukur kemiripan bahan makanan. Inputan dari fungsi ini adalah path file inputan, path file outputan, ukuran chuksize. Data bahan makanan yang akan dikomparasi adalah sejumlah 271,823 bahan makanan.

Chunksize merupakan suatu atribut pada library pandas untuk membaca/mengolah suatu file csv dengan limitasi ukuran baris yang ditentukan oleh banyak chunksize. Hal ini bertujuan untuk mengoptimasi pengolahan data dengan tidak menyimpan terlalu banyak data pada memori pemrosesan.

Dalam iterasi pada setiap chunksizenya, berisi iterasi untuk membandingkan tiap satu kata dengan semua kata yang ada pada list itu sendiri.

Berikut baris kode dari ketiga metode fuzzy string matching.

5.3.1.Jaccard Distance

```
1.  import pandas as pd
2.  import textdistance
3.  import datetime
4.  import os
5.  from itertools import islice
6.
7.  # string similarity measure jaccard
8.
```

```
9.  def jaccard_norm(in_f, out_f, size, start, stop):
10.     print(datetime.datetime.now()) # print waktu sek
    arang
11.     df_tokenized = pd.read_csv(in_f, delimiter=',',
    sep="\n") # read inputan data
12.
13.     # some stuff to know the progress
14.     ch = 0
15.     df_dummy = pd.DataFrame({'jaccard': [1]})
16.     df_dummy.to_csv("jaccardfix"+str(start)+"-
    "+str(stop-1)+"_"+str(ch)+".txt")
17.
18.     df_slice = df_tokenized.iloc[start:stop , :] # m
    en slice data (berbeda tiap kernel)
19.     nameslice = "slice "+str(start)+"-"+str(stop-
    1)+".csv" # nama untuk slice data yang akan dijadika
    n csv
20.     df_slice.to_csv(nameslice, header=['code','ingre
    dient']) # menjadikan slice data menjadi csv
21.
22.     reader = pd.read_csv(nameslice, delimiter=',', s
    ep="\n", chunksize=size, index_col=0) # meload slice
    data, dengan CHUNKSIZE
23.
24.     for chunk in reader:
25.         il=0
26.         df_lv = pd.DataFrame({'ingredient1': [],'ing
    redient2': [], 'jaccard': []})
27.         for index, row in chunk.iterrows():
28.             text1 = row['ingredient']
29.             for index2, row2 in islice(df_tokenized.
    iterrows(), index+1, None):
30.                 text2 = row2['ingredient']
31.                 sim = textdistance.jaccard.normalize
    d_similarity(text1.lower(),text2.lower()) # lowercas
    ing
32.                 il+=1
33.                 if sim>=0.8:
34.                     df_lv = df_lv.append({'ingredien
    t1': row['ingredient'],'ingredient2': row2['ingredie
    nt'], 'jaccard': sim}, ignore_index=True)
35.         df_lv.to_csv(out_f, index=False, header=Fals
    e, mode='a')
36.         renamebefore = "jaccardfix"+str(start)+"-
    "+str(stop-1)+"_"+str(ch)+".txt"
```

```
        ch+=1
        renameafter = "jaccardfix"+str(start)+"-
"+str(stop-1)+"_"+str(ch)+".txt"
        os.rename(renamebefore, renameafter)
    print(datetime.datetime.now())

jaccard_norm('tokenized_filtered.csv','jaccard_norm.
csv',100)
```

5.3.2.JaroWinkler Distance

```
import pandas as pd
import textdistance
import datetime
import os
from itertools import islice

# string similarity measure jarowinkler

def jarowinkler_norm(in_f, out_f, size):
    print(datetime.datetime.now())
    reader = pd.read_csv(in_f, delimiter=',', sep="\
n", chunksize=size)
    df_tokenized = pd.read_csv(in_f, delimiter=',',
sep="\n")
    ch = 0
    for chunk in reader:
        il=0
        df_lv = pd.DataFrame({'ingredient1': [],'ing
redient2': [], 'jaccard': []})
        for index, row in chunk.iterrows():
            text1 = row['ingredient']
            for index2, row2 in islice(df_tokenized.
iterrows(), index+1, None):
                text2 = row2['ingredient']
                sim = textdistance.jaro_winkler.norm
alized_similarity(text1.lower(),text2.lower()) # low
ercasing
                il+=1
                if sim>=0.8:
```

```
                    df_lv = df_lv.append({'ingredient1': row['ingredient'],'ingredient2': row2['ingredient'], 'jaccard': sim}, ignore_index=True)
        df_lv.to_csv(out_f, index=False, header=False, mode='a')
        renamebefore = "jarowinklerfix"+str(ch)+".txt"
        ch+=1
        renameafter = "jarowinklerfix"+str(ch)+".txt"
        os.rename(renamebefore, renameafter)
#         print(ch)
#         print(datetime.datetime.now())
```

5.3.3. Levenshtein Distance

```
import pandas as pd
import textdistance
import datetime
import os
from itertools import islice

# string similarity measure levenshtein

def levenshtein_norm(in_f, out_f, size):
    print(datetime.datetime.now())
    reader = pd.read_csv(in_f, delimiter=',', sep="\n", chunksize=size)
    df_tokenized = pd.read_csv(in_f, delimiter=',', sep="\n")
    ch = 0
    for chunk in reader:
        il=0
        df_lv = pd.DataFrame({'ingredient1': [],'ingredient2': [], 'levenshtein': []})
        for index, row in chunk.iterrows():
            text1 = row['ingredient']
            for index2, row2 in islice(df_tokenized.iterrows(), index+1, None):
                text2 = row2['ingredient']
                sim = textdistance.levenshtein.normalized_similarity(text1.lower(),text2.lower()) # lowercasing
                il+=1
                if sim>=0.8:
```

```
                    df_lv = df_lv.append({'ingredien
t1': row['ingredient'],'ingredient2': row2['ingredie
nt'], 'levenshtein': sim}, ignore_index=True)
        df_lv.to_csv(out_f, index=False, header=Fals
e, mode='a')
        renamebefore = "levenshteinfix"+str(ch)+".tx
t"
        ch+=1
        renameafter = "levenshteinfix"+str(ch)+".txt
"
        os.rename(renamebefore, renameafter)
#         print(ch)
#         print(datetime.datetime.now())
```

5.3.3. Wordnet Leacock-Chodorow

Untuk pengukuran mengunakan metode wordet similarity, langkah pertama yang dilakukan adalah dengan menyeleksi kata-kata bahan makanan. Hal ini dibutuhkan untuk mempercepat proses pengukuran kemiripan. Karena pada wordnet, sebuah kata yang tidak terdaftar dalam library word tidak bisa diproses pengukuran kemiripannya. Berikut baris kodenya.

```
import pandas as pd
import textdistance
import datetime

# filtering word which in wordnet synset or not

def filtering(in_f, out_f, size):
    from nltk.corpus import wordnet as wn
    print(datetime.datetime.now())
    reader = pd.read_csv(in_f, delimiter=',', sep="\
n", chunksize=size)
    ch =0
    for chunk in reader:
        df_filtering = pd.DataFrame({'code': [],'ing
redient': [],})
        for index, row in chunk.iterrows():
            if type(row['ingredient']) is str: # mem
filter apakah word berupa string
                if wn.synsets(row['ingredient'], 'n'
): # memfilter apakah word ada di wordnet (noun)
```

```
                    df_filtering = df_filtering.appe
nd({'code': row['code'],'ingredient': row['ingredien
t'],}, ignore_index=True)
        df_filtering.to_csv(out_f, index=False, head
er=False, mode='a')
        print(datetime.datetime.now())
        ch+=1
        print(ch)

filtering('tokenized_nodup.csv', 'tokenized_filtered
.csv',1000)
```

Setelah kata bahan makanan dilakukan seleksi, maka didapatkan luaran hanya bahan makanan yang terdapat pada wordnet. Data bahan makanan yang sudah diseleksi inilah yang dijadikan inputan untuk pengukuran kemiripan menggunakan wordnet, berikut baris kodenya.

```
# wordnet similarity measure leacock-chodorow

def lch_norm(in_f, out_f, size):
    from nltk.corpus import wordnet as wn
    print(datetime.datetime.now())
    reader = pd.read_csv(in_f, delimiter=',', sep="\n
", chunksize=size)
    df_tokenized = pd.read_csv(in_f, delimiter=',', s
ep="\n")
    for chunk in reader:
        err=0
            il=0
            df_lch = pd.DataFrame({'ingredient1': [],
'ingredient2': [], 'norm': []})
            for index, row in chunk.iterrows():
                syn1 = wn.synsets(row['ingredient'],
'n')[0]
                for index2, row2 in df_tokenized.iter
rows():
                    syn2 = wn.synsets(row2['ingredien
t'], 'n')[0]
                    sim = syn1.lch_similarity(syn2)
                    norm = (sim-0)/(3.63758615972638-
0)
                    il+=1
                    if norm>=0.8:
```

```
20.                     df_lch = df_lch.append({'ingredi
    ent1': row['ingredient'],'ingredient2': row2['ingred
    ient'], 'norm': norm}, ignore_index=True)
21.         df_lch.to_csv(out_f, index=False, header=Fal
    se, mode='a')
22.         print("iteration: "+str(il))
23.         print(datetime.datetime.now())
24.
25. lch_norm('tokenized_filtered.csv','E:\@Final Project
    \!NEXT STAGE\data_similarity\lch_norm3.csv',100)
```

5.4. Integrasi Data Bahan Makanan Beserta Produknya ke Database Halal Nutrition Food

Berikut proses melakukan integrasi data produk makanan dan bahan makanannya ke database halal:

5.4.1 Penggantian bahan makanan yang mirip

File masukan untuk proses ini adalah pemetaan antara kode produk makanan dengan bahan makanan. File ini merupakan hasil dari tokenisasi pada tahapan pre-processing. File masukan kedua adalah hasil pengukuran kemiripan bahan makanan yang memetakan bahan makanan yang mirip.

Tabel 5.6 cuplikan pemetaan kode produk dengan bahan makanan

Food_code	ingredient
1199	potassium sorbate
1199	carotene
1199	aroma
1663	whole milk
1663	sugar
1663	maize starch
1663	cocoa
1663	agar agar
2264	baguette poite wine bread baguette
2264	wheat flour
2264	yeast
2264	gluten
2264	braised wheat flour
2264	yeast inactivated

2264	ascorbic acid

Tabel 5.7 cuplikan pemetaan bahan makanan yang mirip

No.	Find	Replace
1	cellulose gums	cellulose gum
2	ascorbic acid e	ascorbic acid
3	l ascorbic acid	ascorbic acid
4	e ascorbic acid	ascorbic acid
5	baking powder '	baking powder
6	baking powder r	baking powder
7	altantic salmon	atlantic salmon
8	veqetable oils	vegetable oil
9	vegetable oils	vegetable oil
10	vegetables oils	vegetable oil

Dengan menggunakan masukan file dari hasil pengukuran kemiripan bahan makanan, dilakukan penggantian bahan makanan yang mirip dengan bahan makanan terkait. Proses ini dilakukan menggunakan bantuan EmEditor.

5.4.2 Memberi id ingredient

Setelah pemetaan produk makanan dengan bahan makanan yang ada sesuai dengan hasil pengukuran kemiripan, maka bahan makanan diberi id. Cara yang dilakukan adalah dengan menggunakan group by berdasakan nama bahan makanan. Kemudian masing-masing grup berdasarkan bahan makanan tersebut diisi dengan id_ingredient yang sama, berikut baris kodenya.

```
1.  df_replaced = pd.read_csv("tokenized_tahap2_dup_l
    ev_wordnet_replaced.csv", delimiter=',', sep="\t"
    ,header=None)
2.  df_replaced.columns = ["code", "ingredient","id_i
    ng"]
3.
4.  df_grup = df_replaced.groupby(['ingredient'])
5.  lists = list(df_grup.groups)
6.
```

```
ch = 0
df_dummy = pd.DataFrame({'integration': [1]})
df_dummy.to_csv("replacing_id_ing_fix_"+str(ch)+"
.txt")

for item in lists: # loooping each key in grouppe
d dataframe
    df_grup_item = df_grup.get_group(item) # gett
ing df grouped
    storeidfirst = df_grup_item['id_ing'].iloc[0]
# get first id to reference
    all_index = df_grup_item.index # get all
    df_replaced['id_ing'].iloc[all_index] = store
idfirst
    renamebefore = "replacing_id_ing_fix_"+str(ch
)+".txt"
    ch+=1
    renameafter = "replacing_id_ing_fix_"+str(ch)
+".txt"
    os.rename(renamebefore, renameafter)

df_replaced.to_csv("df_mapped.csv")
```

5.4.3 Mendapatkan daftar foodproduct

Sebagai masukan untuk tabel *foodproducts* pada database, maka daftar bahan makanan dibuat dari file pemetaan produk makanan dan bahan makanan pada tahap sebelumnya. Daftar foodproduct ini dibuat dengan melakukan penghapusan duplikat row berdasarkan kolom food_code. Berikut cuplikan baris kodenya.

```
df_product_ingredient = pd.read_csv("df_mapped.cs
v", index_col=0)
df_product = df_product_ingredient.drop(columns=[
"ingredient", "id_ing"])
df_product.drop_duplicates(subset="code", inplace
=True)
df_product
```

Kemudian data dengan food_code yang unik ini dilakukan join dengan file pada tahap pre-processing sebelum tokenisasi,

dimana terdapat kode produk dan nama produk di dalamnya, sehingga didapatkan nama produk.

Tabel 5.8 cuplikan file yang dilakukan join

Food_code	product_name	brands
2264	baguette poitevin	crous resto
3827	suedois saumon	crous
4559	peanuts	torn & glasser
5470	baguette bressan	crousresto'
16087	organic salted nut mix	grizzlies
16094	organic polenta	bob's red mill
16100	breadshop honey gone nuts granola	unfi
16117	organic long grain white rice	lundberg

5.4.4 Mendapatkan daftar ingredient

Sebagai masukan untuk tabel *ingredients* pada database, maka daftar bahan makanan dibuat dari file pemetaan produk makanan dan bahan makanan pada tahap sebelumnya. Daftar ingredient ini dibuat dengan melakukan penghapusan duplikat row berdasarkan kolom ingredient, berikut baris kodenya.

```
1. df_product_ingredient = pd.read_csv("df_mapped.cs
   v", index_col=0)
2. df_ingredient = df_product_ingredient.drop(column
   s=["code"])
3. df_ingredient.drop_duplicates(subset="id_ing", in
   place=True)
4. df_ingredient
```

Hasil akhir tabel yang dihasilkan dari keempat tahapan diatas memiliki struktur sebagaimana yang telah dijelaskan pada Tabel 4.5,

Tabel 4.6 dan Tabel 4.7 pada tahap perancangan.

BAB VI
HASIL DAN PEMBAHASAN

Pada bab ini akan dijelaskan hasil serta analisis terhadap hasil yang diperoleh dari proses implementasi yang telah dibahas pada bab sebelumnya.

6.1 Hasil

6.1.1 Hasil Pengukuran Kemiripan bahan Makanan

Pengujian dari hasil pengukuran kemiripan bahan makanan ini menggunakan pengukuran *precision* dan *recall*. Pengujian tersebut menggunakan tabel 2x2 dengan berisi variabel berbeda pada masing-masing cellnya. Yaitu true positive (TP), true negative (TN), false positive (FP), dan false negative (FN).

Recall adalah tingkat keberhasilan sistem dalam memberikan informasi yang benar.

$$\text{Recall} = \frac{TP}{TP+FN}$$

Sedangkan precision adalah tingkat ketepatan antara informasi yang sebenarnya dengan hasil yang diberikan oleh sistem.

$$\text{Precision} = \frac{TP}{TP+FP}$$

Literatur yang digunakan sebagai bahan validasi pada pengujian ini adalah buku Dictionary of Food Ingredients Fourth Edition [30], dan pencarian google.

Berikut hasil pengukuran kemiripan bahan makanan dan pengujiannya.

6.1.1.1 Jaccard distance

Pengukuran kemiripan menggunakan Jaccard Distance menghasilkan 757,440 pasang data yang dianggap mirip oleh sistem.

Berikut tabel hasil pengujian dari pengukuran kemiripan menggunakan jaccard distance.

Tabel 6.1 Hasil pengujian jaccard

Jaccard Distance		
Jumlah sampel: 200	True (dengan benar diduga)	False (dengan salah diduga)
Positive (pengukuran mirip)	25	6
Negative (pengukuran tidak mirip)	137	31

Berdasarkan tabel di atas, dapat diukur recall dan precision dari pengukuran kemiripan sebagai berikut.

TP: 22, FP: 9, TN: 158, FN: 11

Recall = $\frac{TP}{TP+FN} = 44.6\%$

Precision = $\frac{TP}{TP+FP} = 80.6\%$

6.1.1.2 Jaro-winkler distance

Pengukuran kemiripan menggunakan Jaro-Winkler Distance menghasilkan 31,271,070 pasang data yang dianggap mirip oleh sistem.

Berikut tabel hasil pengujian dari pengukuran kemiripan menggunakan jaro-winkler distance.

Tabel 6.2 Hasil pengujian jaro-winkler

Jaro-Winkler Distance		
Jumlah sampel: 200	True (dengan benar diduga)	False (dengan salah diduga)
Positive (pengukuran mirip)	29	6
Negative (pengukuran tidak mirip)	137	27

Berdasarkan tabel di atas, dapat diukur recall dan precision dari pengukuran kemiripan sebagai berikut.

TP: 27, FP: 0, TN: 158, FN: 15

Recall = $\frac{TP}{TP+FN} = 51.8\%$

Precision = $\frac{TP}{TP+FP} = 82.9\%$

6.1.1.3 Levenshtein distance

Pengukuran kemiripan menggunakan Levenshtein Distance menghasilkan 57,044 pasang data yang dianggap mirip oleh sistem.

Berikut tabel hasil pengujian dari pengukuran kemiripan menggunakan levenshtein distance.

Tabel 6.3 Hasil pengujian levenshtein

Levenshtein Distance		
Jumlah sampel: 200	True (dengan benar diduga)	False (dengan salah diduga)
Positive (pengukuran mirip)	26	0
Negative (pengukuran tidak mirip)	143	30

Berdasarkan tabel di atas, dapat diukur recall dan precision dari pengukuran kemiripan sebagai berikut.

TP: 27, FP: 0, TN: 158, FN: 15

Recall = $\frac{TP}{TP+FN} = 46.4\%$

Precision = $\frac{TP}{TP+FP} = 100\%$

6.1.1.4 Wordnet LCH similarity

Pengukuran kemiripan menggunakan Wordnet Leacock-Chodorow similarity menghasilkan 3,085 pasang data yang dianggap mirip oleh sistem.

Berikut tabel hasil pengujian dari pengukuran kemiripan menggunakan wordnet.

Tabel 6.4 Hasil pengujian wordnet

Wordnet LCH similarity		
Jumlah sampel: 200	True (dengan benar diduga)	False (dengan salah diduga)
Positive (pengukuran mirip)	42	59
Negative (pengukuran tidak mirip)	84	14

Berdasarkan tabel di atas, dapat diukur recall dan precision dari pengukuran kemiripan sebagai berikut.

TP: 59, FP: 99, TN: 42, FN: 0

Recall = $\frac{TP}{TP+FN} = 75.0\%$

Precision = $\frac{TP}{TP+FP} = 41.6\%$

6.1.1.5 Wordnet LCH similarity dan jaccard distance

Penggabungan pengukuran kemiripan antara wordnet dan jaccard distance menghasilkan hasil pengujian sebagai berikut.

Tabel 6.5 Hasil pengujian wordnet & jaccard

Wordnet & Jaccard		
Jumlah sampel: 200	True (dengan benar diduga)	False (dengan salah diduga)
Positive (pengukuran mirip)	45	65
Negative (pengukuran tidak mirip)	78	11

Berdasarkan tabel di atas, dapat diukur recall dan precision dari pengukuran kemiripan sebagai berikut.

TP: 59, FP: 99, TN: 42, FN: 0

Recall = $\frac{TP}{TP+FN} = 80.4\%$

Precision = $\frac{TP}{TP+FP} = 40.9\%$

6.1.1.6 Wordnet LCH similarity dan jaro-winkler distance

Penggabungan pengukuran kemiripan antara wordnet dan jaro-winkler distance menghasilkan hasil pengujian sebagai berikut.

Tabel 6.6 Hasil pengujian wordnet & jaro-winkler

Wordnet & Jaro-Winkler		
Jumlah sampel: 200	True (dengan benar diduga)	False (dengan salah diduga)
Positive (pengukuran mirip)	45	65
Negative (pengukuran tidak mirip)	78	11

Berdasarkan tabel di atas, dapat diukur recall dan precision dari pengukuran kemiripan sebagai berikut.

TP: 59, FP: 99, TN: 42, FN: 0

Recall = $\frac{TP}{TP+FN} = 80.4\%$

Precision = $\frac{TP}{TP+FP} = 40.9\%$

6.1.1.7 Wordnet LCH similarity dan levenshtein distance

Penggabungan pengukuran kemiripan antara wordnet dan levenshtein distance menghasilkan hasil pengujian sebagai berikut.

Tabel 6.7 Hasil pengujian wordnet & levenshtein

Wordnet & Levenshtein		
Jumlah sampel: 200	True (dengan benar diduga)	False (dengan salah diduga)
Negative (pengukuran tidak mirip)	42	59
Positive (pengukuran mirip)	84	14

Berdasarkan tabel di atas, dapat diukur recall dan precision dari pengukuran kemiripan sebagai berikut.

TP: 59, FP: 99, TN: 42, FN: 0

Recall = $\frac{TP}{TP+FN} = 75.0\%$

Precision = $\frac{TP}{TP+FP} = 41.6\%$

6.1.2 Performa kode program dan komputer yang digunakan untuk pengukuran kemiripan bahan makanan

Pengukuran kemiripan yang dilakukan dihitung waktu yang dibutuhkan untuk selesainya masing-masing metode pengukuran kemiripan, dan banyak perangkat komputer yang dibutuhkan, dimana spesifikasi perangkat keras komputer yang digunakan telah dijelaskan pada Tabel 5.1. Berikut adalah

performa kode program dan komputer yang digunakan untuk pengukuran kemiripan bahan makanan.

Tabel 6.8 Performa

No	Metode	Input (jumlah kata)	Output (jumlah pasang kata yang mirip)	Waktu dan jumlah PC
1	Jaccard	271,824	757,440	88 jam, 7 PC
2	Jaro-Winkler		31,271,070	78 jam, 7 PC
3	Levenshtein		57,044	173 jam, 7 PC
4	Wordnet	16,400 (terfilter hanya kata yang terdapat pada database leksikal wordnet saja)	3,085	1.5 jam, 1 PC

6.1.3 Hasil Integrasi ke database halal

Integrasi data yang dilakukan menggunakan hasil pengukuran kemiripan bahan makanan Levenshtein distance dan wordnet similarity sebagai bahan masukan. Berikut informasi hasil integrasi data yang telah dilakukan:

Tabel 6.9 hasil integrasi data

Keterangan	Jumlah
Data bahan makanan sebelum implementasi hasil pengukuran kemiripan:	271,823

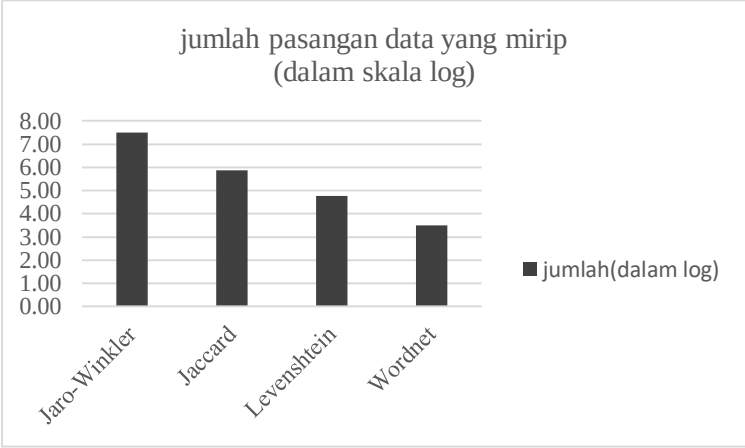
Keterangan	Jumlah
Data bahan makanan setelah implementasi hasil pengukuran kemiripan:	83,531
Data bahan makanan yang telah ada pada database halal beririsan dengan data baru:	2,529
Data bahan makanan yang berhasil diintegrasikan:	81,001
Data jumlah produk baru yang berhasil diintegrasikan	39,519

Spesifikasi komputer yang digunakan

6.2 Pembahasan

6.2.1 Jumlah keseluruhan pasangan data yang mirip

Dari total 271,823 bahan makanan yang menjadi inputan untuk proses pengukuran kemiripan bahan makanan, berikut jumlah hasil komparasi yang dianggap mirip oleh sistem (memenuhi threshold 80%).

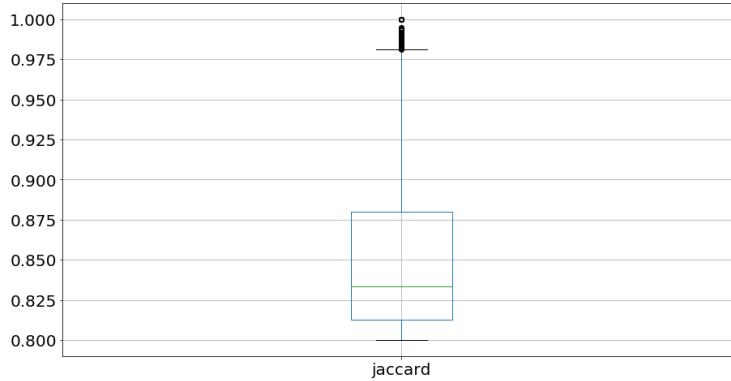


Gambar 6.1 Diagram jumlah keseluruhan pengukuran kemiripan

Keterangan jumlah pasang data yang mirip

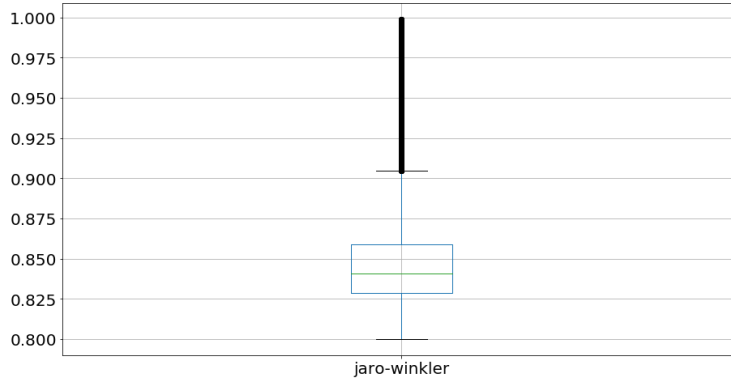
Jaro-Winkler : 31,271,070
Jaccard : 757,440
Levenshtein : 57,044
Wordnet : 3,085

6.2.2 Boxplot keseluruhan pengukuran kemiripan



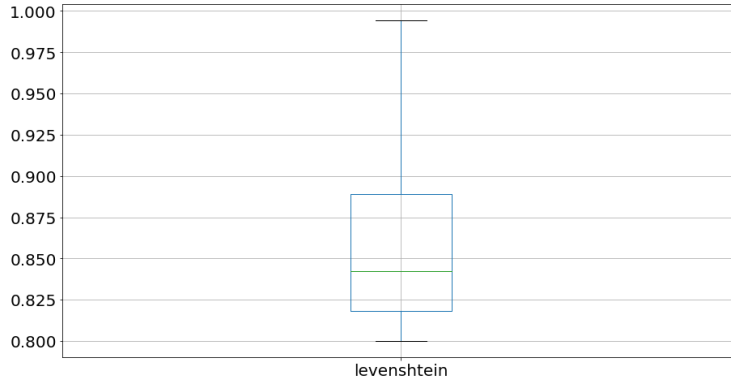
Gambar 6.2 boxplot jaccard distance

Visualisasi boxplot hasil pengukuran kemiripan bahan makanan menggunakan jaccard distance dapat dilihat pada Gambar 6.2. Data hasil pengukuran kemiripan yang divisualisasikan adalah yang melewati threshold 80%(0.8). Boxplot menunjukkan bahwa quartil 1 0.82, quartil 2 atau mediannya adalah 0.84, dan quartil 3 0.89.



Gambar 6.3 boxplot jaro-winkler distance

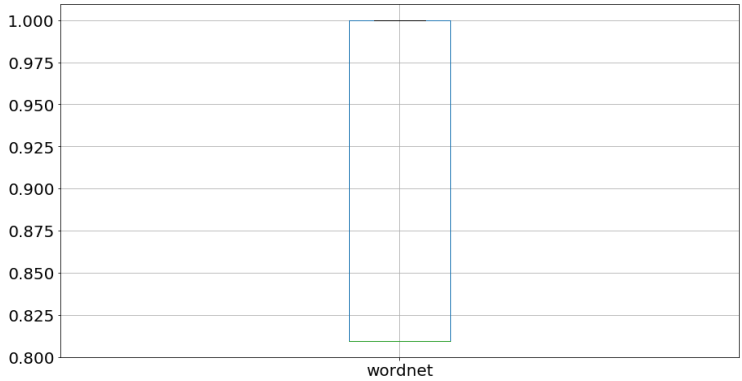
Visualisasi boxplot hasil pengukuran kemiripan bahan makanan menggunakan jaro-winkler distance dapat dilihat pada Gambar 6.3. Data hasil pengukuran kemiripan yang divisualisasikan adalah yang melewati threshold 80%(0.8). Boxplot menunjukkan bahwa quartil 1 0.83, quartil 2 atau mediannya adalah 0.84, dan quartil 3 0.86.



Gambar 6.4 boxplot levenshtein distance

Visualisasi boxplot hasil pengukuran kemiripan bahan makanan menggunakan levenshtein distance dapat dilihat pada Gambar 6.4Gambar 6.2. Data hasil pengukuran kemiripan yang

divisualisasikan adalah yang melewati threshold 80%(0.8). Boxplot menunjukkan bahwa quartil 1 0.81, quartil 2 atau mediannya adalah 0.83, dan quartil 3 0.88.



Gambar 6.5 boxplot wordnet

Visualisasi boxplot hasil pengukuran kemiripan bahan makanan menggunakan levenshtein distance dapat dilihat pada Gambar 6.5Gambar 6.2. Data hasil pengukuran kemiripan yang divisualisasikan adalah yang melewati threshold 80%(0.8). Boxplot menunjukkan bahwa quartil 1 0.81, quartil 2 atau mediannya adalah 0.83, dan quartil 3 0.88.

6.2.3 Keseluruhan *precision* dan *recall*

Berikut rangkuman seluruh *recall* dan *precision* dari semua pengukuran kemiripan.

Tabel 6.10 Daftar seluruh *recall* dan *precision*

Butir Uji	Nama metode	Precision	Recall
T.1.	Jacard Distance	80.6%	44.6%
T.2.	Jaro-Winkler Distance	82.9%	51.8%
T.3.	Levenshtein Distance	100.0%	46.4%
K.1.	Wordnet LCH similarity	41.6%	75.0%
G.1.	Wordnet & jaccard	40.9%	80.4%
G.2.	Wordnet & jaro-winkler	40.9%	80.4%
G.3.	Wordnet & levenshtein	41.6%	75.0%

6.2.4 Pembahasan keseluruhan kemiripan bahan makanan

Jumlah keseluruhan pasangan data yang mirip

Pada hasil pengukuran kemiripan, dapat dilihat bahwa pengukuran kemiripan kontekstual (wordnet) jauh lebih sedikit menghasilkan data bahan makanan yang mirip dibandingkan dengan metode pengukuran kemiripan tekstual (jaccard, levenshtein, jaro-winkler). Hal ini dikarenakan beberapa faktor, diantaranya, wordnet tidak memproses kata-kata yang tidak terdapat dalam kamus/library wordnet. Jadi kata bahan makanan yang salah ketik (contoh: ‘pinrapple’; yang seharusnya ‘pineapple’) tidak akan masuk ke dalam proses pengukuran kemiripan wordnet. Sedangkan pada metode pengukuran kemiripan berdasarkan tekstual kata, hal tersebut tetap dimasukkan dalam proses pengukuran kemiripan.

Namun demikian wordnet memiliki *recall* yang cukup baik (75%) dibandingkan dengan pengukuran kemiripan yang lain. Dalam arti, wordnet cukup berhasil memberikan banyak dugaan kemiripan antara dua bahan makanan yang memang secara nyata (setelah divalidasi) mirip. Hal ini dikarenakan wordnet memang memiliki hierarki kata sehingga bisa memaknai kata secara arti bukan secara tekstual. Berikut cuplikan data sebagai gambaran:

Tabel 6.11 cuplikan data perbandingan ukuran kemiripan

No	Data1	Data2	Real	Wordnet	Jaccard	Jaro	Lev
1	cu	copper	true	1	0.17	0.72	0.72
2	ns	nitrogen	true	1	0.2	0.73	0.73
3	ses	se	true	1	0.25	0.75	0.75
4	meat	boeuf	true	0.8	0	0	0
5	colour	dyes	true	0.8	0	0	0

Kemudian, diantara ketiga metode pengukuran data secara tekstual/fuzzy string yaitu jaccard, jaro-winkler dan levenshtein distance, dilihat dari recall dan precisionnya, yang terbaik adalah levenshtein distance. Bisa dilihat pada tabel

Tabel 6.10, bahwa pada pengujian dengan 200 sampel, levenshtein memiliki *precision* 100% dan *recall* 46.4%. walaupun *recall* sedikit lebih rendah dibandingkan jaro-winkler (51.8%) , namun tidak lebih rendah dari jaccard dan levenshtein lebih unggul dalam precision. Dengan kata lain, pengukuran levenshtein distance berhasil memberikan ketepatan yang tinggi terhadap kesamaan bahan makanan yang diduga secara tepat benar mirip. Hal ini dikarenakan levenshtein menghitung jarak edit terpendek untuk mengubah satu string menjadi string lainnya. Edit yang dilakukan bisa merupakan salah satu dari ketiga jenis edit, yaitu *insertion*, *deletion*, dan *subtitution*. Contoh:

Levenshtein distance diantara “kitten” dan “sitting” adalah 3, karena terdapat tiga pengeditan untuk mengubah string satu ke yang lain, dan tidak ada cara lain untuk melakukannya kurang dari tiga pengeditan:

1. **k**itten → **s**itten (subtitusi karakter "s" terhadap "k")

2. sitt**e**n → sitt**i**n (subtitusi karakter "i" terhadap "e ")
3. sittin → sittin**g** (penambahan karakter "g" di akhir).

Sementara untuk metode pengukuran kemiripan jaro-winkler, sangat banyak bahan makanan yang dainggap mirip padahal sebenarnya tidak mirip. Dengan menggunakan perhitungan seperti halnya pada 2.2, jaro-winkler menghasilkan kemiripan di atas threshold jauh lebih banyak daripada pengukuran kemiripan mengguanakan metode yang lain (**Error! Reference source not found.**). Metode pengukuran kemiripan yang digunakan jaro-winkler adalah dengan

1. Menghitung panjang string,
2. Menemukan jumlah karakter yang sama di dalam dua string, dan
3. Menemukan jumlah transposisi

Sedangkan pengukuran kemiripan menggunakan jaccard distance adalah dengan mengukur irisan antar dua string. Semakin banyak irisan yang ada, semakin tinggi pula similaritynya. Hal ini kurang relevan karena bisa jadi dua string yang panjang memiliki nilai kemiripan yang tinggi hanya karena jumlah irisan yang banyak, padahal tidak lebih mirip dari dua string yang lebih pendek.

Maka dari itu, gabungan antara pengukuran kemiripan menggunakan levenshtein distance dan wordnet yang ditemukan palilng optimal untuk digunakan dalam pengukuran kemipan dan akan digunakan untuk mengintegrasikan data bahan makanan ke database halal nutrition food.

BAB VII
KESIMPULAN DAN SARAN

Bab ini menjelaskan kesimpulan yang didapat dari seluruh proses pengerjaan tugas akhir dan saran perbaikan untuk penelitian kedepannya untuk penelitian serupa di masa mendatang.

7.1 Kesimpulan

Kesimpulan yang dapat diambil dari pengerjaan tugas akhir ini adalah sebagai berikut.

1. Implementasi hasil pengukuran kemiripan bahan makanan secara signifikan mengurangi redundansi data pada bahan makanan. Dimana data bahan makanan dari open food fact sebelum dilakukan implementasi hasil pengukuran kemiripan adalah 271,823, kemudian setelah dilakukan implementasi pengukuran kemiripan menjadi 83,531 (berkurang 70%).
2. Pengukuran kemiripan bahan makanan secara tekstual menggunakan Levenshtein distance merupakan metode yang paling akurat dibandingkan dengan Jaro-Winkler dan Jaccard distance. Hal ini juga didukung algoritma Levenshtein yang lebih kompleks dibadingkan dengan dua metode pengukuran secara tekstual kata lainnya, yaitu mempertimbangkan edit distance insertion, deletion dan subtitution dalam algoritma pengukurannya. Dalam hasil pengujian, levenshtein memiliki precision 100.0% dan recall 46.4%. Wordnet menunjukkan performa yang baik terhadap pengukuran kemiripan bahan makanan secara koseptual. Dimana berdasarkan hasil pengujian, wordnet memiliki precision 41.6% dan recall 75.0%.
3. Penggabungan pengukuran kemiripan bahan makanan menggunakan Levenshtein distance dan Wordnet LCH merupakan kombinasi yang cukup baik untuk mengukur kemiripan bahan makanan dengan

mempertimbangkan kemiripan tekstual kata dan makna kata, dimana precision dan recall gabungan kedua metode pengukuran kemiripan ini adalah 41.6% dan 75.0%.

4. Integrasi data bahan makanan dan produk makanan ke database halal nutrition food yang berhasil dilakukan adalah 81,001 data bahan makanan dan 39,519 produk makanan baru.

7.2 Saran

Dalam pengerjaan tugas akhir, saran yang diharapkan dapat bermanfaat untuk pengembangan penelitian ke depan, yaitu:

1. Melakukan pengukuran kemiripan arti/sinonim bahan makanan menggunakan WordNet dengan metode yang lain, seperti Wu and Palmer's, Resnik, atau Jiang & Conrath.
2. Menerapkan pengukuran kemiripan konseptual terhadap fitur pencarian pada aplikasi Halal Nutrition Food sehingga fungsi pencarian bisa menjadi leih relevan.
3. Mengintegrasikan data makanan yang baru, dengan mengukur kemiripan bahan makanan dan juga mengukur kemiripan nama produk/brand. Karena pada tugas akhir ini, pengukuran kemiripan hanya dilakukan kepada bahan makanan untuk mengintegrasikan data makanan yang baru ke database halal nutrition food yang sudah ada.

DAFTAR PUSTAKA

[1] B. P. Statistik, "Sensus Penduduk," Badan Pusat Statistik, Jakarta, 2010.

[2] L. MUI, "LPPOM MUI," LPPOM MUI, [Online]. Available: http://www.halalmui.org/mui14/index.php/main/go_to_section/130/1511/page/1. [Accessed 1 February 2019].

[3] L. MUI, "LPPOM MUI," January 2018. [Online]. Available: http://www.halalmui.org/images/stories/pdf/LSH/LSH LN-LPPOM%20MUI.pdf. [Accessed 1 February 2019].

[4] L. MUI, "Data Sertifikasi Halal LPPOM MUI Periode 2011–2018," LPPOM MUI, Jakarta, 2019.

[5] "Halal Nutrition Food," [Online]. Available: http://halal.addi.is.its.ac.id/about. [Accessed 18 January 2019].

[6] "Open Food Facts," [Online]. Available: https://world.openfoodfacts.org/discover. [Accessed 18 January 2019].

[7] B. F. Yusuf, "Integrasi Data dan VIsualisasi Graf Pada Aplikasi Halal Nutrition Food," *TUGAS AKHIR – KS141501,* 2018.

[8] C. Leacock and M. Chodorow, "Combining local context and WordNet Similarity for Word Sense Identification," in *WordNet: An electronic lexical database* , 1998, pp. 265-283.

[9] M. Warin, "Using WordNet and Semantic Similarity to Disambiguate an Ontology," 2004.

[10] P. W. Hongzhe Liu, "Assessing Sentence Similarity Using WordNet based Word Similarity," *JOURNAL OF SOFTWARE,* vol. 6, no. 6, pp. 1451-1458, 2013.

[11] J. Fatawi, "Rancang Bangun Perangkat Lunak Linked Open Data Halal Dan Gizi Pada Produk Makanan Dan Minuman," 2016.

[12] A. M. Fajriyadi, "Peningkatan Relevansi Pencarian Produk Halal dalam Aplikasi Halal Nutrition Menggunakan Algoritma OKAPI BM25F," 2017.

[13] A. A. Firmansyah, "Pengembangan Pencarian Produk Terkait Menggunakan Euclidian Distance dan Cosine Similarity pada Aplikasi Halal Food Nutrition," 2017.

[14] A. C. Najib, "Rancang Bangun Aplikasi Android Halal Nutrition Food Menggunakan Kombinasi Query-Independent dan Query-Dependent Ranking," 2017.

[15] "The Blog of Open Food Facts," [Online]. Available: https://en.blog.openfoodfacts.org/news/open-food-facts-celebrates-its-5th-anniversary-with-more-than-1000-cakes. [Accessed 18 January 2019].

[16] "W3C," [Online]. Available: https://www.w3.org/RDF/. [Accessed 18 January 2019].

[17] "W3C Working Group Note," 24 June 2014. [Online]. Available: https://www.w3.org/TR/rdf11-primer/. [Accessed 18 January 2019].

[18] "Lucene," [Online]. Available: http://lucene.apache.org/. [Accessed 18 January 2019].

[19] "WorNet: a lexical database for English," [Online]. Available: https://wordnet.princeton.edu/. [Accessed 18 January 2019].

[20] "Meta::CPAN," [Online]. Available: https://metacpan.org/pod/release/SID/WordNet-Similarity-1.04/lib/WordNet/Similarity/path.pm. [Accessed 18 January 2019].

[21] J. H. Govinda Grings, "A Java Library for Fuzzy String Matching," 2012.

[22] R. &. F. M. Wagner, "The string-to-string correction problem," *Journal of the ACM (JACM)*, vol. 21, no. 1, p. 168–173, 1974.

[23] V. Levenshtein, " Binary codes capable of correcting deletions, insertions, and reversals," *In Soviet Physics Doklady,* vol. 10, no. 8, p. 707–710, 1966.

[24] W. E. Winkler, "The state of record linkage and current research problems," *Statistics of Income Division, Internal Revenue Service Publication,* vol. R99/04, 1999.

[25] W. E. Winkler, "Overview of record linkage and current research directions," *Research Report Series, RRS,* 2006.

[26] S. P. S. R. Anna Kurniawati, "Implementasi Algoritma Jaro-Winkler Distance untuk Membandingkan Kesamaan Dokumen Berbahasa Indonesia," 2010.

[27] P. Jaccard, "Étude comparative de la distribution florale dans une portion des Alpes et des Jura," *Bull Soc Vaudoise Sci Nat,* vol. 37, pp. 547-579, 1901.

[28] M. Shipman, "NC State University," 21 March 2014. [Online]. Available: https://news.ncsu.edu/2014/05/baking-soda-powder/. [Accessed 4 March 2019].

[29] "National Center for Biotechnology Information," [Online]. Available: https://www.ncbi.nlm.nih.gov/mesh/68012965. [Accessed 4 March 2019].

[30] M. M. a. Y. H. H. P. Robert S. Igoe, Dictionary of Food Ingredients Fourth Edition, Gaithersburg: Aspen Publishers, Inc, 2001.

Halaman ini sengaja dikosongkan

A. LAMPIRAN A: DATA

Semua berkas lampiran dapat dilihat pada http://bit.ly/lampiranTAhalal2019

A.1 Stopword

A.2 Data bahan makanan sebelum pre-processing

A.3 Data sebelum ditokenisasi

A.4 Data bahan makanan setelah pre-processing

A.5 Pengujian recall dan precision

A.6 Hasil pengukuran kemiripan Jaccard

A.7 Hasil pengukuran kemiripan Levenshtein

A.8 Hasil pengukuran kemiripan Wordnet

B. LAMPIRAN B: KODE PROGRAM

Semua berkas lampiran dapat dilihat pada http://bit.ly/lampiranTAhalal2019

B.1 Kode program pengukuran kemiripan Wordnet

B.2 Kode program pengukuran kemiripan Jaccard

B.3 Kode program pengukuran kemiripan Jaro-Winkler

B.4 Kode program pengukuran kemiripan Levenshtein

Halaman ini sengaja dikosongkan

BIODATA PENULIS

Penulis bernama Miftahul Jannah, lahir di Surabaya, 25 Januari 1998. Merupakan anak kedua dari empat bersaudara. Penulis telah menempuh pendidikan formal di SDIT Ar-Rahmah Lumajang, SMPIT Albanna Denpasar, dan SMAIT As Syifa Boarding School Subang.

Pada tahun 2015, penulis melanjutkan studi ke jenjang pendidikan yang lebih tinggi di Institut Teknologi Sepuluh Nopember sebagai mahasiswi Departemen Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi. Penulis terdaftar sebagai mahasiswi dengan nomor induk (NRP) 05211540000153. Selama masa perkuliahan, penulis aktif di beberapa organisasi, mulai dari tentor keilmiahan HMSI, Lembaga Dakwah Jurusan Sistem Informasi, dan Lembaga Dakwah Kampus (JMMI ITS). Penulis juga pernah aktif di beberapa kepanitiaan acara sebagai staf/koordinator umumnya di bagian publikasi dan dokumentasi, dalam berbagai acara mulai dari tingkat departemen, tingkat kampus, maupun tingkat Surabaya/sekitarnya. Selain itu penulis juga pernah aktif menjadi supervisor asrama mahasiswa di bawah naungan Yayasan SDM Iptek Surabaya.

Pada tahun keempat perkuliahan, penulis mengambil bidang minat laboratorium Akuisisi Data dan Diseminasi Informasi (ADDI) dan mendalami topik Integrasi Data. Penulis dapat dihubungi melalui email jannaher98@gmail.com.