TUGAS AKHIR - KI141502

VISUALISASI SIMILARITAS TOPIK PENELITIAN DENGAN PENDEKATAN KARTOGRAFI MENGGUNAKAN SELF-ORGANIZING MAPS (SOM)

BUDI PANGESTU TANUJAYA
NRP 5113100064

Dosen Pembimbing I
Dr. Chastine Fatichah, S.Kom., M.Sc.

Dosen Pembimbing II
Diana Purwitasari, S.Kom., M.Sc.

Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

TUGAS AKHIR – KI141502

VISUALISASI SIMILARITAS TOPIK PENELITIAN DENGAN PENDEKATAN KARTOGRAFI MENGGUNAKAN SELF-ORGANIZING MAPS (SOM)

BUDI PANGESTU TANUJAYA
NRP 5113100064

Dosen Pembimbing I
Dr. Chastine Fatichah, S.Kom., M.Sc.

Dosen Pembimbing II
Diana Purwitasari, S.Kom., M.Sc.

Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

(Halaman ini sengaja dikosongkan)

UNDERGRADUATE THESES – KI141502

VISUALIZATION OF RESEARCH TOPIC SIMILARITY WITH CARTOGRAPHIC APPROACH USING SELF-ORGANIZING MAPS (SOM)

BUDI PANGESTU TANUJAYA
NRP 5113100064

First Advisor
Dr. Chastine Fatichah, S.Kom., M.Sc.

Second Advisor
Diana Purwitasari, S.Kom., M.Sc

Department of Informatics
Faculty of Information Technology
Sepuluh Nopember Institute of Technology
Surabaya 2017

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

VISUALISASI SIMILARITAS TOPIK PENELITIAN DENGAN PENDEKATAN KARTOGRAFI MENGGUNAKAN SELF-ORGANIZING MAPS (SOM)

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Cerdas dan Visi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
BUDI PANGESTU TANUJAYA
NRP: 5113100064

Disetujui oleh Pembimbing Tugas Akhir:

1. Dr. Chastine Fatichah, S.Kom., M.Cs
   (NIP. *197512202001122002*) ..... (Pembimbing 1)

2. Diana Purwitasari, S.Kom., M.Sc
   (NIP. *197804102003122001*) ..... (Pembimbing 2)

SURABAYA
JULI, 2017

(Halaman ini sengaja dikosongkan)

VISUALISASI SIMILARITAS TOPIK PENELITIAN DENGAN PENDEKATAN KARTOGRAFI MENGGUNAKAN SELF-ORGANIZING MAPS (SOM)

Nama Mahasiswa	**: BUDI PANGESTU TANUJAYA**
NRP	**: 5113100064**
Jurusan	**: Teknik Informatika FTIF-ITS**
Dosen Pembimbing 1	**: Dr. Chastine Fatichah, S.Kom., M.Sc.**
Dosen Pembimbing 2	**: Diana Purwitasari, S.Kom., M.Sc.**

Abstrak

Penelitian merupakan salah satu hal yang penting dalam pengembangan bidang keilmuan sehingga dinilai perlu diciptakan sebuah visualisasi Peta Keterkaitan Antar Topik Riset Penelitian, agar mampu memberikan ide dan gambaran bagi calon peneliti dari Indonesia tentang potensi Topik Penelitian yang dapat dikembangkan. Self-organizing Maps merupakan algoritma Neural Network yang dinilai cocok untuk visualisasi Peta dengan pendekatan Kartografi dikarenakan struktur keduanya yang mirip. Teknik Clustering juga digunakan untuk menyederhanakan Peta Jaringan Neuron hasil SOM.

Pada penelitian kali ini, akan digunakan Data Penelitian Tugas Akhir dari Resits.its.ac.id sebagai data input. Pemrosesan Data Mining pada data teks seringkali memiliki kendala dalam kata-kata yang terdapat pada corpus terlalu kotor atau biasa disebut stopwords, dan besarnya dimensi fitur yang didapat dari data teks sangat besar. Maka dari itu, perlu dilakukan preprocessing pada data teks yang digunakan meliputi Stopwords Removal, dan Tokenizing. Setelah melalui preprocessing, dilakukan Ekstraksi Fitur menggunakan Term Frequency – Inverse Document Frequency (TF-IDF). Reduksi fitur menggunakan Principal Component Analysis (PCA) dikenakan guna mereduksi fitur dari Data Input yang dinilai terlalu banyak.

Setelah itu, Data Input dilatih dan dipetakan ke dalam 2 dimensi menggunakan metode Unsupervised Learning Self-organizing Maps (SOM). Terakhir, Teknik Clustering Kombinasi K-means dan Hierarchical Clustering dikenakan pada Jaringan Peta SOM guna mengelompokkan neuron-neuron yang terbentuk. Hasil akhir dari penelitian ini ilaha Peta Visualisasi Similaritas Topik Penelitian.

Berdasarkan hasil uji coba, dapat disimpulkan bahwa ekstraksi fitur dan Teknik cluster yang digunakan sudah tepat divalidasi dengan Silhouette Score sebesar 0.5215, dan Cophenet Correlation Coefficient sebsar 0.977. Uji coba diatas menunjukkan bahwa K-means Clustering yang digunakan menghasilkan Cluster yang Cohesive dan Separable ditandai dengan hasil Silhouette Score dan Cophenet Correlation Coefficient yang besar.

Kata kunci: Topik Penelitian, Text Mining, Visualisasi, Kartografi, Term Frequency – Inverse Document Frequency (TF-IDF), Self-organizing Maps, K-means Clustering, Hierarchical Clustering.

VISUALIZATION OF RESEARCH TOPIC SIMILARITY WITH CARTOGRAPHIC APPROACH USING SELF-ORGANIZING MAPS (SOM)

Student's Name	**: BUDI PANGESTU TANUJAYA**
Student's ID	**: 5113100064**
Department	**: Teknik Informatika FTIF-ITS**
First Advisor	**: Dr. Chastine Fatichah, S.Kom., M.Sc.**
Second Advisor	**: Diana Purwitasari, S.Kom., M.Sc.**

Abstract

Research is one of the most important thing in educational sector, therefore, it's assessed that a visualization of Similarity between Research Topic is needed to be able to give ideas about potential research topic to Indonesian Researcher. Self-organizing Maps is an Neural Network algorithm that is suitable for Cartographic Approach Visualization because of the similar structure. Clustering Techniques are also needed to simplify the Neural Network Map for better visualization.

In this research, Undergraduate Thesis Repository from Resits.its.ac.id is used as Input Data. Data Mining on texts data often has issues with many meaningless words that exist in the corpus or usually called as stopwords, also the size of features used to describe each document is relatively big. Therefore, preprocessing needs to be applied to the data includes Stopwords Removal and Tokenizing. After preprocessing, Feature Extraction using Term Frequency – Inverse Document Frequency is applied. Next, Principal Component Analysis (PCA) is applied as the Feature Reduction Method. After that, Input Data is applied into an Unsupervised Learning Self-organizing Maps (SOM) to map the data as 2 dimensional networks. Lastly, Clustering using the combination of K-means and Hierarchical Clustering is applied to the neural networks resulted from SOM. The final output is a Visualization of Research Topic Similarity.

Based on the results of the trial, it can be concluded that the feature extraction method and clustering techniques used are proper validated with Silhouette Score in the amount of 0.5215 and 0.977 in Cophenet Correlation Coefficient. Moreover, users also assess that the Visualized Map of Research Topic Similarity is informative. The Experiment Results above concludes that the Clusters resulted from K-means Clustering are Cohesive and Separable based on the high value of Silhouette Score and Cophenet Correlation Coefficient.

Keywords : Researches Topic, Text Mining, Visualization, Cartographic, Term Frequency – Inverse Document Frequency (TF-IDF), Self-organizing Maps, K-means Clustering, Hierarchical Clustering

KATA PENGANTAR

Puji Tuhan kepada Tuhan Yang Maha Esa yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul:

"**Visualisasi Similaritas Topik Penelitian dengan Pendekatan Kartografi menggunakan Self-organizing Maps (SOM)**"

Terselesaikannya Tugas Akhir ini tidak terlepas dari bantuan dan dukungan banyak pihak, Oleh karena itu melalui lembar ini penulis ingin mengucapkan terima kasih dan penghormatan kepada:

1. Tuhan Yang Maha Esa yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir.
2. Keluarga Penulis tercinta, Lucky Tanujaya, Joo Melia Setiawati, Budi Chandra Tanujaya, Budi Stevie Tanujaya yang senantiasa memberikan semangat, dukungan, dan doa kepada penulis.
3. Ibu Dr. Chastine Fathicah, S.Kom., M.Sc. dan Ibu Diana Purwitasari, S.Kom., M.Sc. selaku pembimbing I dan II yang telah membimbing dan memberikan motivasi, nasihat dan bimbingan dalam menyelesaikan Tugas Akhir ini.
4. Ibu Wijayanti Nurul Khotimah, S.Kom., M.Ksc. selaku dosen wali penulis yang telah memberikan arahan kepada penulis selama menjalani perkuliahan di Teknik Informatika ITS.
5. Setyassida Novian Putra Damara (Ovan), selaku sahabat penulis yang senantiasa memberikan semangat, doa, dukungan, dan menghibur penulis, juga selalu mengingatkan penulis agar menyelesaikan Tugas Akhir ini, dan memberikan masukan-masukan terkait dengan Tugas Akhir ini.

6. Bapak Darlis Herumurti, S.Kom., M.Kom. selaku kepala jurusan Teknik Informatika ITS dan segenap dosen dan karyawan Teknik Informatika ITS yang telah memberikan ilmu dan pengalaman kepada penulis selama menjalani masa studi di Teknik Informatika ITS.
7. Nindyasari Dewi Utari, selaku sahabat penulis yang selalu mengingatkan penulis untuk mengerjakan Tugas Akhir ini.
8. Sahabat-sahabat penulis dari angkatan TC 2013, maupun kakak dan adik kelas TC 2011, TC 2012, TC 2014, dan TC 2015 yang tidak dapat disebutkan satu persatu namun sangat membantu dalam semangat, kelancaran dan selesainya Tugas Akhir ini.
9. Serta semua pihak yang yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa laporan Tugas Akhir ini masih memiliki banyak kekurangan. Oleh karena itu dengan segala kerendahan hati penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan penulis kedapannya. Selain itu, penulis berharap laporan Tugas Akhir ini dapat berguna bagi pembaca secara umum.

Surabaya, Juni 2017

DAFTAR ISI

(Halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

DAFTAR KODE SUMBER

(Halaman sengaja dikosongkan)

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pemerintah memberikan perhatian penuh terutama kepada para kalangan akademisi untuk melakukan penelitian, seperti dukungan dana serta lomba-lomba keilmiahan. Kegiatan ekstrakurikuler keilmiahan juga dikembangkan mulai pendidikan tingkat menengah hingga perguruan tinggi. Sebagai salah satu perguruan tinggi di Indonesia, Institut Teknologi Sepuluh Nopember (ITS) Surabaya dengan para peneliti yang ada di dalamnya aktif memberikan kontribusi terhadap dunia penelitian Indonesia melalui publikasi jurnal dan seminar penelitian secara rutin setiap tahunnya. [1] Para peneliti dalam kapasitasnya sebagai penyedia iptek harus turut serta berperan dalam inovasi nasional. Kegiatan penelitian/riset selama ini sering terjadi antara satu dengan lain tidak ada keterkaitan. Perlu diusahakan agar kegiatan penelitian dapat dilakukan secara holistik, lebih fokus, lebih kontekstual dan ada kerjasama antar-peneliti dalam penentuan topik penelitian. [2] Selain itu, menurut survey yang dilakukan Menteri Riset, Teknologi dan Pendidikan Tinggi (Menristekdikti) jumlah publikasi yang ditelurkan peniliti Indonesia masih sangat sedikit. Pada survey yang dilakukan per Maret 2016, tercatat hanya 4.500 hingga 5.500 karya yang berhasil dipublikasikan. Jumlah tersebut tergolong kecil jika dibandingkan dengan jumlah penduduk Indonesia yang mencapai 250 juta jiwa.

Saat ini ITS sudah memiliki Sistem Repositori Peneliti. Sistem Repositori Peneliti merupakan sistem informasi yang secara khusus memberikan informasi kepada masyarakat seputar dunia penelitian yang ada di ITS. Beberapa fitur yang terdapat dalam sistem tersebut yaitu pengguna dapat melakukan pencarian peneliti dengan kriteria tertentu, melihat daftar publikasi jurnal penelitian terakhir, serta fitur lainya. Pada sistem informasi

tersebut pengguna dapat melakukan pencarian peneliti berdasarkan pengelompokan area peneliti (fakultas). Pada sistem informasi tersebut juga sudah memiliki visualisasi data kerjasama peneliti dalam bentuk graph yang menarik dan mudah dipahami. Namun, Sistem Repositori Peneliti ini belum memiliki visualisasi peta yang mampu menggambarkan keterkaitan topik antar disiplin ilmu. Oleh karena itu dalam tugas akhir ini akan dibuat sebuah modul yang akan menjadi bagian dari fitur Sistem Informasi Repositori Peneliti. Modul yang akan dibuat ini akan berfokus pada visualisasi peta keterkaitan topik antar disiplin ilmu. Dengan adanya visualisasi tersebut, keterkaitan antar disiplin ilmu dapat ditampilkan secara informatif dan menarik.

Data teks seringkali memiliki jumlah fitur yang sangat banyak, sedangkan pada Tugas Akhir ini Data Dokumen Penelitian ingin divisualisasikan ke dalam peta 2 dimensi. *Self-organizing Maps* (SOM) adalah salah satu tipe dari jaringan saraf tiruan yang dilatih menggunakan *unsupervised learning* untuk mendapatkan representasi data dengan dimensi yang sedikit [3]. SOM merupakan alat bantu yang efisien untuk merepresentasikan data dengan dimensi yang tinggi direduksi menjadi 2 dimensi saja [4]. SOM merupakan algoritma jaringan saraf tiruan yang umum digunakan untuk permasalahan ini.

Dalam tugas akhir ini akan digunakan data dokumen penelitian yang tersedia pada Sistem Repositori Peneliti ITS. Kemudian ditentukan *keyword* dari tiap dokumen tersebut yang akan dimasukkan ke dalam *vector-space model*. Kemudian, dari pemodelan tersebut dapat diketahui kemiripan antar dokumen dengan mengamati frekuensi munculnya kata-kata yang sama antar dokumen. Data *vector-space model* dokumen ini digunakan sebagai input untuk membangun *Self-organizing Maps*. Terakhir, dilakukan *K-means Clustering* dan *Hierarchical Clustering* terhadap map yang telah dibentuk untuk menyederhanakan visualisasi peta tersebut.

Harapan yang ingin dicapai dalam tugas akhir ini, para peneliti, mahasiswa ataupun masyarakat umum dapat mengetahui

keterkaitan topik antar disiplin ilmu sehingga dapat digunakan sebagai acuan dalam pembuatan penelitian di masa yang akan datang.

1.2 Rumusan Masalah

Tugas Akhir ini mengangkat beberapa rumusan masalah sebagai berikut:

1. Bagaimana melakukan pemilihan *keyword* dan memetakannya ke dalam matriks yang akan digunakan sebagai input data pada *Self-organizing Maps* (SOM)?
2. Bagainmana cara memvisualisasikan peta yang telah diperoleh dari hasil SOM agar dapat dengan mudah diinterpretasikan?

1.3 Batasan Permasalahan

Permasalahan yang dibahas pada Tugas Akhir ini memiliki batasan sebagai berikut:

1. Dataset yang digunakan adalah *corpus* penelitian dari *resits.its.ac.id*.
2. Jumlah data yang digunakan adalah 13.300 dokumen dari database *resits.its.ac.id*..
3. Metode diimplementasikan menggunakan Python.

1.4 Tujuan

Tujuan dari Tugas Akhir ini adalah sebagai berikut:

1. Mengimplementasikan algoritma *Term Frequency* untuk ekstraksi fitur dari dokumen penelitian.
2. Mengimplementasikan metode *Principal Component Analysis* (PCA) untuk mereduksi data hasil ekstraksi fitur.

3. Mengimplementasikan metode *Self-organizing Maps* (SOM) untuk merepresentasikan data hasil ekstraksi fitur ke dalam peta 2D
4. Mengimplementasikan *K-means Clustering dan Hierarchical Clustering* untuk menyederhanakan visualisasi peta SOM

1.5 Manfaat

Manfaat dari tugas akhir ini adalah mampu memberikan gambaran bagi peneliti Indonesia mengenai potensi Topik Penelitian yang dapat dikembangkan kedepannya melalui sebuah Peta Visualisasi Topik Penelitian yang menggambarkan topik-topik yang kerap diteliti beserta keterkaitannya.

1.6 Metodologi

Pembuatan Tugas Akhir ini dilakukan dengan menggunakan metodologi sebagai berikut:

1.6.1 Penyusunan Proposal Tugas Akhir

Tahapan awal dari Tugas Akhir ini adalah penyusunan Proposal Tugas Akhir. Proposal Tugas Akhir berisi pendahuluan, deskripsi dan gagasan metode – metode yang dibuat dalam Tugas Akhir ini. Pendahuluan ini terdiri atas hal yang menjadi latar belakang diajukannya Tugas Akhir, rumusan masalah yang diangkat, batasan masalah untuk Tugas Akhir, dan manfaat dari hasil pembuatan Tugas Akhir ini. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan Tugas Akhir. Terdapat pula sub bab jadwal kegitan yang menjelaskan jadwal pengerjaan Tugas Akhir.

1.6.2 Studi Literatur

Tugas Akhir ini menggunakan literatur *paper* yang berasal dari jurnal internasional bereputasi yaitu IEEE dan *Sciencedirect* untuk mencari informasi yang dapat dijadikan referensi dalam pengerjaan Tugas Akhir ini. Selain itu juga digunakan sejumlah referensi buku dan literatur lain yang berhubungan dengan ekstraksi fitur dari data dokumen dan metode – metode yang diusulkan pada Tugas Akhir ini berupa metode *Term Frequency-Inverse Document Frequency* (TF-IDF), *Self-organizing Maps* (SOM), *Principal Component Analysis* (PCA), *K-means Clustering, Hierarchical Clustering.*

1.6.3 Implementasi Perangkat Lunak

Implementasi merupakan tahap untuk membangun metode-metode yang sudah diajukan pada proposal Tugas Akhir. Untuk membangun algoritma yang telah dirancang sebelumnya, maka dilakukan implementasi dengan menggunakan suatu perangkat lunak. Pada tugas akhir ini, perangkat lunak yang digunakan adalah Python.

1.6.4 Pengujian dan Evaluasi

Pada tahap ini metode dari perangkat lunak yang telah disusun dan diimplementasikan diuji coba dengan menggunakan data teks *corpus* Penelitian Tugas Akhir Institut Teknologi Sepuluh Nopember. Setelah itu, hasil pengujian dievaluasi dengan menggunakan nilai *Silhouette Score* dan *Cophenet Correlation Coefficient*.

1.6.5 Penyusunan Buku

Pada tahap ini dilakukan penyusunan buku yang menjelaskan seluruh konsep, teori dasar dari metode yang

digunakan, implementasi, serta hasil yang telah dikerjakan sebagai dokumentasi dari pelaksanaan Tugas Akhir.

1.7 Sistematika Penulisan Laporan

Sistematika penulisan laporan Tugas Akhir adalah sebagai berikut:

1. Bab I. Pendahuluan
   Bab ini berisikan penjelasan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika penulisan dari pembuatan Tugas Akhir.
2. Bab II. Tinjauan Pustaka
   Bab ini berisi kajian teori dari metode dan algoritma yang digunakan dalam penyusunan Tugas Akhir ini. Secara garis besar, bab ini berisi tentang *Concept Map*, *Self-organizing Maps* (SOM), *Principal Component Analysis* (PCA), *K-means clustering*, dan *Hierarchical Clustering*..
3. Bab III. Perancangan Perangkat Lunak
   Bab ini berisi pembahasan mengenai perancangan dari metode *Self-organizing Maps* (SOM), *Principal Component Analysis* (PCA), *K-means clustering*, dan *Hierarchical Clustering* yang digunakan untuk memvisualisasikan dokumen penelitian ke dalam bentuk map.
4. Bab IV. Implementasi
   Bab ini menjelaskan implementasi yang berbentuk kode sumber dari proses pengelolaan *Self-organizing Maps* (SOM), *Principal Component Analysis* (PCA), *K-means clustering*, dan *Hierarchical Clustering* yang digunakan untuk memvisualisasikan dokumen penelitian ke dalam bentuk map.
5. Bab V. Hasil Uji Coba dan Evaluasi
   Bab ini berisikan hasil uji coba dari Ekstraksi Fitur, *Self-organizing Maps*, *K-means Clustering* dan *Hierarchical Clustering* yang digunakan untuk memvisualisasikan data

teks *corpus* Penelitian Tugas Akhir Institut Teknologi Sepuluh Nopember pada sistem.

6. Bab VI. Kesimpulan dan Saran
   Bab ini merupakan bab yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan, masalah-masalah yang dialami pada proses pengerjaan Tugas Akhir, dan saran untuk pengembangan solusi ke depannya.
7. Daftar Pustaka
   Bab ini berisi daftar pustaka yang dijadikan literatur dalam Tugas Akhir.
8. Lampiran
   Dalam lampiran terdapat tabel-tabel data hasil uji coba dan kode sumber program secara keseluruhan.

(Halaman sengaja dikosongkan)

BAB II
TINJAUAN PUSTAKA

Bab ini berisi pembahasan mengenai teori-teori dasar yang digunakan dalam Tugas Akhir. Teori-teori tersebut diantaranya adalah *Self-organizing Maps* (SOM), *Principal Component Analysis* (PCA), *K-means Clustering*, *Hierarchical Clustering* dan beberapa teori lain yang mendukung pembuatan Tugas Akhir.

`

2.1 Kartografi

Kartografi merupakan ilmu yang mempelajari bagaimana cara membuat peta dengan nilai estetika yang baik dan dicampur dengan seni bagaimana cara penyajian peta sehingga mudah dibaca dan dimengerti. [3] Kartografi sendiri seringkali digunakan untuk merepresentasikan Data *Spatial* dalam sebuah peta, namun pada jaman *modern* pendekatan Kartografi mulai dilirik untuk merepresentasikan Data *Non-Spatial* seperti Data Informasi *Textual*. [20] Representasi Kartografi untuk *Non-Spatial Data* disebut sebagai *Information Cartography*, dan pengaplikasian informasi ke dalam metode Kartografi pada dasarnya ingin mencapai 2 hal utama yakni:

1. Detail yang lebih banyak dapat disajikan ke dalam Peta Kartografi dibandingkan dengan representasi konvensional seperti graf ataupun *Concept Map*
2. Memanfaatkan proses kognitif manusia dalam menalar Peta Geografis ke dalam *Information Cartography* dengan memanfaatkan *landmark* dalam peta seperti wilayah dan tempat.

Langkah pertama untuk mengaplikasikan informasi Data Tekstual menjadi sebuah peta kartografi adalah dengan membentuk Peta Dasar (*Base Map*) yang berisikan dokumen-dokumen dari *corpus* yang digunakan. Setelah itu, diberikan detil-detil informasi yang diingkan terhadap *Base Map* yang telah terbentuk. Proses

pembuatan *Informational Cartography* mirip dengan pembuatan Peta Tematik pada umumnya.

Gambar 2. 1 Contoh Information Cartography [3]

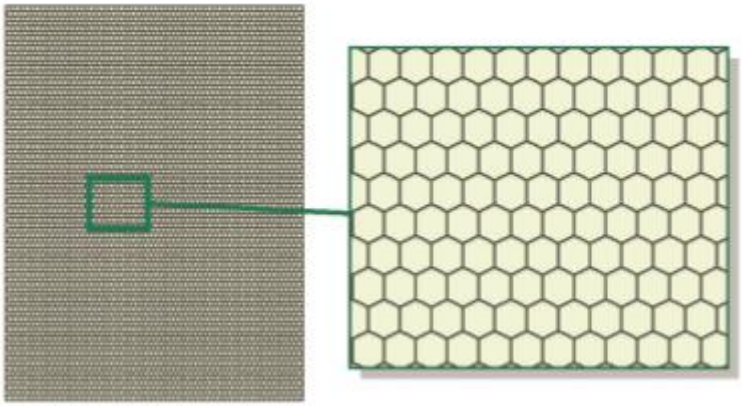
Gambar 2.2 diatas merupakan contoh penggunaan *Information Cartography*. Tanda batas warna hijau menunjukkan batas level pertama dari topik-topik yang terkelompok dalam 1 *cluster* beserta dengan label yang menunjukkan *cluster* tersebut berwarna hijau. Sedangkan garis batas dan label yang berwarna merah adalah garis batas dan label dari *cluster* yang dihasilkan dari gabungan *cluster* pada level 1.

2.2 *Self-organizing Maps* (SOM)

Self-organizing Maps(SOM) adalah salah satu bentuk metode yang sering digunakan dari jaringan saraf tiruan. SOM merupakan salah satu bentuk metode *unsupervised learning*, dimana proses *training* tidak membutuhkan label awal daripada data yang digunakan. Tujuan utama dari jaringan saraf tiruan ini adalah merepresentasikan data dengan dimensi yang besar ke dalam bentuk 2D agar mudah divisualisasikan [10]. Input dari teknik ini berupa hasil observasi dengan n-dimensi. Layer output terdiri dari sebuah jaringan yang berisikan noda-noda yang memiliki nilai *weight* masing-masing, disusun dalam kisi 2D [3]. *Weight* pada noda-noda ini akan terus berubah saat fase *training* dijalankan agar menyerupai data yang diinputkan kedalam jaringan [10]. Noda-noda dihubungkan ke tetangganya umumnya membentuk rancangan kisi persegi ataupun rancangan kisi *hexagonal*. Berbeda dengan metode jaringan saraf tiruan lainnya, SOM tidak memiliki *hidden layer*. Proses training SOM dapat memakan waktu yang panjang jika dimensi dari input terlalu besar. Pembuatan *Base Map* pada *Information Cartography* dapat dicapai dengan menggunakan algortima *Self-organizing Maps* (SOM). SOM dinilai cocok untuk merepresentasikan data secara kartografik dikarenakan struktur dari SOM sendiri sangat mirip dengan Peta pada umumnya.

Hasil *training* dari SOM dapat digunakan dalam beberapa cara, sebagai contoh hasil ini dapat di digunakan untuk memvisualisasikan tiap *input vector* individu, untuk mengetahui area topik yang paling relevan terhadap *vector* tersebut. Bentuk visualisasi yang populer adalah metode *U-matrix* yang mengkomputasi keterkaitan antar *vector*. Hasil dari perhitungan ini dapat direpresentasikan sebagai *cluster*. Observasi data baru dapat dengan mudah dimasukkan ke dalam SOM yang telah di *training*, karena hanya perlu mencari *vector* yang paling

signifikan nilainya terhadap observasi tersebut dan mengupdate nilai dari neuron pada jaringan saraf tiruan yang telah terbentuk.



Gambar 2. 2 Self-organizing Maps berukuran 60x70 neurons [3]

2.3 K-Means

K-Means adalah salah satu algoritma klastering (pengelompokkan) yang paling populer. Algoritma ini mengelompokkan objek ke beberapa klaster, dimana jumlah klaster telah ditentukan sebelumnya. *K-Means* dimulai dengan pemilihan *centroid* (titik tengah) secara random, satu untuk masing-masing klaster. Setelah itu, masing-masing data yang ingin dikelompokkan dihitung jaraknya (*disimilarity*) dengan masing-masing *centroid.* Perhitungan *disimilarity* dilakukan menggunakan *Euclidian Distance* dengan rumus [3]:

$$d(P,Q) = \sqrt{\sum_{j=1}^{n} (Xj(P) - Xj(Q))^2} \quad (2.1)$$

Pada rumus 2.1, d(P,Q) adalah jarak antara titik P pada *cluster* Q, Xj(P) adalah koordinat untuk titik P ke j dan Xj(Q) adalah koordinat dari *centroid* Q. Setelah dilakukan perhitungan *disimilarity*, data tersebut akan dikelompokkan ke dalam klaster yang mempunya nilai jarak paling kecil. Iterasi terus dilakukan sampai *stopping criteria* terpenuhi. Dalam hal ini, *stopping criteria* yang digunakan adalah data dalam sebuah klaster tidak berubah-ubah lagi. *Pseudocode* algoritma *K-Means* ditunjukkan pada **Error! Reference source not found.**2.

Masukan : K : Jumlah klaster
D : Dataset yang terdiri dari n objek
Keluaran : Beberapa klaster yang masing-masing berisi beberapa data
Metode :
(1). Pilih secara acak k objek sebagai *centroid.*
(2). *do*
(3). Masukkan masing-masing objek ke klaster yang mempunyai nilai disimilaritas paling rendah (jarak paling kecil) berdasarkan rata-rata klaster.
(4). Perbarui terus nilai rata-rata klaster
(5). *until* data dalam klaster tidak berubah

Gambar 2. 3 Pseudocode Algoritma K-means

2.4 *Hierarchical Clustering*

. *Hierarchical Clustering* adalah sebuah metode untuk analisis *cluster* yang bertujuan untuk membangun sebuah hierarki dari *cluster* yang telah terbentuk. Adapun 2 strategi utama dalam pembangungan *Hierarchical Clustering*:

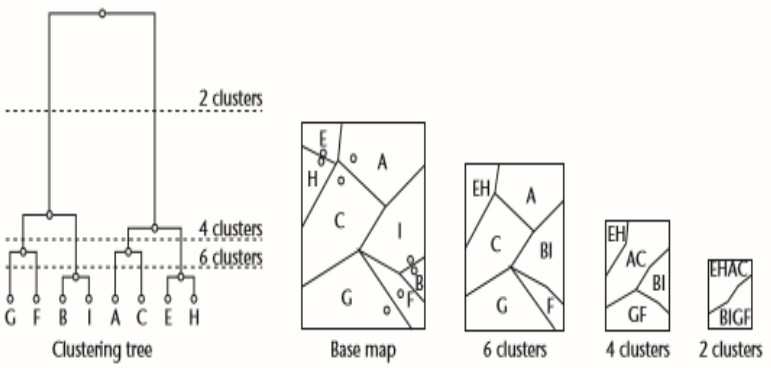
1. *Agglomerative*: pendekatan solusi '*bottom-up',* tiap *cluster* awalnya adalah *cluster* individual, lalu akan digabungkan seiring naiknya tingkat hierarki.
2. *Divisive*: pendekatan solusi '*top-down',* semua cluster awalnya meruapakan satu kesatuan, lalu akan dipecah seiring turunnya tingkat hierarki.

Pemilihan metode *Hierarchical Clustering* untuk penggabungan topik-topik yang telah terbentuk pada tahapan *SOM* dalam Tugas Akhir ini dirasa sangat cocok untuk menyederhanakan representasi visual [3]. Sudah banyak metode yang diakui dan digunakan untuk mengkomputasi *cluster* [6][7], namun *Hierarchical Clustering* mudah dibangun, Menggabungkan metode *clustering* ini ke dalam Map 2D dari SOM dapat membantu kontrol dari detail yang ingin ditampilkan pada level *zoom* tertentu.

Hierarchical Clustering memiliki 4 macam *linkage* [16] untuk menghitung kriteria penggabungan antar *cluster* antara lain:

1. *Single Linkage*: kriteria penentuan *cluster* yang digabungkan tiap iterasi, berdasarkan pada jarak member terdekat dari kedua *cluster* (*minimum pairwise distance)*
2. *Complete Linkage*: kriteria penentuan *cluster* yang digabungkan tiap iterasi, berdasarkan pada jarak member terjauh dari kedua *cluster*
3. *Average Linkage*: kriteria penentuan *cluster* yang digabungkan tiap iterasi, berdasarkan pada jarak rata-rata antar semua member dari kedua *cluster* tersebut.
4. *Ward's Method*: kriteria penentuan *cluster* yang menggabungkan *cluster* berdasarkan penambahan nilai

variance dari *cluster* yang terbentuk paling minimal. Setiap *cluster* dianggap sebagai *singletons* dan biasa direpresentasikan dengan *centroid* nya. Metode ini cocok digunakan untuk data berdistribusi *Multivariate Normal*.

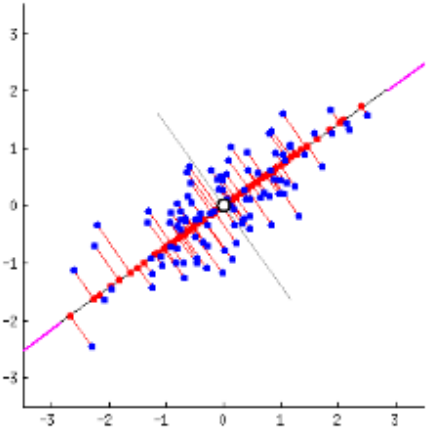


Gambar 2. 4 Penggunaan Hierarchical Clustering untuk menyederhanakan representasi map [3]

:

2.5 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) adalah salah satu teknik reduksi dimensi yang merubah dataset dengan dimensi besar (dimensi - *m*) ke dimensi rendah ruang fitur ortogonal (dimensi - *n*, dimana *m* > *n*) akan tetapi tetap mempertahankan informasi dari dimensi besar dataset yang asli. Setiap fitur ortogonal yang dihasilkan disebut sebagai "*Principal Component´* atau PC. PCA memakai dekomposisi dengan *Eigenvectors* dan *Eigenvalue*s. *Eigenvectors* yang memiliki *Eigenvalue*s terbesar adalah vektor yang searah dengan bidang yang memiliki korelasi tertinggi. Secara garis besar topologi PCA dapat dilihat pada Gambar 2.4.



Gambar 2. 5 Topologi PCA [8]

Tahap pertama yang dilakukan dalam PCA adalah membuat matriks varians-kovarians pada diagonal yang mengandung nilai varians dan sisi lainnya mengandung nilai kovarians. Matriks varians-kovarians mempunyai dimensi *n* x *n* dimana *n* adalah jumlah atribut/dimensi dari data, X_i dan Y_i adalah data dan $\bar{X}$ dan $\bar{Y}$ adalah rata – rata dari masing – masing atribut data. Rumus

mencari matriks varians-kovarians dapat dilihat pada persamaan 2.2 dan 2.3 [8].

$$var(x) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)} \quad (2.2)$$

$$covar(x,y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X}) (Y_i - \bar{Y})}{(n-1)} \quad (2.3)$$

dimana X_i adalah nilai X ke i, $\bar{X}$ adalah rata-rata nilai X, Y_i adalah nilai Y ke i, $\bar{Y}$ adalah rata-rata nilai Y, dan n adalah jumlah data.

Selanjutnya, PCA akan mencari *eigenvector* dan *eigenvalues* dari setiap matriks kovarians. Untuk mencari eigenvector dapat dilihat pada persamaan 2.4.

$$\Sigma X = \lambda X \quad (2.4)$$

dimana Σ adalah matriks kovarians dan X adalah vector bukan nol yang memenuhi persamaan tersebut. Sedangkan *Eigenvalue* adalah representasi skalar dari tingkatan varian dalam PC yang sesuai. Untuk mencari *eigenvalues* dapat dilihat pada persamaan 2.5.

$$\Sigma v = \lambda v \quad (2.5)$$

Dimana Σ adalah kovarians matriks, λ adalah *eigenvalue* dan v adalah eigen vektor.

Tahapan selanjutnya PC akan diurutkan berdasarkan tingkat kesesuaian *eigenvalue* sehingga PC yang paling pertama mempresentasikan varian dataset yang paling signifikan. Proses pereduksian dimensi dilakukan dengan mentransformasikan data

awal ke set variabel yang baru (PC) yang tidak berkorelasi satu sama lain. Dalam kata lain, jumlah PC akan selalu lebih kecil atau sama dengan jumlah variabel awal [17].

2.6 Silhouette Score

Silhouette Score merupakan sebuah metode interpretasi dan validasi teknik *Clustering. Silhouette score* merupakan nilai pengukuran dari seberapa mirip suatu data terhadap *cluster*-nya (*Cohesion*) dibandingkan dengan *cluster* anggota lain (*Separation*) [18]. *Silhouette Score* memiliki rentang nilai dari -1 sampai dengan 1, dimana nilai yang semakin besar menandakan bahwa suatu data semakin mirip dengan *cluster*-nya dan berbeda jauh dengan *cluster* anggota yang lain. Jika banyak *cluster* yang terbentuk memiliki nilai *Silhouette Score* yang tinggi, dapat disimpulkan bahwa jumlah *cluster* yang dipilih untuk data *input* sudah tepat.

Silhouette Score dapat dihitung dengan rumus berikut:

$$s(i) = \frac{\mathrm{b(i)} - a(i)}{\max\{\mathrm{a(i)}, \mathrm{b(i)}\}} \qquad (2.6)$$

s(i) merupakan *Silhouette Score Cluster* ke *i*, *b(i)* ialah nilai rata-rata ketidakmiripan terendah *cluster* ke *i* dengan *cluster* lain atau biasa disebut *neighbouring cluster*, *a(i)* ialah nilai rata-rata ketidakmiripan dari semua data yang ada pada *cluster* ke *i*. Nilai rata-rata dari semua *Silhouette Score* untuk semua *cluster* dapat menggambarkan seberapa *cohesive* dan *separable cluster* yang terbentuk.

2.7 Cophenetic Correlation Coefficient

Cophenet Correlation Coefficient adalah sebuah metode untuk memvalidasi algoritma *Hierarchical Clustering. Cophenet*

Correlation Coefficient menghitung seberapa mirip *cluster-cluster* yang digabungkan pada algoritma *Hierarchical Clustering*. Metode ini juga memvalidasi hasil *dendogram* yang dihasilkan *Hierarchical Clustering* sudah merepresentasikan dengan baik jarak asli dari *cluster* yang digabungkan [19]. Sejak diperkenalkan oleh Sokal dan Rohif, metode *Cophenet Correlation Coefficient* umum digunakan untuk mengevaluasi efisiensi Teknik *Clustering* yang digunakan. *Cophenet Correlation Coefficient* memiliki *range* dari 0 sampai 1, dengan nilai yang lebih besar menyatakan bahwa penggabungan *Cluster-Cluster* sudah efisien.

Cophenet Correlation Coefficient dapat dihitung dengan rumus sebagai berikut:

$$c = \frac{\sum i < j(x(i,j) - \bar{x})(t(i,j) - \bar{t})}{\sqrt{(\sum i < j(x(i,j) - \bar{x})^2)(\sum i < j\,(t(i,j) - \bar{t})^2)}} \tag{2.7}$$

c adalah nilai *Cophenet Correlation Coefficient, x(i,j)* adalah *Ecludiean Distance* antara *cluster* ke *i* dan *j*, *t(i,j)* adalah *dendogrammic distance* dari *cluster i* dan *j* yang merupakan ketinggian titik dimana kedua *cluster* ini digabungkan pada *Hierarchical Cluster*, $\bar{x}$ adalah rata-rata dari *x(i,j)* dan $\bar{t}$ adalah rata-rata dari *t(i,j)*.

(Halaman sengaja dikosongkan)

BAB III
PERANCANGAN SISTEM

Bab ini menjelaskan tentang perancangan dan pembuatan sistem perangkat lunak. Sistem perangkat lunak yang dibuat pada Tugas Akhir ini adalah mengolah data *text* penelitian, menggunakan *corpus* Tugas Akhir dari Institut Teknologi Sepuluh Nopember. Dilakukan praproses terlebih dahulu pada data dengan *stopword removal, punctuation removal,* dan tokenisasi. *Keyword Indexing* digunakan terhadap judul dari corpus, yang selanjutnya *keyword* tersebut digunakan sebagai fitur tiap dokumen. TF-IDF digunakan sebagai metode ekstraksi fitur dari data yang telah melalui praproses, proses tersebut menghasilkan matriks dokumen-term yang selanjutnya direduksi fiturnya menggunakan *Principal Component Analysis* (PCA) dan dipetakan ke dalam jaringan peta *Self-organizing Maps* (SOM). *K-means Clustering* dan *Hierarchical Clustering* diaplikasikan untuk menyederhanakan jaringan peta yang telah terbentuk. Untuk penentuan label digunakan metode TF untuk setiap hasil *cluster.* Pada bab ini pula akan dijelaskan gambaran umum sistem dalam bentuk *flowchart.*

3.1 Data

Pada sub bab ini akan dijelaskan mengenai data yang digunakan sebagai masukan perangkat lunak untuk selanjutnya diolah dan dilakukan pengujian sehingga menghasilkan data keluaran yang diharapkan.

3.1.1 Data Masukan

Data masukan adalah data yang digunakan sebagai masukan awal dari sistem. Data yang digunakan dalam perangkat lunak Visualisasi Similaritas Topik Penelitian menggunakan *Self-organizing Maps* (SOM) adalah data *repository* Tugas Akhir

Institut Teknologi Sepuluh Nopember (ITS). *Corpus* ini berisi 13.324 Data Tugas Akhir dari 5 fakultas yang ada di ITS dengan rincian 2194 Data dari Fakultas FMIPA, 5360 Data dari Fakultas FTI, 3078 Data dari Fakultas FTSP, 1156 Data dari Fakultas FTK, dan 1250 Data dari Fakultas FTIF.

Tabel 3.1 Jumlah Corpus dipisahkan berdasarkan Kode Fakultas

KODE FAKULTAS	NAMA FAKULTAS	JUMLAH
1	FMIPA	2194
2	FTI	5360
3	FTSP	3078
4	FTK	1156
5	FTIF	1250

Pada Tugas Akhir ini akan digunakan Judul, dan Abstrak dari setiap data Penelitian Tugas Akhir. Digunakan 3 skenario penggunaan data yakni:

1. Data Tugas Akhir seluruh ITS sejumlah 13.324 Data
2. Data Tugas Akhir Fakultas FTSP Sejumlah 1.156 Data
3. Data Tugas Akhir Jurusan Teknik Informatika sejumlah 825 Data.

NRP	JUDUL	ABSTRAK
1107100060	Desain Dan Fabrikasi Sumber Arus Terkendali Secara Realtime Pada Generator Gas HHO	Telah dirancang dan dibuat sebuah alat yang mampu memberikan daya untuk mencatu arus listrik konstan pada peralatan generator gas
1109100013	RANCANG BANGUN ALAT PEMILAH BAWANG MERAH BERDASARKAN UKURAN	Telah dilakukan perancangan dan pembuatan alat pemilah bawang merah berdasarkan ukuran diameternya. Alat ini memanfaatkan

Gambar 3.1 Contoh Data Masukan

Seperti yang dapat dilihat pada Gambar 3.2, kolom pertama adalah NRP Mahasiswa dari data Tugas Akhir tersebut, kolom kedua adalah Judul dari data Tugas Akhir tersebut, dan kolom ketiga adalah abstrak dari data Tugas Akhir tersebut.

3.1.2 Data Keluaran

Data masukan akan dilakukan praproses terlebih dahulu untuk menghilangkan kata-kata *stopwords* maupun tanda baca dan dilakukan tokenisasi. Judul dari data masukan akan digunakan sebagai kata kunci/fitur dari tiap dokumen. Abstrak dari data yang telah melalui praproses diproses menggunakan *Term Frequency-Inverse Document Frequency* (TF-IDF), *Principal Component Analysis* (PCA), *Self-organizing Maps* (SOM), *K-means Clustering,* dan *Hierarchical Clustering*. Selanjutnya, program yang dikembangkan dapat mengeluarkan hasil berupa Visualisasi Peta Topik dari data masukan, juga S*ilhouette Score* dan *Cophenet Correlation Coefficient* sebagai validasi kelayakan *cluster* yang telah terbentuk. Untuk setiap tahap akan didapatkan hasil output sebagai berikut:

1. Praproses: Data Teks yang telah dihilangkan *stopwords* nya dan telah ditokenisasi.
2. Ekstraksi Fitur: Matriks *Term-Documents* beserta nilai TF-IDF.
3. SOM: Jaringan Peta *Neuron* yang telah terlatih dengan Data Input. Tiap *Neuron* pada Jaringan ini berisikan dokumen-dokumen yang terasosiasi paling dekat dengan *neuron* tersebut.
4. *K-means* dan *Hierarchical Clustering*: Jaringan Peta *Neuron* yang telah terkelompok berdasarkan similaritas beserta *Silhouette* dan *Cophenet Score.* Proses ini juga menghasilkan label dari tiap *cluster*.
5. Visualisasi: Peta Similaritas Topik Penelitian.

Contoh hasil *output* untuk tiap proses akan dibahas pada sub-bab berikutnya.

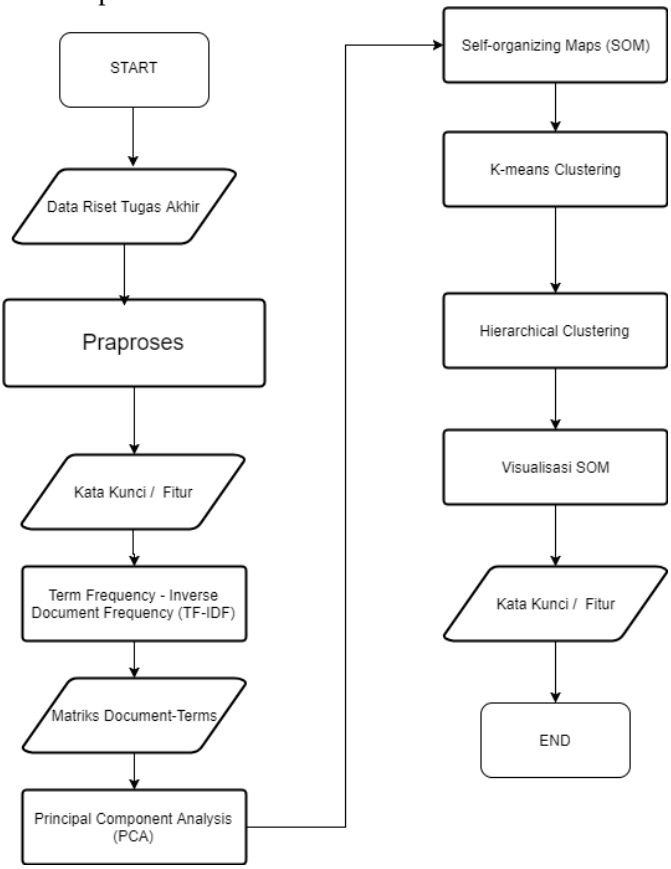
3.2 Desain Umum Sistem

Rancang bangun perangkat lunak Visualisasi Topik Penelitian akan dibangun menggunakan metode *Term Frequency-Inverse Document Frequency* (TF-IDF), *Principal Component Analysis* (PCA), *Self-organizing maps* (SOM), *K-means Clustering*, dan *Hierarchical Clustering.* Proses pertama yang dilakukan sistem yaitu melakukan praproses berupa *stopwords removal* dan tokenisasi.

Tahapan selanjutnya dari praproses adalah melakukan penentuan kata kunci/fitur yang digunakan (*Keyword Indexing*), dimana pada tugas akhir ini, kata kunci diambil dari judul pada *corpus* yang digunakan. Setelah itu matriks *document-term*s dibentuk melalui perhitungan TF-IDF setiap dokumen. Sebelum matriks *document-terms* digunakan untuk proses *training*, dilakukan reduksi dimensi data menggunakan *Principal Component Analysis* (PCA). Tujuan utama dari PCA yaitu untuk memperingan kinerja mesin *training* terutama dalam hal waktu komputasi. Cara kerja PCA yaitu mentransformasi fitur ke dimensi lain dengan menghitung tingkat kepentingan informasi dari data yang akan diambil berdasarkan nilai *eigen value* data tersebut.

Hasil dari PCA yaitu data hasil ekstraksi fitur dengan dimensi data yang baru. Data tersebut selanjutnya dianggap sudah siap untuk masuk ke proses *training* menggunakan metode jaringan *Kohonen* SOM. Selanjutnya dilakukan kombinasi *K-means Clustering* dan *Hierarchical Clustering* guna menyederhanakan visualisasi jaringan peta SOM yang terbentuk. *Silhouette Scoring* digunakan untuk menentukan jumlah *cluster* yang optimal untuk digunakan pada metode *K*-means, sedangkan *Cophenet Correlation Coefficient* berguna sebagai validator performa *Hierarchical Clustering.* Setelah proses *clustering*, ditentukan label dari tiap *cluster* menggunakan pembobotan *Term*

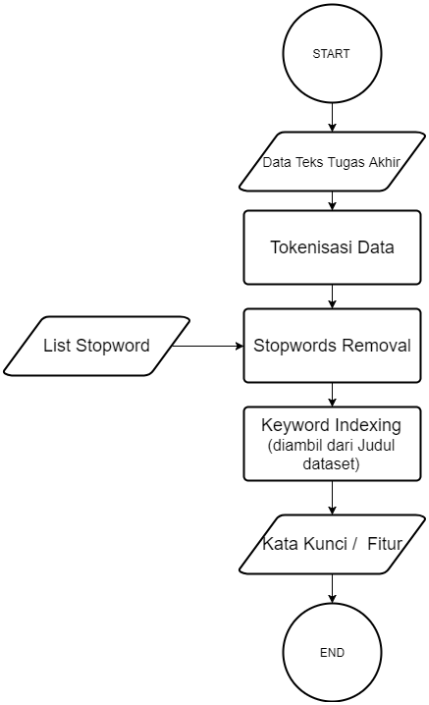
Frequency dan diambil 3 *terms* dengan nilai tertinggi. Terakhir, jaringan peta yang telah disederhanakan divisualisasikan agar mempermudah interpretasi kedekatan topik dari Data Tugas Akhir yang digunakan. Secara umum, proses dari sistem Visualisasi Topik Penelitian menggunakan *Self-organizing Maps* dan Kombinasi *K-means Clustering* dan *Hierarchical Clustering* dapat dilihat pada Gambar 3.3.



Gambar 3.2 Diagram Alur Keseluruhan Sistem

3.2.1 Praproses

Praproses adalah suatu proses/langkah yang dilakukan untuk membuat data mentah menjadi data yang berkualitas (input yang baik untuk proses data mining). Secara garis besar, proses *preprocessing* pada Tugas Akhir ini dapat dilihat pada gambar 3.4 di bawah.



Gambar 3.3 Langkah Praproses Data Tugas Akhir

Langkah 1. Proses pertama yang dilakukan adalah mengelola data mentah yang didapat dari *repository* riset Tugas Akhir milik Institut Teknologi Sepuluh Nopember menjadi data yang dapat

digunakan pada sistem Visualisasi Topik Penelitian menggunakan *Self-organizing Maps* (SOM) dan Kombinasi *K-means Clustering* dan *Hierarchical Clustering*. Data *repository* ini sudah tersimpan dalam *database* dan berformat SQL. Langkah-langkah untuk mengambil data tersebut ke dalam Sistem adalah sebagai berikut:

A) Gunakan *library SQL Connector for Python* untuk menghubungkan *Python* kepada *database MySQL*.
B) Lakukan *query* untuk mengambil kolom Judul dan Abstrak dari data yang tersimpan.
C) Inisialisasi sebuah *dictionary* untuk meyimpan indeks dokumen, judul, dan abstrak dari dokumen tersebut.
D) Simpan semua data yang didapat dari *query SQL* ke dalam *dictionary* yang telah diinisialisasi.

Hasil dari langkah ini berupa *dictionary* yang berisikan setiap judul dan abstrak dokumen penelitian yang ada dapat dilihat pada Gambar 3.2.

Langkah 2. Proses berikutnya adalah Judul dan Abstrak dari data yang didapat pada langkah 1 ditokenisasi guna mempermudah proses berikutnya. Tokenisasi adalah proses untuk memisahkan sebuah kalimat menjadi bagian unit yang lebih kecil atau kata-kata. Hasil dari Langkah berikut adalah kumpulan kata-kata dari kalimat pada Judul dan Abstrak.

$$Token_{\boldsymbol{judul}} = \begin{bmatrix} "metode" \\ "segmentasi" \\ "sebagai" \\ "dan" \\ "jaringan" \\ "konstruksi" \end{bmatrix}$$

$$Token_{\boldsymbol{abstrak}} = \begin{bmatrix} "metode" \\ "citra" \\ "analisis" \\ "berdasarkan" \\ "dan" \\ "manet" \end{bmatrix}$$

Langkah 3. Proses berikutnya yaitu melakukan *stopwords removal* untuk membersihkan data dari kata-kata yang menjadi noise pada proses *training.* Daftar kata-kata yang termasuk *stopwords* merupakan kata-kata penghubung yang diambil dari Kamus Besar Bahasa Indonesia dan penambahan kata-kata manual yang sering diamati muncul dan tidak bermakna pada *corpus* yang digunakan. Kata-kata yang telah ditokenisasi dicek apakah termasuk dalam daftar *stopwords*, jika kata tersebut meruapakan *stopwords*, maka kata tersebut akan dihilangkan dan tidak dipakai. Hasil dari langkah ini adalah sebagai berikut.

$$Token_{\boldsymbol{judul}} = \begin{bmatrix} "metode" \\ "segmentasi" \\ "jaringan" \\ "konstruksi" \end{bmatrix}$$

$$Token_{\boldsymbol{abstrak}} = \begin{bmatrix} "metode" \\ "citra" \\ "analisis" \\ "manet" \end{bmatrix}$$

Langkah 4. Terakhir dilakukan *keyword indexing* atau penentuan kata kunci yang akan digunakan sebagai fitur dari tiap data dokumen *corpus.* Pada penelitian kali ini, kata kunci diambil dari data judul tiap dokumen yang telah dibersihkan dan ditokenisasi pada proses sebelumnya. Kata kunci disimpan dalam sebuah *dictionary array* yang akan digunakan pada proses berikutnya.

3.2.2 Term Frequency – Inverse Document Frequency (TF-IDF)

Term Frequency – Inverse Document Frequency (TF-IDF) merupakan salah satu metode pembobotan fitur yang umum digunakan dalam mengolah data teks [15]. Pada dasarnya, TF-IDF menghitung frekuensi suatu kata muncul dalam satu dokumen dibandingkan dengan proporsi *inverse* dari frekuensi kata tersebut muncul pada *corpus* dokumen keseluruhan. Dengan

pembobotan ini, kita dapat menghitung seberapa relevan kata tersebut dengan dokumen terkait. Kata-kata yang sering muncul pada sedikit dokumen akan memiliki nilai yang lebih tinggi dibandingkan dengan kata-kata yang sering muncul pada banyak dokumen.

Jika terdapat *corpus* dokumen D, sebuah kata w, dan sebuah dokumen d ϵ D, rumus TF-IDF adalah sebagai berikut

$$wd = fw,d * log\ (|D|/fw,D) \tag{3.1}$$

dimana, wd adalah bobot kata *w* pada dokumen *d*, fw,d adalah frekuensi kemunculan kata *w* pada dokumen *d*, $|D|$ adalah jumlah total dokumen pada *corpus D*, dan fw,D adalah frekuensi kemunculan kata w pada *corpus D*. Langkah-langkah dalam proses perhitungan TF-IDF adalah sebagai berikut:

A) Untuk setiap term pada Token Abstrak untuk tiap dokumen dicek pada *list* kata kunci. Tiap *occurrence term* tersebut pada dokumen ke *i*, nilai TF *term* tersebut pada dokumen ke i ditambahkan.
B) Ulangi langkah A sampai semua dokumen dalam *corpus*.
C) Dilakukan perhitungan *Document Frequency* dengan cara mengecek kemunculan untuk tiap *term* yang ada pada *list* kata kunci muncul pada dokumen dalam *corpus.* Tiap *term* tersebut ditemukan pada dokumen ke *i*, nilai *Document Frequency* ditambahkan.
D) Mengubah hasil *Document Frequency* dari langkah C menjadi *Inverse Document Frequency* dengan rumus $log\ (|D|/fw,D)$ dimana *fw,D* adalah *Document Frequency* dari term *w* dan $|D|$ adalah jumlah dokumen pada *corpus D*.

Hasil dari tahap ini adalah sebagai berikut.

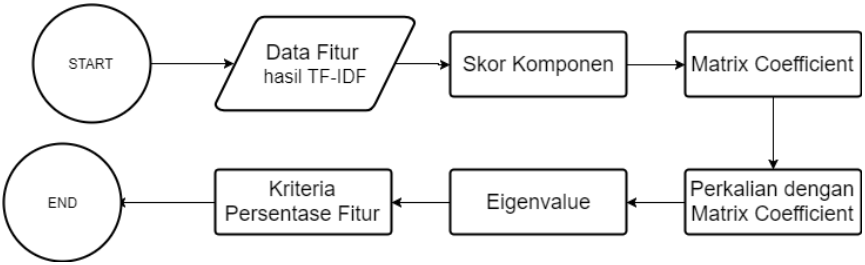
$$TF - IDF_{\boldsymbol{dokumen1}} = \begin{bmatrix} analisis, 2.47712 \\ jalan, 2.17609 \\ informasi, 0.17609 \end{bmatrix}$$

$$TF - IDF_{\boldsymbol{dokumen2}} = \begin{bmatrix} teknologi, 2.47712 \\ jalan, 0 \\ informasi, 3.17609 \end{bmatrix}$$

3.2.3 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) pada Tugas Akhir ini bertujuan untuk mereduksi dimensi dari data hasil ekstraksi fitur menggunakan TF-IDF pada proses sebelumnya. Reduksi dimensi bertujuan untuk memperingan kinerja proses *training* dan mengambil informasi yang relevan agar tidak mempengaruhi validasi data. Prinsip kerja PCA adalah dengan mentransformasikan fitur ke dimensi lain dengan menghitung tingkat kepentingan informasi yang diambil berdasarkan *Eigen Value* dari data tersebut.

Karena tujuan dari PCA adalah mereduksi dimensi fitur, maka output dari proses reduksi dimensi adalah dihasilkan komponen-komponen yang merepresntasikan fitur-fitur data sebelumnya dengan jumlah dimensi yang lebih rendah namun tetap mempunyai nilai kepentingan yang relatif sama. Secara garis besar, proses PCA pada Tugas Akhir ini dapat dilihat pada Gambar 3.5.



Gambar 3.4 Langkah Reduksi Dimensi PCA

Langkah-langkah dalam mengaplikasikan PCA terhadap Data Matriks *term-document* dari hasil TF-IDF adalah sebagai berikut:

A) *Load library* Python *sklearn.decomposisition* untuk implementasi PCA dalam Python.
B) Inisialisasi *object* PCA dengan *parameter* jumlah *Component* yang ingin dihasilkan. Jumlah *Component* yang *optimal* didapatkan melalui *trial & error* dan akan dibahas pada tahap Uji Coba.
C) Kenakan PCA terhadapt matriks *term-document* yang didapat dari proses Ekstraksi Fitur.
D) PCA akan mereduksi fitur-fitur dari data *input* menjadi *Principal Component* (PC) baru yang merepresentasikan Data asal.

Berikut adalah contoh hasil Data setelah direduksi fitur menggunakan PCA.

MATRIKS FITUR AWAL

$$TF-IDF_{\boldsymbol{dokumen1}} = \begin{bmatrix} analisis, 2.47712 \\ jalan, 2.17609 \\ \dots \\ informasi, 0.17609 \end{bmatrix}$$

$$TF-IDF_{\boldsymbol{dokumen2}} = \begin{bmatrix} teknologi, 2.47712 \\ jalan, 0 \\ \dots \\ informasi, 3.17609 \end{bmatrix}$$

MATRIKS FITUR SETELAH REDUKSI

$$PCA_{\boldsymbol{dokumen1}} = \begin{bmatrix} PC\ 1, 3.5672 \\ PC\ 2, 4.5481 \end{bmatrix}$$

$$PCA_{\boldsymbol{dokumen1}} = \begin{bmatrix} PC\ 1, 2.2459 \\ PC\ 2, 7.3412 \end{bmatrix}$$

Dapat dilihat pada contoh diatas bahwa data awal yang memiliki jumlah sebanyak N fitur dapat direduksi menjadi M fitur. Pada dasarnya, PCA melakukan rekonstruksi terhadap tiap fitur yang ada dan menghitung varians untuk tiap fitur. *Principal Component* dibentuk dari fitur-fitur yang memiliki nilai varians tertinggi pada Data Input Awal.

3.3 Self-organizing Maps (SOM)

Self-organizing Maps (SOM) adalah metode *unsupervised learning* yang digunakan pada Tugas Akhir ini. Tujuan utama penggunaan SOM adalah kemampuannya untuk merepresentasikan data dengan dimensi fitur yang besar menjadi 2 dimensi saja, sehingga lebih mudah divisualisasikan.

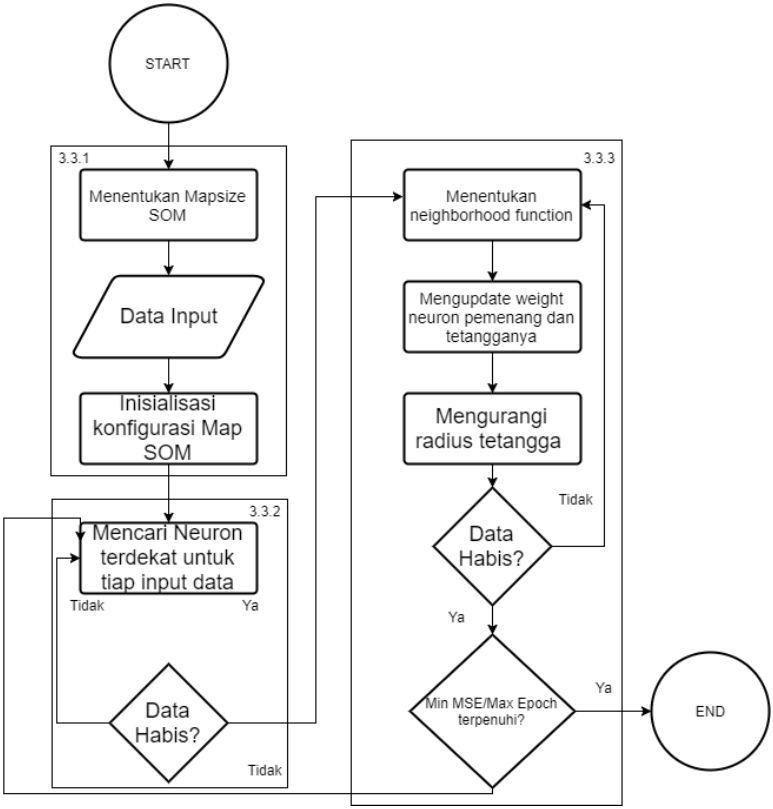
Secara garis besar, tahapan *training SOM* dibagi menjadi 2 tahap yaitu:

1. *Rough Training*: Fase *training* awal dimana *weight* dari tiap *neuron* dan juga nilai *Mean Square Error* (MSE) masih belum konvergen.
2. *Fine Tuning*: Fase *training* akhir dimana *weight* dari tiap *neuron* dan juga nilai *Mean Square Error* (MSE) sudah konvergen. Guna dari fase ini adalah menstabilkan bentuk jaringan yang sudah terbentuk.

Setiap proses *training SOM* dibagi kedalam 3 tahap [10] yaitu:

1. Inisialisasi
2. Sampling dan *Matching*
3. *Updating*

Alur kerja SOM secara garis besar dapat dilihat pada gambar 3.6.



Gambar 3.5 Diagram Alur Self-organizing Maps

3.3.1 Tahap Inisialisasi

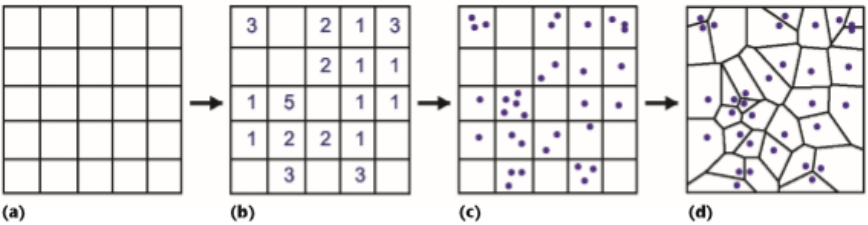
Pada tahap ini, dilakukan konfigurasi beberapa parameter penting dari SOM yang digunakan antara lain, *Map Size*, *Initialization Method*, dan *Training Method*.

Penentuan besar *Map Size* sangat dipengaruhi dengan jumlah data input. Jika input data besar, dan *Map Size* yang

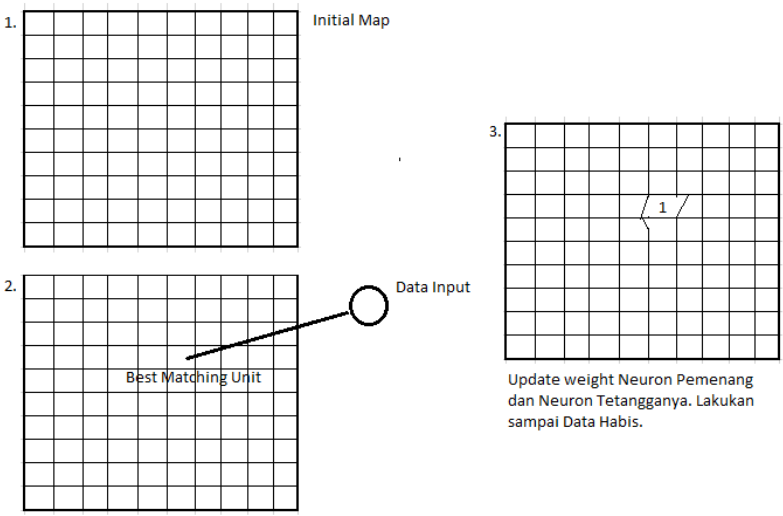
digunakan terlalu kecil, maka persebaran data akan tidak merata dan akurat diakibatkan nilai *weight* neuron yang sering berubah-ubah. Tiap Input Data pada jaringan SOM merepresentasikan 1 Data Dokumen Tugas Akhir. Pada Tugas Akhir ini, dengan jumlah 13.324 input data, *Map Size* yang digunakan adalah 50x50 *neuron* agar persebaran data lebih akurat.

Parameter kedua yang perlu diperhatikan dalam algoritma SOM adalah inisialisasi nilai *weight* dari *neuron-neuron* pada jaringan SOM. Metode inisialisasi yang umum digunakan adalah PCA, dan *Random.* Metode PCA menginisialisasi nilai *weight-weight neuron* proporsional dengan nilai *Eigenvectors* dari *Principal Component* pertama sampai N, dimana *Eigenvectors* ke-N masih memiliki *eigenvalue* yang serupa dari *Eigenvectors* dari *Principal Component* pertama. Sehingga metode PCA menghasilkan hasil yang deterministic dikarenakan *weight initial Neuron* selalu mengikuti *weight* dari *Eigenvector* yang paling merepresentasikan Data Input. Inisialisasi dengan menggunakan PCA juga mempercepat proses *training* diakibatkan karena konfigurasi jaringan awal yang lebih terorganisasi dengan baik [9]. Sedangkan inisialisasi menggunakan metode *Random* menghasilkan hasil yang stokastik dengan probabilitas n^k, dimana n adalah jumlah neuron [11] [12]. Maka dari itu pada Tugas Akhir ini digunakan inisialisasi menggunakan PCA.

Training Method pada SOM memiliki 2 opsi, yaitu *sequential* atau *batch*, dimana metode *sequential* mengupdate weight *neuron* setiap kali data dicari *Best Matching Unit* (BMU) nya. Sedangkan metode *batch,* mencari *Best Matching Unit* untuk semua data input terlebih dahulu sebelum mengupdate nilai *weight* tiap neuron, metode ini lebih cepat dalam hal komputasi tapi relative menghasilkan jaringan SOM yang lebih buruk dibandingkan metode *sequential* [14]. Maka dari itu, digunakan metode *sequential* agar mendapatkan hasil *training* yang terbaik.



Gambar 3.6 Metode Batch dimana Jaringan Neuron baru dilakukan Update ketika Data Input sudah habis



Gambar 3.7 Metode Sequential dimana Jaringan Neuron selalu dikenakan Update sampai Data Input sudah habis

3.3.2 Sampling dan Matching

Tahap Sampling dan *Matching* adalah tahap pencarian *neuron* pemenang atau biasa disebut *Best Matching Unit* (BMU)

dari tiap data input. Rumus jarak yang digunakan pada Tugas Akhir ini adalah *eucledian distance* dengan rumus sebagai berikut.

$$d(P,Q) = \sqrt{\sum_{j=1}^{n} (Xj(P) - Xj(Q))^2} \quad (3.2)$$

Dimana *Xj(P)* adalah fitur ke j dari Input Data *P*, dan *Xj(Q)* adalah fitur ke *j* dari Neuron *Q*.

3.3.3 Updating

Tahap Updating adalah tahap akhir dari *training* jaringan SOM, dimana untuk setiap data yang masuk, *weight* dari *neuron* pemenang dan *neuron* tetangganya diupdate. Nilai awal Radius Ketetanggan biasa diinisialisasi dengan rumus *MapSize/4* untuk *Map Size* dengan jumlah besar [14]. Nilai ketetanggan yang digunakan adalah *gaussian neighborhood* dengan rumus sebagai berikut: (3.3)

$$h_{ij}^{c} = \alpha(t).e^{\left(\frac{-\|R_c - R_{ij}\|^2}{2(\eta_{ij}^{c}(t))^2}\right)}$$

Dimana *α(t)* adalah *learning rate*, dimana pada tugas akhir ini digunakan *learning rate* 1.0, R_c dan R_{ij} adalah vector 2 dimensi yang berisikan indeks dari W_c dan W_{ij} dimana c adalah neuron pemenang dan *ij* adalah letak neuron yang dihitung dan η_{ij}^{c} adalah nilai radius dari kedua *neuron* yang dihitung, nilai ini berkurang seiring iterasi bertambah dan dihitung menggunakan rumus *exponential decay* sebagai berikut:

(3.4)

$$\eta_{ij}^{c}(t) = \eta(0).e^{\left(\frac{-t}{T}\right)}$$

Berikut langkah-langkah SOM secara urut:

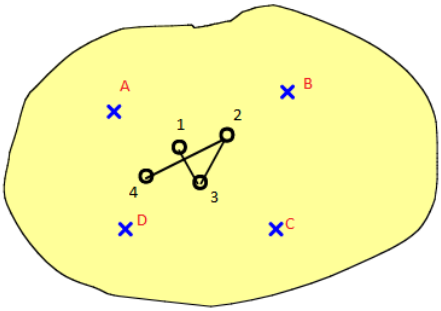
A) Dokumen dari resits.its.ac.id diambil kolom Judul dan Abstraknya.
B) Judul dari Dokumen ini akan digunakan sebagai kata kunci/fitur untuk tiap dokumen. Dimisalkan dari 4 Dokumen didapatkan 100 kata kunci/fitur.
C) Tiap Dokumen akan diekstraksi fitur menggunakan TF-IDF untuk diambil *occurrence* tiap *term* kata kunci pada dokumen tersebut. Melalui proses ini didapat matriks berukuran Jumlah Dokumen - Jumlah Kata Kunci, jika dicontohkan dengan 4 dokumen, maka didapatkan matriks berukuran 4x100.
D) Matriks tersebut direduksi fitur menggunakan PCA. Melalui proses PCA, matriks berukuran 4x100 dari hasil C direduksi dimensinya menjadi 4x20.
E) Matriks pada hasil D akan digunakan sebagai Data Input pada jaringan SOM. Inisialisasi Jaringan Peta *Neuron* dengan *size* 2x2. Maka terbentuk 4 *Neuron* dengan masing-masing *Neuron* memiliki *weight* berdimensi sejumlah fitur dari matriks input yaitu 20. Maka dari itu didapatkan 4 *neuron* dengan masing-masing memiliki 20 fitur. Inisialisasi *Weight Neuron* berdasarkan *Eigenvectors* dengan *Eigenvalue* tertinggi dari PCA Data Input. Radius Ketetanggan Awal dinisialisasi dengan nilai 2 dan *Learning Rate* 0.2.

$$\boldsymbol{Neuron1} = \begin{bmatrix} F1,10.2341 \\ \dots \\ F20,14.3561 \end{bmatrix}$$

$$\boldsymbol{Neuron2} = \begin{bmatrix} F1,5.8342 \\ \dots \\ F20,8.3421 \end{bmatrix}$$

$$\boldsymbol{Neuron3} = \begin{bmatrix} F1,1.7498 \\ \dots \\ F20,2.3921 \end{bmatrix}$$

$$\boldsymbol{Neuron4} = \begin{bmatrix} F1,4.4612 \\ \dots \\ F20,9.2842 \end{bmatrix}$$



Gambar 3.8 Visualisasi Jaringan Awal dengan lingkaran adalah Neuron dan Silang adalah Data Input

F) Dilakukan kalkulasi *Neuron Pemenang* untuk tiap Data Input dengan menghitung *Euclidean Distance* dari tiap fitur-fitur Data Input terhadap *weight* tiap *neuron.* Beikut perhitungan *Neuron Pemenang* terhadap Data Input Dokumen C

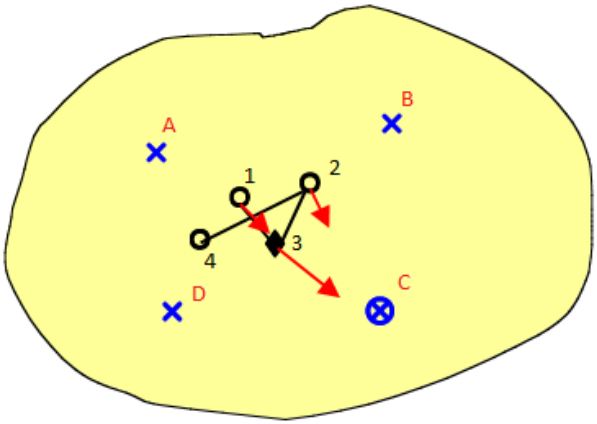
$$PCA_{\boldsymbol{dokumenC}} = \begin{bmatrix} PC\ 1, 3.5672 \\ \dots \\ PC\ 20, 4.5481 \end{bmatrix}$$

$$d(P,Q) = \sqrt{\sum_{j=1}^{n} (FiturDataInput(j) - FiturNeuron(j))^2}$$

$$d(P,Q) = \sqrt{(3.5672 - 1.7498)^2 + (4.5481 - 2.3921)^2}$$

$$d(P, Q) = 2.963$$

Didapatkan *Neuron Pemenang* untuk Data Input C adalah *Neuron C* dengan jarak 2.963. Maka *Weight* dari *Neuron Pemenang* dan Tetangganya *diupdate*. Karena Radius ketetanggan saat ini adalah 2, maka hanya *neuron* 1 dan 2 yang dianggap tetangga dari *neuron* 3.



Gambar 3.9 Visualisasi Jaringan SOM setelah Data Input Pertama dimasukkan. Tanda Diamond menandakan Neuron Pemenang. Neuron Pemenang dan Tetangganya diupdate mendekati Data Input

G) *Weight* dari *neuron* pemenang dan tetangganya akan di*update* berdasarkan nilai fitur dari Data Input pada saat itu. Berikut rumus *update weight* dari *neuron.*

$$WeightNeuron(i)(j) = \partial FiturDokumen(x)(j) + WeightNeuron(i)(j)$$

$$WeightNeuron(i)(j) = 0.2 * 3.5762 + 1.7498$$

$$WeightNeuron(i)(j) = 2.46504$$

Dimana ∂ adalah *learning rate*, i adalah indeks dari *neuron* pemenang, j adalah indeks fitur dari vector

weight, dan *x* adalah indeks dari Dokumen iterasi tersebut.

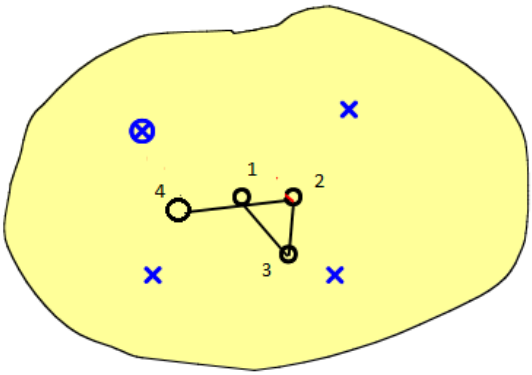
H) Kurangi Radius dari *Neighborhood* setiap iterasi.

$$Radius(T) = Radius(T-1)e^{-\frac{t}{MaxEpoch}}$$

$$Radius(T) = 2x2.71^{-\frac{1}{50}}$$

$$Radius(T) = 1.8$$

I) Jaringan SOM setelah melalui iterasi pertama akan terbentuk seperti berikut akibat penyesuaian terhadap data *input* C.



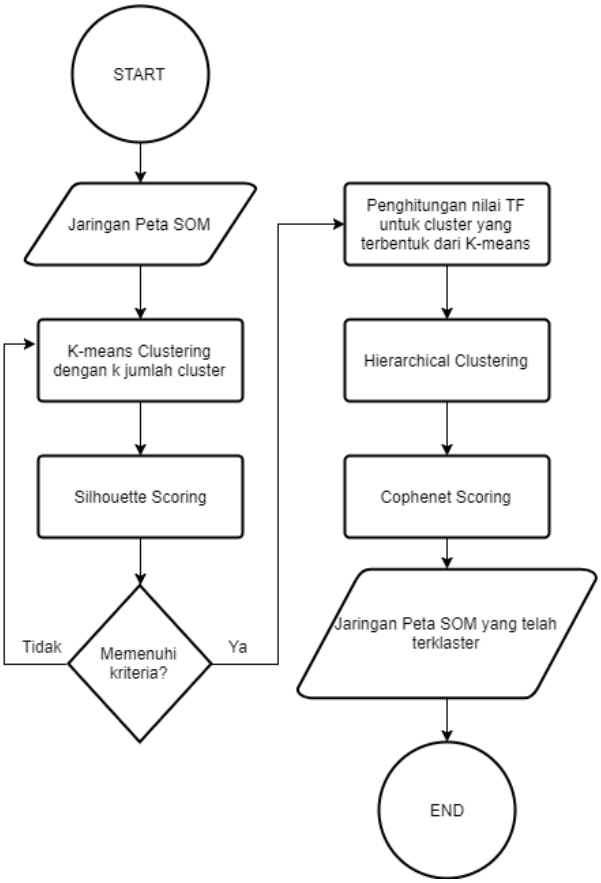
Gambar 3.10 Visualisasi Jaringan SOM yang telah diupdate setelah Jaringan Peta SOM disesuaikan dengan Data Dokumen C

J) Ulangi Langkah A-D sampai Data Input Habis atau sampai Nilai Error sudah konvergen.

3.4 Kombinasi K-means Clustering dan Hierarchical Clustering

Seperti yang telah dijelaskan pada bab sebelumnya, kombinasi metode *K-means Clustering* dan *Hierarchical Clustering* digunakan pada Tugas Akhir ini untuk menyederhanakan jaringan peta yang terbentuk dari proses *training* SOM agar visualisasi menjadi lebih bagus dan terstruktur. *K-means* dipilih sebagai Teknik *Clustering* pada Jaringan SOM dikarenakan struktur keduanya yang mirip.

Pertama-tama yang dilakukan adalah melakukan *K-means Cluster* terhadap *neuron-neuron* yang telah terbentuk pada jaringan SOM. Jumlah *cluster* yang digunakan pada metode *K-means Cluster* merupakan hasil uji coba dari Tugas Akhir ini dengan mengganti jumlah *cluster* yang digunakan dan divalidasi menggunakan *Silhouette Scoring.* Setelah didapatkan k jumlah *cluster* dari *K-means Clustering,* dilakukan perhitungan *Term Frequency* terhadap tiap dokumen yang terdapat pada *cluster* yang akan digunakan sebagai identitas/fitur *Cluster* pada. Selanjutnya *cluster-cluster* ini dikelompokkan lebih lanjut menggunakan *Agglomerative Hierarchical Clustering*. *Cutoff Distance* dan jenis *Linkage* yang digunakan pada *Hierarchical Clustering* akan menjadi scenario uji coba pada Tugas Akhir ini dengan mengganti batas *threshold cutoff distance* yang digunakan dan mencoba beberapa variasi *Linkage* yang akan divalidasi menggunakan *Cophenet Correlation Coefficient*. Secara umum proses kombinasi *K-means Clustering* dan *Hierarchical Clustering* dapat dilihat pada diagram alur 3.11



Gambar 3.11 Langkah-Langkah K-means dan Hierarchical Clustering

Langkah 1. Jaringan peta yang telah terbentuk dari proses *training Self-organizing Maps* (SOM) dikenakan *K-means Clustering* untuk mendapatkan *cluster-cluster* dari *neuron* pada jaringan peta SOM. *Neuron* akan di*cluster* berdasarkan weightnya. K-*means Clustering* dipilih dikarenakan kemiripan struktur antar SOM dan *K-means Clustering.* SOM dan *K-means*

Clustering dasarnya adalah memetakan data ke dalam representasi 2 dimensi berdasarkan posisi/kemiripan data tersebut pada bidang 2 dimensi. Hasil dari tahap ini dapat dilihat pada lampiran bagian C4. Berikut adalah contoh perhitungan kemiripan antar *Neuron dan Cluster* menggunakan *Eucledian Distance*. Contoh menggunakan 2 buah *Cluster* yang *centroid nya* diinisialisasi *random* dari *Neuron* pada jaringan SOM input dan akan diupdate setiap iterasi.

$$\boldsymbol{Neuron1} = \begin{bmatrix} 1.7498 \\ 2.3921 \end{bmatrix}$$

$$\boldsymbol{Centroid1} = \begin{bmatrix} 10.2341 \\ 14.3561 \end{bmatrix}$$

$$\boldsymbol{Centroid2} = \begin{bmatrix} 4.4612 \\ 9.2842 \end{bmatrix}$$

$$d(P,Q) = \sqrt{\sum_{j=1}^{n} (FiturNeuron1(j) - FiturCentroid1(j))^2}$$

$$d(P,Q) = \sqrt{(1.7498 - 10.2341)^2 + (2.3921 - 14.3561)^2}$$

$$d(P,Q) = 14.66$$

$$d(P,Q) = \sqrt{\sum_{j=1}^{n} (FiturNeuron1(j) - FiturCentroid2(j))^2}$$

$$d(P,Q) = \sqrt{(1.7498 - 4.4612)^2 + (2.3921 - 9.2842)^2}$$

$$d(P,Q) = 7.41$$

Berdasarkan perhitungan diatas, dapat disimpulkan bahwa *Neuron 1* lebih dekat terhadap *Centroid Cluster* ke 2, Maka dari

itu *Neuron 1* dimasukkan ke dalam anggota *cluster* 2. Setelah itu, *Centroid* dari *cluster 2* diupdate dengan nilai rata-rata dari semua anggotanya,

$$CentroidCluster2 = \sum \frac{Fitur1, Fitur2, .. FiturX}{N}$$

$$CentroidCluster2 = \frac{(4.4612 + 1.7498), (9.2842 + 2.3921)}{2}$$

$$CentroidCluster2 = 3.1055\ , 5.8315$$

Langkah 2. Pemilihan jumlah K pada *K-means Clustering* pada langkah 1 ditentukan melalui *trial and error.* Maka dari itu, perlu dilakukan sebuah validasi terhadap *clustering* yang telah dilakukan. *Silhouette Scoring* digunakan untuk memvalidasi *cluster-cluster* yang telah terbentuk. *Silhouette Score* merupakan Teknik validasi untuk menilai *cohesion* dan *separability* dari *cluster* yang terbentuk. *Silhouette Score* memiliki nilai -1 sampai 1 dimana nilai lebih besar menunjukkan bahwa *cluster* yang terbentuk semakin bagus. Langkah-langkah untuk proses ini ialah sebagai berikut:

A) Hitung disimilaritas *cluster* ke i terhadap *cluster* lainnya
B) Hitung disimilaritas data-data pada *cluster* ke i
C) *Silhouette Score* dihitung dengan rumus

$$s(i) = \frac{\mathrm{b(i)} - a(i)}{\max\{\mathrm{a(i)}, \mathrm{b(i)}\}}$$

Dimana *s(i)* adalah *Silhouette Score cluster* ke I, *b(i)* adalah disimilaritas *cluster* ke i dengan *neighbouring cluster* nya dijelaskan pada Bab 2 dan *a(i)* adalah disimilaritas dari data-data pada *cluster i.*

D) Ulangi untuk semua *cluster.*
E) Hitung rata-rata dari semua *Cluster Silhouette Score*

Hasil dari tahap ini adalah nilai dari *Silhouette Score Cluster* yang dihasilkan.

$$SilhouetteScore = \begin{bmatrix} 1 : 0.521 \\ 2 : 0.498 \\ \cdots \\ 5 : 0.564 \end{bmatrix}$$

$$SilhouetteScore_{avg} = [0.5125]$$

Langkah 3. Tahap berikutnya adalah dilakukan perhitungan TF untuk tiap *cluster* yang terbentuk dari *K-means Clustering.* Tujuan perhitungan ini adalah untuk mendapatkan fitur dari tiap *cluster* yang akan digunakan untuk perhitungan metode *Hierarchical Clustering.* Langkah-langkah tahap ini adalah sebagai berikut:

A) Untuk setiap term pada dokumen untuk tiap *cluster* dicek pada *list* kata kunci. Tiap *occurrence term* tersebut pada *cluster* ke i, nilai TF *term* tersebut pada *cluster* ke i ditambahkan.
B) Ulangi langkah A sejumlah *n clusters*.

Hasil dari tahap ini ialah *array term-frequency* untuk tiap *cluster* sebagai berikut.

$$TF_{\boldsymbol{cluster1}} = \begin{bmatrix} analisis, 30 \\ jalan, 28 \\ \dots \\ informasi, 7 \end{bmatrix}$$

$$TF_{\boldsymbol{cluster2}} = \begin{bmatrix} teknologi, 57 \\ aplikasi, 78 \\ \dots \\ analisis, 7 \end{bmatrix}$$

Langkah 4. *Cluster-cluster* yang terbentuk dari *K-means Clustering* pada tahap ini dikenakan *Hierarchical Clustering* untuk mengelompokkan *cluster-cluster* tersebut ke dalam kelompok yang lebih besar. Metode *linkage Ward's Method* digunakan pada *Hierarchical Clustering* tahap ini.

Langkah 5. Dilakukan validasi metode *Hierarchical Clustering* dengan menggunakan *Cophenet Correlation Coefficient. Cophenetic Correlation Coefficient* menghitung seberapa mirip *cluster-cluster* yang digabungkan pada algoritma *Hierarchical*

Clustering. Metode ini juga memvalidasi hasil *dendogram* yang dihasilkan *Hierarchical Clustering* sudah merepresentasikan dengan baik jarak asli dari *cluster* yang digabungkan. Rumus *Cophenet Correlation Coefficient* adalah sebagai berikut

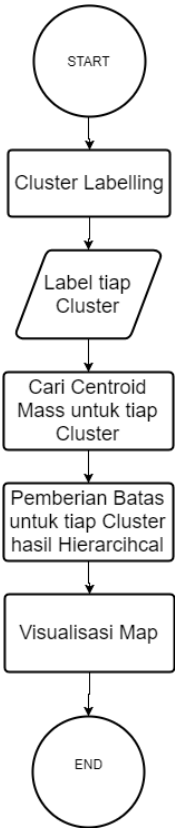
$$c = \frac{\sum i < j(x(i,j) - \bar{x})(t(i,j) - \bar{t})}{\sqrt{(\sum i < j(x(i,j) - \bar{x})^2)(\sum i < j\,(t(i,j) - \bar{t})^2)}}$$

c adalah nilai *Cophenet Correlation Coefficient*, x(i,j) adalah *Ecludiean Distance* antara *cluster* ke I dan j, t(i,j) adalah *dendogrammic distance* dari *cluster* I dan j yang merupakan ketinggian titik dimana kedua *cluster* ini digabungkan pada *Hierarchical Cluster*, $\bar{x}$ adalah rata-rata dari x(i,j) dan $\bar{t}$ adalah rata-rata dari t(i,j). Hasil dari tahapan ini adalah *Cophenet Correlation Coefficient* yang dihasilkan dari *Hierarchical Clustering.*

$$CophenetCorrelationCoefficient \quad = [0.987]$$

3.5 Visualisasi Peta Similaritas Topik Penelitian

Langkah terakhir dalam sistem Visualisasi Peta Similaritas Topik Penlitian ini adalah tahap Visualisasi Peta yang telah terbentuk dari hasil *training* dan *clustering* yang dilakukan. Alur proses untuk tahap Visualisasi Peta Similaritas Topik Penelitian dapa dilahat pada gambar 3.12



Gambar 3.12 Langkah-langkah proses Visualisasi Peta Similaritas Topik Penelitian

Langkah 1. Hal pertama yang dilakukan pada proses ini adalah menentukan label dari tiap *cluster* yang telah terbentuk. Untuk mendapatkan label, dilakukan perhitungan *term-frequency* (TF) untuk setiap kata pada *cluster* yang terbentuk, berikut adalah langkah-langkah untuk menghitung TF untuk tiap *cluster:*

A) Inisiasi *dictionary* berisikan semua kata-kunci yang didapatkan dari judul.

B) Inisiasi *array* sejumlah *N-Cluster* berisikan *dictionary* kata kunci dari langkah A.
C) Untuk setiap *cluster* lakukan perhitungan TF untuk setiap kata yang terdapat pada *cluster* tersebut.
D) Ulangi untuk semua *cluster*.
E) Dilakukan *sorting* pada hasil dari tahap C dan D, dan untuk tiap *cluster* diambil 3 kata-kata dengan nilai TF tertinggi.
F) Dikenakan *threshold* untuk kata-kata yang digunakan sebagai label *cluster*. *Threshold* ditentukan dengan definisi minimal 50% dari TF tertinggi pada *cluster* tersebut.

Dari langkah ini, akan didapat label-label yang digunakan untuk merepresentasikan Visualisasi tiap *Cluster* seperti berikut.

$$Label_{Cluster1} = \begin{bmatrix} "Node" \\ "Routing" \\ "Protokol" \end{bmatrix}$$

$$Label_{Cluster2} = \begin{bmatrix} "Game" \\ "Pemain" \end{bmatrix}$$

$$Label_{Cluster3} = \begin{bmatrix} "Klasifikasi" \\ "Akurasi" \\ "Citra" \end{bmatrix}$$

$$Label_{Cluster4} = \begin{bmatrix} "Segmentasi" \\ "Citra" \end{bmatrix}$$

$$Label_{Cluster5} = \begin{bmatrix} "Perangkat" \\ "Informasi" \\ "Pengguna" \end{bmatrix}$$

Langkah 2. Dilakukan perhitungan *Centroid Mass* untuk tiap *cluster* yang terbentuk. Hal ini dilakukan untuk mendapatkan posisi peletakan label pada visualisasi Peta secara otomatis.

Centroid Mass adalah titik *centroid* dari *cluster* tersebut dengan rumus sebagai berikut.

$$CentroidMass = \sum \frac{Row, Coloumn}{N}$$

Berikut adalah langkah-langkah untuk mendapatkan *Centroid Mass* tiap *cluster*:

A) Lakukan sumasi pada indeks baris dan kolom untuk tiap data pada *cluster* ke i.
B) Hitung rata-rata indeks baris dan kolom untuk *cluster* ke I dengan cara membagi hasil sumasi *cluster* I dengan jumlah dari anggota *cluster* ke i.
C) Ulangi sejumlah *N-cluster*.

Dari langkah-langkah diatas akan didapatkan *Centroid* dari tiap *cluster* sebagai berikut.

$$Centroid_{cluster1} = \begin{bmatrix} kolom = 2 \\ baris = 5 \end{bmatrix}$$

$$Centroid_{cluster2} = \begin{bmatrix} kolom = 4 \\ baris = 13 \end{bmatrix}$$

$$Centroid_{cluster3} = \begin{bmatrix} kolom = 12 \\ baris = 14 \end{bmatrix}$$

$$Centroid_{cluster4} = \begin{bmatrix} kolom = 14 \\ baris = 5 \end{bmatrix}$$

$$Centroid_{cluster5} = \begin{bmatrix} kolom = 3 \\ baris = 8 \end{bmatrix}$$

Centroid ini didapatkan dari hasil rata-rata nilai indeks baris dan kolom untuk tiap *cluster* dan merepresentasikan nilai tengah dari *cluster* tersebut. *Centroid* ini akan digunakan sebagai acuan posisi peletakan otomatis untuk label yang didapat dari langkah 1.

Langkah 3. Pada langkah ini dilakukan pemberian garis batas untuk *cluster-cluster* yang digabungkan pada metode *Hierarchical Clustering.* Garis batas ini berfungsi untuk menandakan bahwa *cluster-cluster* yang berada pada batas yang sama merupakan *cluster* yang mirip satu sama lain.
Langkah-langkah pada tahapan ini adalah sebagai berikut:

A) Gabungkan *cluster-cluster* awal dari hasil *K-means* berdasarkan hasil *Hierarchical Clustering* yang dikenakan, dengan cara mengubah label *cluster* yang digabungkan menjadi satu nilai yang sama.
B) Dengan menggunakan fugnsi *contour* pada *library matplotlib*, Peta Visualisasi diberi garis batas sesuai dengan label *cluster* yang terbentuk.

Langkah 4. Langkah terakhir adalah visualisasi dari Peta Hasil *training* SOM dan gabungan *K-means Clustering* dengan *Hierarchical Clustering*. Langkah-langkah pada tahapan ini adalah:

A) Menggunakan *library* visualisasi dari *Python* yaitu *Matplotlib.*
B) Gunakan fungsi *pcolormesh* untuk menampilkan peta yang telah didapat dari proses-proses sebelumnya. Fungsi ini secara otomatis memberikan warna yang berbeda untuk tiap *cluster* yang berbeda.
C) Gunakan fungsi *text* pada *matplotlib* untuk memberikan teks label pada Peta yang telah terbentuk
D) Gunakan fungsi *contouring* untuk memberikan garis batas pada tiap *cluster* yang digabungkan melalui metode *Hierarchical Clustering.*

Hasil akhir dari tahap ini dapat dilihat pada lampiran bagian C.

BAB IV
IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi dari perancangan yang sudah dilakukan pada bab sebelumnya. Implementasi berupa kode sumber untuk membangun program. Sebelum masuk ke penjelasan implementasi, akan ditunjukkan terlebih dahulu lingkungan untuk melakukan implementasi.

4.1 Lingkungan Implementasi

Implementasi Visualisasi Similaritas Topik Penelitian dengan Pendekatan Kartografi menggunakan Self-organizing Maps (SOM) menggunakan spesifikasi perangkat keras dan perangkat lunak seperti yang ditunjukkan pada Tabel 4.1.

Tabel 4.1 Spesifikasi lingkungan implementasi

Perangkat	Jenis Perangkat	Spesifikasi
Perangkat Keras	Prosesor	Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz 2.39 GHz
	Memori	8 GB 1600
Perangkat Lunak	Sistem Operasi	Windows 10 Pro
	Perangkat Pengembang	Spyder

4.2 Implementasi

Pada sub bab implementasi akan menjelaskan bagaimana pembangunan perangkat lunak secara detail dan menampilkan kode sumber yang digunakan mulai dari tahap *preprocessing* sampai tahap visualisasi. Proses-proses yang memakan waktu yang relatif lama diimplementasikan dengan *multiprocessing* dalam Tugas Akhir ini guna memangkas waktu yang dibutuhkan

proses tersebut. *Multiprocess* diimplementasikan dengan mendistribusikan job-job kedalam beberapa proses dan digunakan *socket* untuk komunikasi data antar proses tersebut

4.2.1 Implementasi Pemprosesan Data Sebelum Praproses

Langkah awal dalam pembuatan sistem Visualisasi Similaritas Topik Penelitian menggunakan *Self-organizing Maps* ialah mengambil data yang diinginkan dari *repository* Tugas Akhir Institut Teknologi Sepuluh Nopember. Data yang didapat masih berupa .sql yang tersimpan di dalam *database* MySQL, maka dari itu perlu digunakan *python connector for* MySQL untuk mengambil data tersebut. Seperti yang dijelaskan pada bab sebelumnya, kolom yang diperlukan untuk Tugas Akhir ini ialah Judul dan Abstrak dari data Tugas Akhir. Kode Sumber tahapan awal dapat dilihat pada Kode Sumber 4.1 dibawah.

```
1.  def Collect():
2.      db = mysql.connector.connect(host='127.0.0.1', port='3306',databas
    e='dsaa',user='root')
3.
4.      cursor = db.cursor()
5.      stringz = "select JUDUL, MW_MA_ABSTRAK_ID, SUBSTRING(
    MW_MA_NRP,1,2) from tugasakhir"
6.      cursor.execute(stringz)
7.      data = cursor.fetchall()
8.      return data
9.  stop_words = {}
10. stop_words = set()
11. with open("stop.txt", "r") as ins:
12.     for line in ins:
13.         stop_words.add(line.rstrip('\n'))
```

Kode Sumber 4.1 Pemrosesan Data Awal

Pada Kode Sumber 4.1 hal pertama yang dilakukan adalah membuat fungsi Collect() dimana kerja fungsi ini ialah melakukan koneksi ke dalam database dan mengambil semua data

Judul dan Abstrak yang dibutuhkan. Setelah itu, pada baris 9-13 adalah kode untuk membuat daftar *stopwords* yang digunakan dalam Tugas Akhir ini. *Stopwords* diambil dari file .txt dan dimasukkan ke dalam *set* agar pengecekan *stopwords* lebih efisien.

4.2.2 Implementasi *Term Frequency – Inverse Document Frequency* (TF-IDF)

Hasil dari proses data awalan masih berupa objek baris dan kolom berisikan Judul dan Abstrak dari tiap data. Objek tersebut perlu diubah menjadi matriks 2 dimensi *document-term* dimana untuk setiap dokumen akan memiliki fitur kata kunci. Langkah pertama yang harus dilakukan ialah mendapatkan kata kunci yang akan digunakan sebagai fitur. Kata kunci diekstraksi dari semua Judul yang terdapat pada *corpus* yang telah dibersihkan dari *stopwords* dan tanda baca. Implementasi ekstraksi kata kunci dapat dilihat pada Kode Sumber 4.2.

```
1.  def TFMatrix(data):
2.      global dokumen
3.      global stop_words
4.
5.      keyindex = {}
6.      cekkeyindex = {}
7.      i = 0
8.      for datum in data:
9.          keywords = re.sub('[^a-zA-Z-.,]+', ' ', datum[0])
10.         keywords = word_tokenize(keywords)
11.         keywords = map(lambda x: x.lower(), keywords)
12.         keywords = filter(lambda x: x not in stop_words, keywords)
13.         keywords = filter(lambda x: x not in string.punctuation, keywor
    ds)
14.         for keyword in keywords:
15.             keyindex[keyword] = 0
16.         info = {
17.             "docno": i,
18.             "judul": datum[0],
19.             "content": datum[1],
```

```
20.             "label": datum[2],
21.             "cek": word_tokenize(datum[1])
22.         }
23.         print i
24.         i += 1
25.         dokumen.append(info)
```

Kode Sumber 4.2 Kode program ekstraksi kata kunci

Pada Kode Sumber 4.2, baris 9-13 bertujuan untuk membersihkan judul dari *stopwords* dan tanda baca, juga menormalisasikan data judul menjadi *lower case.* Baris 14-15 berguna untuk memasukkan tiap kata kunci ke dalam *list* sehingga didapatkan *list* yang berisi semua kata kunci yang akan digunakan. Baris 16-25 berfungsi untuk memasukkan tiap dokumen dalam *corpus* kedalam *dictionary* untuk kepentingan proses selanjutnya. Hasil dari Kode Sumber 4.2 adalah sebuah list berisikan kata-kata kunci yang akan digunakan sebagai fitur dari dokumen pengujian. Hambatan dalam proses ini ialah proses ini harus dijalankan berulang kali ketika implementasi dikarenakan daftar *stopwords* masih belum lengkap, maka dari itu untuk setiap uji coba daftar *stopwords* ditambahkan agar lebih kaya.

Setelah didapatkan kata kunci yang dibutuhkan, tahap selanjutnya ialah menghitung *document frequency* dari setiap kata kunci tersebut. Proses perhitungan dapat dilihat pada Kode Sumber 4.3 dibawah.

```
1.  def CountIDF(data,keyindex):
2.      print 'IDF'
3.      for doc in data:
4.          exist = {}
5.          for word in doc['cek']:
6.              if keyindex.has_key(word):
7.                  if not exist.has_key(word):
8.                      keyindex[word] += 1
9.                      exist[word] = 'done'
10.     return keyindex
```

Kode Sumber 4.3 Perhitungan Document Frequency

Pada Kode Sumber 4.3, tiap kata kunci dicek apakah terdapat pada dokumen-dokumen dalam *corpus*, dapat dilihat pada baris ke 6 pada Kode Sumber 4.3. Kemudian proses pengecekan berikutnya pada baris 7 bertujuan untuk mengecek apakah kata-kata tersebut sudah pernah ditambahkan frekuensinya pada pengecekan dokumen yang sama. Dengan demikian, dari proses tersebut didapatkan *document frequency* untuk tiap kata kunci yang akan digunakan nantinya untuk perumusan *inverse document frequency*.

Setelah itu, proses selanjutnya ialah menghitung *term frequency* (TF) pada tiap dokumen. Proses ini memakan waktu paling lama dari semua Tahapan Tugas Akhir ini, dikarenakan kata kunci yang didapatkan untuk *corpus* yang digunakan relative cukup besar. Maka dari itu, pada proses perhitungan TF diimplementasikan *multiprocessing* agar mempercepat perhitungan. Kode Sumber untuk implementasi perhitungan TF dapat dilihat pada Kode Sumber 4.4.

```
1.  def CountTF(data,keyindex,out_q,bawah):
2.      print 'TF'
3.      som = np.zeros((len(data),len(keyindex)))
4.      j = 0
5.      for doc in data:
6.          for word in doc['cek']:
7.              k=0
8.              for key in keyindex:
9.                  if word == key:
10.                     som[j][k] += 1
11.                     break
12.                 k+=1
13.         print j
14.         j+=1
15.     print 'Count TF Selesai'
16.     out_q.put((bawah,som))
17. count = len(dokumen)/4
```

```
18.  bawah = 0
19.  i =0
20.  minn = 0
21.  maxx = len(dokumen)
22.  out_q = Queue()
23.  for i in range(4):
24.      atas = bawah + count
25.      if i == 3:
26.          if atas < maxx:
27.              atas = maxx
28.      if atas > maxx:
29.          atas = maxx
30.      jobs = []
31.      p = Process(target=CountTF, args=(dokumen[bawah:atas],keyindex
     ,out_q,bawah))
32.      jobs.append(p)
33.      p.start()
34.      bawah = atas
35.  tempqueue = []
36.  for i in range(4):
37.      tempqueue.append(out_q.get())
38.  for job in jobs:
39.      job.join()
40.  sorted(tempqueue,key=itemgetter(0))
41.
42.  i = 0
43.  for temp in tempqueue:
44.      if i == 0:
45.          som = temp[1]
46.          i = i+1
47.      else:
48.          som = np.concatenate((som,temp[1]),axis=0)
```

Kode Sumber 4.4 Perhitungan Term Frequency dan Implementasi Multiprocessing

Pada baris 5-6 ialah kode untuk melakukan iterasi untuk tiap kata-kata pada dokumen dalam *corpus.* Selanjutnya pada baris 8-14 adalah proses untuk perhitungan TF pada tiap dokumen. Salah satu cara untuk mengkomunikasikan data dalam *multiprocessing* ialah dengan menggunakan *queue socket,* hal ini

diimplementasikan pada baris 16. Pada barus 30-34 ialah kode untuk pemanggilan fungsi perhitungan TF secara *multiprocessing* dengan menggunakan 4 proses *asynchronous*. Baris 36-40 merupakan kode yang berfungsi untuk mengambil semua data hasil proses *multiprocessing* dan yang terakhir pada baris 42-48 merupakan proses penggabungan semua data hasil *multiprocessing* kedalam satu obyek array.

Dikarenakan ukuran fitur yang cukup besar, maka digunakan *Principal Component Analysis* (PCA) untuk mereduksi fitur tersebut. Implementasi PCA dapat dilihat pada Kode Sumber 4.5.

```
1.  pca = PCA(n_components=2500)
2.  som = pca.fit_transform(som)
3.  print(pca.explained_variance_ratio_)
```

Kode Sumber 4.5 Implementasi PCA untuk reduksi fitur

Implementasi menggunakan *library sklearn.* Untuk menentukan banyak *component* yang dipakai untuk proses selanjutnya, dilakukan pengecekan terhadap *variance ratio component* yang dihasilkan. Baris 1 dan 2 digunakan untuk mereduksi fitur. Baris 3 digunakan untuk mengecek *variance ratio* dari tiap *component* yang terbentuk. *Dari* implementasi tersebut, diputuskan menggunakan 2500 *component* yang merepresentasikan 85% dari fitur asal.

4.2.3 Implementasi *Self-organizing Maps* (SOM)

Proses selanjutnya setelah didapatkan matriks *document-terms* ialah melakukan *training* pada jaringan SOM guna menempatkan dokumen-dokumen dalam *corpus* ke dalam jaringan 2 dimensi untuk mempermudah visualisasi. Implementasi algoritma SOM dilakukan menggunakan *library* SOMPY pada *python.* Implementasi algoritma ini dapat dilihat pada Kode Sumber 4.6.

```
1.  def SOMTrain(som):
2.      mapsize = [50,50]
3.      datasom = sompy.SOMFactory.build(som, mapsize, mask=None, m
    apshape='planar', lattice='rect',
4.          initialization='pca', neighborhood='gaussian', training='seq', nam
    e='sompy')
5.      datasom.train(n_job=1, verbose='info')
6.      return datasom
```

Kode Sumber 4.6 Implementasi Self-organizing Maps (SOM)

Baris 2-4 pada Kode Sumber 4.6 merupakan konfigurasi awal jaringan SOM yang akan digunakan. Pada implementasi digunakan konfigurasi *mapsize* 50x50 yang ditentukan berdasarkan uji coba dan referensi [3], *rectangular lattice*, inisialisasi menggunakan PCA, *gaussian neighborhood function*, mode *training sequential*. Justifikasi penggunaan parameter sudah dijelaskan pada bab sebelumnya. Baris 5 adalah kode yang berfungsi untuk menjalankan proses *training.* Proses *training* tidak dipecah kedalam beberapa *job* dikarenakan diamati jika proses *training* dipecah, jaringan *training* yang dihasilkan tidak optimal diakibatkan pemecahan proses *training* kedalam bagian-bagian yang lebih kecil. Hasil dari proses ini ialah jaringan peta SOM yang sudah terkonfigurasi menyesuaikan dengan data input.

4.2.4 Implementasi K-means Clustering

Setelah proses *training* menggunakan *Self-organizing Maps* dijalankan, jaringan peta SOM yang telah terbentuk akan dilakukan *clustering* dengan tujuan untuk menyederhanakan visualisasi yang akan ditampilkan. *Clustering* pada Tugas Akhir ini seperti yang dijelaskan pada bab 3.4 menggunakan kombinasi dari algoritma *K-means Clustering* dan *Hierarchical Clustering.*

Tahap yang dilakukan adalah *K-means Clustering* terlebih dahulu. Tahapan implementasi *K-means Clustering* dapat dilihat pada Kode Sumber 4.7.

```
1.  def RecordCluster(ncluster, cl, result):
2.      isicluster = {}
3.      global dokumen
4.      hasilcluster = np.empty(len(dokumen))
5.      for x in xrange(0,ncluster):
6.          isicluster[x] = []
7.
8.      i = 0
9.      for item in result:
10.         isicluster[cl[item]].append(i)
11.         hasilcluster[i] = cl[item]
12.         i = i+1
13.     return isicluster,hasilcluster
14.
15.
16. def save_results(isicluster):
17.     global dokumen
18.     sumdokumencluster = []
19.     with open('results.txt', 'w') as fout:
20.         for clusterx,titles in isicluster.iteritems():
21.             fout.write('Cluster ke-' + str(clusterx) + '\n')
22.             y = 1
23.             for judul in titles:
24.                 fout.write(str(y) + '.' + str(dokumen[judul]["judul"].encode('
    utf-8")) + '\n')
25.                 y = y+1
26.             sumdokumencluster.append(y)
27.     return sumdokumencluster
28. cl = datasom.cluster(n_clusters=12)
29. ncluster = 12
30. tfidf = []
31. isicluster,hasilcluster = RecordCluster(ncluster,cl,result)
32. sumdokumencluster = save_results(isicluster)
33. silhouette = silhouette_score(som,hasilcluster)
34. print silhouette
```

Kode Sumber 4.7 Implementasi K-means Clustering

Fungsi RecordCluster pada baris 1-13, merupakan fungsi untuk mengambil data hasil prediksi dokumen untuk tiap *cluster*, sehingga dari fungsi ini didapat *list* isi dokumen untuk tiap *cluster*. Sedangkan fungsi save_results pada baris 16-27 berfungsi untuk mengoutputkan data dokumen untuk tiap *cluster* ke dalam file *results.txt*, dan juga menghitung jumlah dokumen untuk tiap *cluster*.

Validasi jumlah *cluster* untuk parameter *K-means Clustering* dilakukan dengan menggunakan *Silhouette Scoring* ditunjukkan pada baris 33. Setelah dilakukan uji coba, didapatkan 12 merupakan jumlah *cluster* yang paling optimal. Dari proses ini, didapatkan hasil klasterisasi untuk tiap *neuron* pada jaringan peta SOM yang telah melalui proses *training* sebelumnya.

4.2.5 Implementasi Hierarchical Clustering

Clustering lebih lanjut dikenakan pada hasil klasterisasi pada tahap *K-means Clustering* sebelumnya. Hal ini dilakukan bertujuan untuk menyederhanakan kembali visualisasi yang didapat dari *K-means Clustering* dengan cara mengelompokkan *cluster* yang tingkat kemiripannya tinggi. *Hierarchical Clustering* pada Tugas Akhir ini diimplementasikan menggunakan *library sklearn*, dan divalidasi menggunakan *Cophenet Correlation Coefficient* untuk memastikan bahwa hasil *Hierarchical Clustering* yang didapat layak. Proses implementasi *Hierarchical Clustering* dapat dilihat pada Kode Sumber 4.8.

```
1.  def TFHierarchical(dict_cluster,keyindex):
2.      print 'Hierarchical Counting'
3.      hierarchical = np.zeros((len(dict_cluster),len(keyindex)))
4.      j = 0
5.      for cluster in dict_cluster:
6.          for key,value in cluster.iteritems():
7.              k=0
8.              for check in keyindex:
9.                  if key == check:
```

```
10                  hierarchical[j][k] = value
11                  break
12              k+=1
13          print j
14          j+=1
15      print 'Matrix Hierarchical selesai'
16      return hierarchical
17
18  def HClustering(hierarchical):
19      HCluster = linkage(hierarchical, 'ward')
20      plt.figure(figsize=(25, 10))
21      plt.title('Hierarchical Clustering Dendrogram')
22      plt.xlabel('sample index')
23      plt.ylabel('distance')
24      dendrogram(
25          HCluster,
26          leaf_rotation=90.,  # rotates the x axis labels
27          leaf_font_size=8.,  # font size for the x axis labels
28      )
29      plt.show()
30      return HCluster
31  hierarchical = TFHierarchical(dict_cluster,keyindex)
32  HCluster = HClustering(hierarchical)
33  c, coph_dists = cophenet(HCluster, pdist(hierarchical))
34  print c
```

Kode Sumber 4.8 Implementasi Hierarchical Clustering

Langkah pertama yang dilakukan untuk mengimplementasikan *Hierarchical Clustering* ialah mendapatkan *Term-Frequency* (TF) untuk tiap *cluster* yang terbentuk, TF untuk tiap kata kunci inilah yang akan digunakan untuk mengukur kedekatan antar *cluster* pada algoritma *Hierarchical Clustering*. Fungsi TFHierarchical pada baris 1-16 merupakan implementasi dari penghitungan TF untuk tiap *cluster*. Selanjutnya fungsi HClustering pada baris 18-30 digunakan untuk memproses tiap *cluster* dalam *Hierarchical Clustering*. Baris 20-29 berguna untuk memvisualisasikan hasil *Hierarchical Clustering* dalam dendogram. *Hierarchical*

Clustering pada Tugas Akhir ini diimplementasikan menggunakan *Ward's Method* sebagai *linkage* nya, dikarenakan metode tersebut menghasilkan *Cophenet Correlation Coefficient* yang tertinggi dibandingkan metode *linkage* lainnya. Validasi menggunakan *Cophenet Correlation Coefficient* diiimplementasikan pada baris 33-34.

4.2.6 Implementasi Cluster Labelling

Setelah didapatkan *cluster-cluster* yang dibutuhkan untuk visualisasi dari proses *clustering* sebelumnya, yang dibutuhkan berikutnya adalah *label* untuk tiap *cluster* tersebut. Label tiap *cluster* diambil dari 3 kata kunci yang memiliki nilai TF tertinggi pada *cluster* tersebut. TF dianggap lebih cocok dibandingkan TF-IDF untuk menjadi acuan label dikarenakan label seharusnya menggambarkan keseluruhan isi dokumen yang termasuk dalam *cluster* tersebut, jika menggunakan TF-IDF maka kata kunci yang sering muncul di suatu dokumen tapi tidak di tempat lain akan memiliki nilai yang tinggi. Perhitungan TF dilakukan menggunakan fungsi pada Kode Sumber 4.4 dan implementasi ekstraksi label dari tiap *cluster* dapat dilihat pada Kode Sumber 4.9.

```
1.  def compute_tf_percluster(tf,sumdokumencluster):
2.      hasil_label = []
3.      dict_cluster = []
4.      i = 0
5.      for tfc in tf:
6.          tf_dict = {}
7.          for key,value in tfc.iteritems():
8.              tf_dict[key] = value[0]
9.          newA = dict(sorted(tf_dict.iteritems(), key=operator.itemgetter(1)
    , reverse=True)[:3])
10.         hasil_label.append(newA)
11.         i = i+1
12.         dict_cluster.append(tf_dict)
13.     return hasil_label,dict_cluster
```

Kode Sumber 4.9 Implementasi Cluster Labelling

Proses pengambilan 3 kata kunci dengan nilai TF tertinggi dapat dilihat pada baris 9-10. Dari proses implementasi ini, didapatkan 3 label untuk tiap *cluster* yang terbentuk dari proses sebelumnya.

4.2.7 Implementasi Visualisasi Peta SOM

Langkah akhir dalam Tugas Akhir ini adalah implementasi visualisasi dari hasil jaringan peta yang telah melalui hasil *training* dan *clustering* pada proses sebelumnya. Implementasi visualisasi menggunakan *library* matplotlib pada *Python*.

A) Visualisasi Map Dasar

Berikut adalah implementasi visualisasi jaringan peta SOM untuk pembentukan map dasar

Implementasi visualisasi jaringan peta SOM dapat dilihat pada Kode Sumber 4.10

```
1.  x = np.arange(0,50,1)
2.  y = np.arange(0,50,1)
3.  xx,yy = np.meshgrid(x,y)
4.  xx = xx.reshape(50*50)
5.  yy = yy.reshape(50*50)
6.  fig = plt.figure(figsize =(16,8))
7.  xi,yi = np.meshgrid(x,y)
8.  ax2 = plt.subplot(121)
9.  prop = fm.FontProperties(fname='PTC55F.ttf')
10. plt.gca().invert_yaxis()
11. plt.pcolormesh(xi,yi,vis,cmap = plt.cm.Accent,shading='gouraud')
12. plt.xticks([])
13. plt.yticks([])
14. plt.show()
```

Kode Sumber 4.10 Implementasi Visualisasi Peta Dasar SOM

Baris 1-7 pada Kode Sumber 4.10 merupakan proses konfigurasi awal untuk membentuk *array* berukuran 50x50 yang dibutuhkan sebagai *parameter* visualisasi matplotlib. Baris 10-14 adalah proses untuk menampilkan visualisasi peta SOM yang telah terbentuk. Hasil akhir dari sistem ini adalah visualisasi Peta Similaritas Topik Penelitian bisa dilihat pada lampiran C8.

B) Visualisasi Map dengan Garis Batas tiap *Cluster*

```
1.  x = np.arange(0,50,1)
2.  y = np.arange(0,50,1)
3.  xx,yy = np.meshgrid(x,y)
4.  xx = xx.reshape(50*50)
5.  yy = yy.reshape(50*50)
6.  fig = plt.figure(figsize =(16,8))
7.  xi,yi = np.meshgrid(x,y)
8.  ax2 = plt.subplot(121)
9.  for i in range(ncluster):
10.     plt.contour(xi,yi,viscontour == i, contours=1,colors=['black'],linewi
    dths=1.2,alpha=.75)
11. prop = fm.FontProperties(fname='PTC55F.ttf')
12. plt.gca().invert_yaxis()
13. plt.pcolormesh(xi,yi,vis,cmap = plt.cm.Accent,shading='gouraud')
14. plt.xticks([])
15. plt.yticks([])
16. plt.show()
```

Kode Sumber 4.11 Implementasi Visualisasi Peta Dasar SOM

Baris 1-7 pada Kode Sumber 4.10 merupakan proses konfigurasi awal untuk membentuk *array* berukuran 50x50 yang dibutuhkan sebagai *parameter* visualisasi matplotlib. Baris 9-10 digunakan untuk menggambar batas tiap *cluster* dimana *cluster* secara *manual* digabungkan menurut hasil dari *Hierarchical Clustering.*

Baris 12-16 adalah proses untuk menampilkan visualisasi peta SOM yang telah terbentuk. Hasil akhir dari sistem ini adalah visualisasi Peta Similaritas Topik Penelitian bisa dilihat pada lampiran Gambar 4.1.

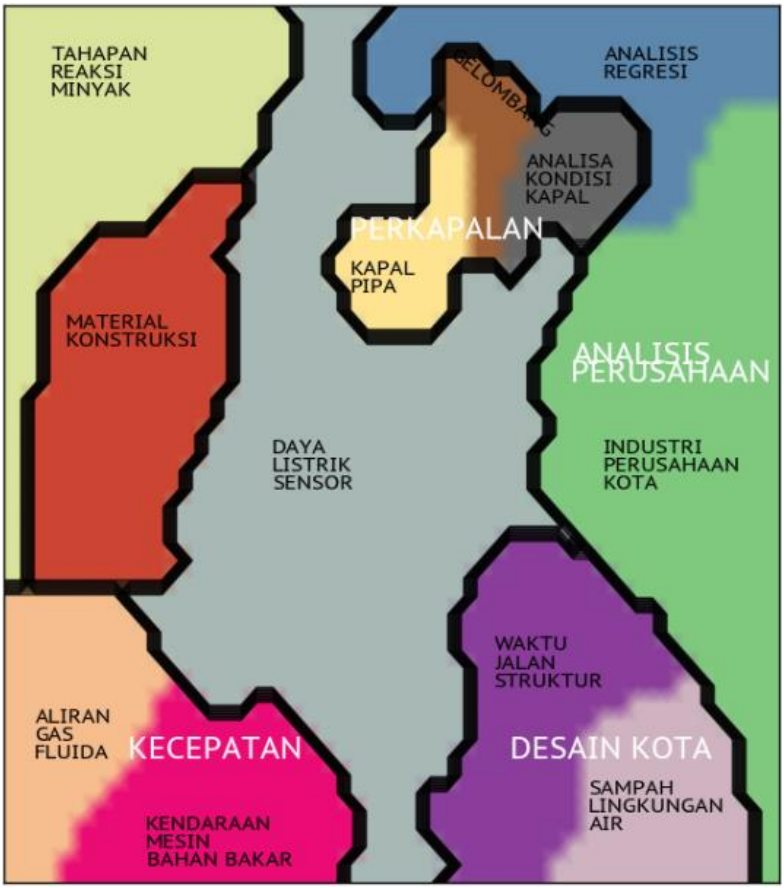
Untuk peletakan otomatis label dari hasil 4.2.6 pada peta yang terbentuk, dilakukan pencarian *Centroid Mass* dari *cluster-cluster* yang telah terbentuk yang nantinya akan dijadikan posisi label dari *cluster* tersebut. Implementasi *Centroid Mass* dapat dilihat pada Kode Sumber 4.12.

```
1.  centroidx = []
2.  centroidy = []
3.
4.  #finding centroid
5.  for i in range(ncluster):
6.      avgrow = 0
7.      avgcol = 0
8.      occu = 0
9.      for j in range(20):
10.         k = 0
11.         for col in vis[j]:
12.             if col == i:
13.                 avgrow = avgrow + k
14.                 avgcol = avgcol + j
15.                 occu = occu + 1
16.             k = k+1
17.     centroidx.append(avgrow/occu)
18.     centroidy.append(avgcol/occu)
19.
20. print centroidx
21. print centroidy
```

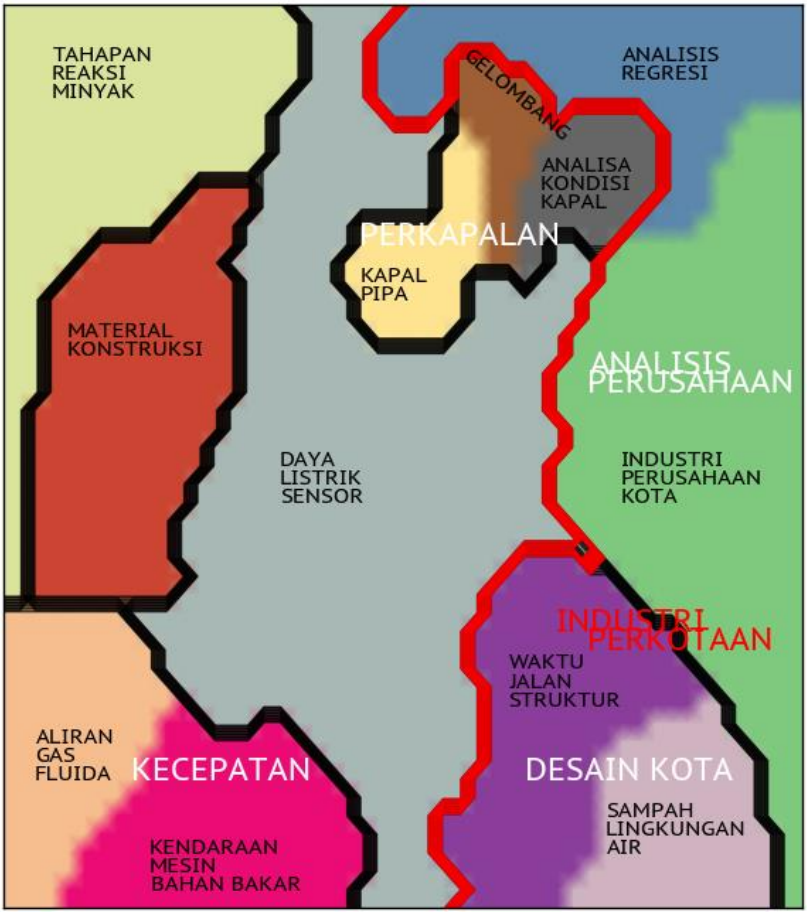
Kode Sumber 4.12 Implementasi Centroid Mass

Pada Kode Sumber di atas, dilakukan perhitungan *centroid* dari tiap *cluster*. Baris 9-16 berguna untuk menghitung *average row* dan *coloumn* dari tiap *cluster*. Baris 17-18 digunakan untuk

memasukkan hasil perhitungan dalam suatu *array.* Baris 20-21 digunakan untuk mencetak hasil dari perhitungan *Centroid Mass.*



Gambar 4.1 Hasil Visualisasi Jaringan Peta SOM Cutoff Distance 4000



Gambar 4.2 Hasil Visualisasi Jaringan Peta SOM Cutoff Distance 6000

(Halaman sengaja dikosongkan)

BAB V
UJI COBA DAN EVALUASI

Bab ini akan dijelaskan mengenai skenario uji coba pada perangkat lunak yang telah dibangun. Setelah itu, hasil uji coba akan dievaluasi kinerjanya sehingga dapat diputuskan apakah perangkat lunak ini mampu menyelesaikan permasalahan yang telah dirumuskan diawal. Secara garis besar, bab ini berisikan pembahasan mengenai lingkungan pengujian, data pengujian, dan uji kinerja.

5.1 Lingkungan Pengujian

Lingkungan pengujian pada uji coba permasalahan Visualisasi Similaritas Topik Penelitian dengan Pendekatan Kartografi menggunakan *Self-organizing Maps* menggunakan spesifikasi keras dan perangkat lunak seperti yang ditunjukkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Lingkungan Uji Coba

Perangkat	Jenis Perangkat	Spesifikasi
Perangkat Keras	Prosesor	Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz 2.39 GHz
	Memori	8 GB 1600
Perangkat Lunak	Sistem Operasi	Windows 10 Pro
	Perangkat Pengembang	Spyder

5.2 Data Pengujian

Data yang digunakan sebagai pengujian dalam perangkat lunak Visualisasi Similaritas Topik Penelitian menggunakan *Self-organizing Maps* (SOM) adalah data *repository* Tugas Akhir Institut Teknologi Sepuluh Nopember (ITS). *Corpus* ini berisi 13.324 Data Tugas Akhir dari 5 fakultas yang ada di ITS dengan rincian 2194 Data dari Fakultas FMIPA, 5360 Data dari Fakultas FTI, 3078 Data dari Fakultas FTSP, 1156 Data dari Fakultas FTK, dan 1250 Data dari Fakultas FTIF.

5.3 Skenario Uji Coba

Sebelum melakukan uji coba, perlu ditentukan skenario yang akan digunakan dalam uji coba. Melalui skenario uji coba ini, perangkat akan diuji apakah sudah berjalan dengan benar, bagaimana performa pada masing-masing skenario dan perbandingan performa antara skenario mana yang memiliki hasil paling baik. Pada Tugas Akhir ini, terdapat berbagai macam skenario uji coba, yaitu:

1. Perhitungan performa berdasarkan *Quantization Error* pada algoritma *Self-organizing Maps* untuk menentukan nilai *Map Size* yang optimal.
2. Perhitungan performa *Principal Component Analysis* dalam reduksi fitur berdasarkan jumlah *component* yang digunakan untuk menentukan nilai *Threshold Variance Ratio.*
3. Perhitungan performa berdasarkan jumlah K *cluster* yang digunakan pada metode *K-means Clustering* untuk mendapatkan nilai K yang optimal.
4. Perbandingan performa berdasarkan metode *linkage* yang digunakan pada metode *Hierarchical Clustering* untuk menentukan metode *linkage* yang paling optimal.

5. Kuisioner terhadap sejumlah sampel target pengguna guna mengukur kejelasan informasi dari peta visualisasi yang terbentuk.

5.3.1 Skenario Uji Coba Performa Self-Organizing Maps

Self-organizing Maps(SOM) adalah metode jaringan *Kohonen Neural Network* yang bekerja dengan cara memetakan data *input* dengan dimensi yang besar ke dalam 2 dimensi (*Many to few mapping)*. Salah satu cara yang umum digunakan untuk metode *many to few mapping* adalah perhitungan *Quantization Error*. *Quantization Error* adalah besaran yang merupakan perhitungan dari selisih dari data *input* yang asli terhadap data dengan *value* yang sudah disesuaikan pada dimensi peta SOM yang digunakan atau biasa disebut dengan *round-off error*. Hasil uji coba untuk tiap percobaan nilai *Map Size* dapat dilihat pada tabel 5.7. Pada referensi *paper* Skupin, digunakan *Map Size* 20x40 untuk Dokumen sebanyak 2.200 Dokumen [3]. Hasil pada tiap percobaan *Map Size* juga dapat dilihat pada lampiran bagian C.

Tabel 5.2 Quantization Error untuk tiap Map Size Uji Coba

Map Size	Quantization Error
25 x 25	49.641
40 x 40	19.752
50 x 50	8.641
75 x 75	11.264

Kesimpilan dari hasil uji coba pada Tabel 5.2 adalah Peta berukuran 50x50 paling optimal dikenakan terhadap data *input* sebanyak 13.324 Dokumen dengan *Quantization Error* 8.641.
Pada lampiran C5, C6, dan C7, dengan percobaan ukuran *Map Size* 40x40, 75x75, dan 100x100 didapatkan hasil peta yang kurang bagus, ditandai dengan tersebarnya *cluster-cluster* yang terbentuk. *Cluster* yang sama didapati pada *neuron* yang berjauhan, menandakan bahwa hasil *update weight* dari *neuron* kurang tepat sehingga *weight* antar *neuron* yang berjauhan memiliki nilai yang serupa.

5.3.2 Skenario Uji Coba Perhitungan Performa *Principal Component Analysis* berdasarkan Jumlah Component

Skenario Uji Coba yang kedua adalah perhitungan performa *Principal Component Analysis* berdasarkan jumlah *Component* yang digunakan. *Principal Component Analysis* dinilai perlu digunakan pada Tugas Akhir kali ini sebab fitur yang diekstraksi dari Data Input relative sangat banyak, maka dari itu PCA diperlukan untuk memangkas waktu *running* pada proses *Training* menggunakan algoritma SOM. Akan dilakukan uji coba untuk beberapa nilai jumlah *component* yang digunakan dari hasil PCA. Uji coba ini dilakukan guna mengetahui jumlah *component* yang optimal dan efisien dinilai berdasarkan *variance ratio* yang dicakup dari *component* yang digunakan, beserta *training time*, dan *quantization error* pada algoritma SOM. *Variance Ratio* adalah nilai yang menyatakan persentase representasi *component* yang digunakan terhadap fitur awal Data Input. Uji coba dilakukan ke dalam algoritma SOM dengan *Map Size* 50x50 sesuai dengan hasil terbaik pada Uji Coba pertama.

Tabel 5.3 Hasil Uji Coba Principal Component Analysis

Jumlah Component	Variance Ratio	Training Time	Quantization Error	Selisih
13.324	100%	1 Hr 21 Min 17 Sec	8.641	0
5000	96%	54 Min 22 Sec	9.072	0.431
3500	92%	40 Min 38 Sec	9.298	0.226
2500	85%	28 Min 02 Sec	9.451	0.153
1500	77%	22 Min 42 Sec	12.361	2.91

Berdasarkan hasil uji coba pada Tabel 5.3, dapat disimpulkan bahwa selisih *Quantization Error* yang dihasilkan dari jumlah komponen 3500 dan 2500 paling kecil, serta memangkas waktu *training* yang relative besar yaitu 12 Menit 36 Detik. Sedangkan untuk selisih *Quantization Error* dari jumlah komponen 2500 dan 1500 menunjukkan angka yang cukup besar yaitu 2.91, serta perbedaan waktu *training* yang tidak signifikan yaitu 5 Menit 20 Detik. Maka dari itu, dinilai Jumlah Komponen 2500 merupakan Jumlah Komponen dengan performa paling optimal.

5.3.3 Skenario Uji Coba Perhitungan Performa berdasarkan jumlah cluster pada metode *K-means Clustering*

Skenario Uji Coba ketiga ialah perhitungan performa menggunakan *Silhouette Scoring* pada percobaan jumlah *cluster* pada metode *K-means Clustering*. *Silhouette Scoring* memiliki rentang nilai dari -1 sampai 1, dimana nilai negatif umumnya menandakan bahwa prediksi data untuk *cluster* tersebut banyak

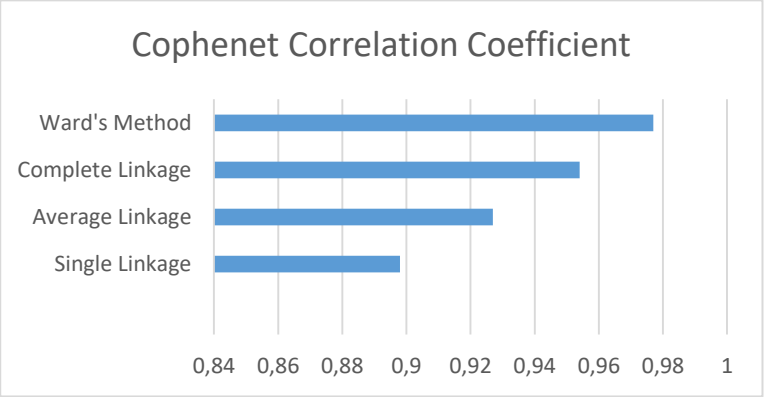
yang tidak tepat. Sedangkan nilai 0 menandakan bahwa *cluster* yang terbentuk masih ambigu, dimana tiap *cluster* yang terbentuk masih memiliki tingkap kesamaan yang tinggi antar *cluster* dan nilai positif jika *cluster* yang dibentuk sudah sesuai dengan data input dan memiliki jarak yang rendah antar tiap data pada *cluster* yang sama (*Cohesivity*) serta *cluster-cluster* yang terbentuk memiliki jarak yang jauh antar sesama *cluster* (*Separability*). Nilai *Silhouette Scoring* untuk tiap percobaan jumlah *cluster* dapat dilihat pada Lampiran B1.

Kesimpulan dari percobaan performa *K-means Clustering* menggunakan *Silhouette Scoring* adalah 12 *cluster* merupakan jumlah yang paling optimal untuk dataset *corpus* Penelitian Tugas Akhir Institut Teknologi Sepuluh Nopember. Ketika digunakan *cluster* dengan jumlah yang lebih sedikit, didapatkan nilai yang lebih rendah diakibatkan masih ada data yang terprediksi masuk ke dalam *cluster* yang salah. Sedangkan ketika digunakan jumlah *cluster* yang lebih banyak, didapatkan nilai *Silhouette Score* juga semakin menurun diakibatkan *cluster* yang terbentuk mirip satu sama lain. Hasil percobaan merupakan hasil rata-rata dari 10 kali percobaan dikarenakan *K-means Clustering* yang hasilnya tidak selalu sama diakibatkan inisialisasi titik *cluster* yang acak.

5.3.4 Skenario Uji Coba Performa Metode Hierarchical Clustering

Pada skenario uji coba yang keeempat akan diuji performa dari metode *Hierarchical Clustering* menggunakan validasi *Cophenet Correlation Coefficient*. *Cophenet Correlation Coefficient* adalah suatu metode pengukuran yang umum digunakan untuk menguji *nested cluster*. Nilai rentang dari *Cophenet Correlation Coefficient* adalah 0 sampai 1 dimana *cluster* yang terbentuk dari *Hierarchical Clustering* semakin

bagus jika nilai *Cophenet* mendekati 1. Hasil performa dari metode *Hierarchical Clustering* dapat dilihat pada gambar 5.1.



Gambar 5.1 Perbandingan Cophenet Correlation Coefficient untuk tiap metode Linkage

Berdasarkan hasil uji coba yang ditunjukkan pada Gambar 5.1, didapat bahwa *Ward's Method* merupakan metode yang cocok untuk dataset yang digunakan pada Tugas Akhir ini secara spesifik dikarenakan data masukkan pada Tugas Akhir ini memiliki dimensi fitur yang sangat besar atau biasa disebut dengan *Multivariate Normal Distribution*. Seperti dijelaskan pada bab 2.4, *Ward's Method* cocok untuk digunakan pada data *Multivariate Normal Distributed.*

5.3.5 Kuisioner Uji Coba

Kuisioner perlu dilakukan sebagai salah satu uji coba dalam Tugas Akhir ini dikarenakan hasil dari Tugas Akhir ini adalah sebuah Peta Visualisasi Similaritas Topik Penelitian yang bertujuan untuk menjadi sebuah pedoman bagi Mahasiswa-mahasiswa guna menentukan topik penelitian yang akan diriset di masa depan. Oleh karena itu, perlu dilakukan validasi terhadap

target pengguna akan kejelasan dan manfaat informasi dari peta yang telah terbentuk.

A. Target Responden

Target responden dari kuesioner ini adalah mahasiswa teknik informatika yang aktif dalam riset di ITS. Keaktifan mahasiswa dalam riset di ITS dapat diketahui melalui beberapa cara yaitu mahasiswa tersebut pernah terlibat atau tergabung dalam riset yang dimiliki oleh dosen, selanjutnya kriteria lainnya adalah mahasiswa tersebut aktif dalam laboratorium di Teknik Informatika sebagai administrator lab atau asisten laboratorium.

B. Skenario Uji Coba

Kuisioner uji coba dilakukan terhadap hasil Visualisasi Peta dengan *Silhouette Score* dan *Cophenet Correlation Score* yang tertinggi, yakni Peta dengan nilai K = 12 dan *Hierarchical Clustering* menggunakan *Ward's Method*. Kuisioner Uji Coba dibagi dalam 4 skenario yakni:

1. Kuisioner pertama mengandung pertanyaan-pertanyaan umum menyangkut kejelasan dan tampilan dari peta yang terbentuk. Skenario pertama bertujuan untuk mengetahui apakah peta yang terbentuk sudah informatif dan menarik untuk dilihat pengguna.
2. Kuisioner kedua mengandung pertanyaa-pertanyaan meliputi *Hierarchical Clustering* yang dikenakan pada peta visualisasi. Skenario kedua bertujuan untuk

mengetahui preferensi dari pengguna perihal tingkat kedalaman dari peta yang dihasilkan.

3. Kuisioner ketiga mengandung pertanyaan-pertanyaan meliputi *labelling* yang dilakukan pada peta visualisasi. Skenario ketiga bertujuan untuk mengetahui tingkat kejelasan label yang dihasilkan pada Tugas Akhir ini.
4. Kuisioner terakhir mengandung pertanyaan-pertanyaan meliputi hasil peta visualisasi yang terfokus pada satu bidang saja. Skenario kuisioner ini bertujuan untuk mengetahui apakah data yang digunakan memiliki performa lebih baik jika hanya berasal dari rumpun yang sama.

5.6.3.1 Kuisioner Skenario Pertama

Skenario pertama bertujuan untuk mengetahui tingkat kejelasan informasi dan estetika dari peta yang telah terbentuk. Peta Visualisasi Similaritas Topik yang ditunjukkan kepada responden ialah peta pada gambar 4.1 yang merupakan visualisasi peta topik dari keseluruhan jurusan yang ada di Institut Teknologi Sepuluh Nopember. Responden disuruh untuk mengukur tingkat kejelasan dan kemenarikan peta visualisasi similaritas topik yang terbentuk. Responden juga dimintai saran dan pendapat mengenai hasil visualisasi peta.

Tahapan-tahapan uji coba untuk mengetahui tingkat kejelasan dan kemenarikan peta bagi pengguna adalah sebagai berikut:

A. Peta Visualisasi Similaritas Topik Penelitian yang terbentuk ditunjukan kepada responden.

B. Responden diberi penjelasan singkat mengenai Peta yang terbentuk.
C. Minta Responden menjawab pertanyaan nomor 1 yakni mengukur tingkat kejelasan dan makna dari peta yang terbentuk.
D. Minta Responden menjawab pertanyaan nomor 2 dan 3 yakni menukur tingkat kemenarikan dan skema warna dari peta yang terbentuk.
E. Minta Responden menyebutkan alasan mengenai jawaban yang diberikan serta kritik dan saran menyangkut peta yang telah terbentuk.

Hasil dari Skenario pertama dapat dilihat pada Tabel 5.4 dibawah

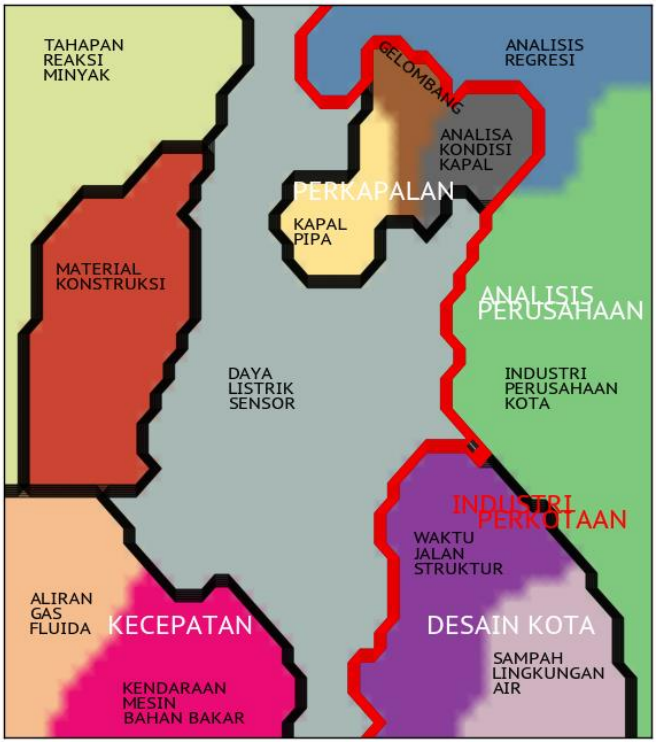
Tabel 5.4 Hasil Kuisioner Skenario Pertama

No	Item	Informasi
1	Apakah peta yang terbentuk memiliki makna dan fungsi yang jelas bagi Anda?	Tidak = 0
		Kurang Jelas = 9
		Jelas = 24
2	Apakah peta yang terbentuk menarik untuk dilihat?	Ya = 27
		Tidak = 6
3	Skema Warna yang lebih baik	Warna-warni = 33
		Monokrom = 0
4	Total jumlah responden	33

Tabel 5.4 menunjukkan bahwa 73% dari responden menilai bahwa peta yang terbentuk memiliki fungsi dan makna yang jelas, begitu juga 81% dari responden menilai bahwa peta yang terbentuk menarik untuk dilihat.

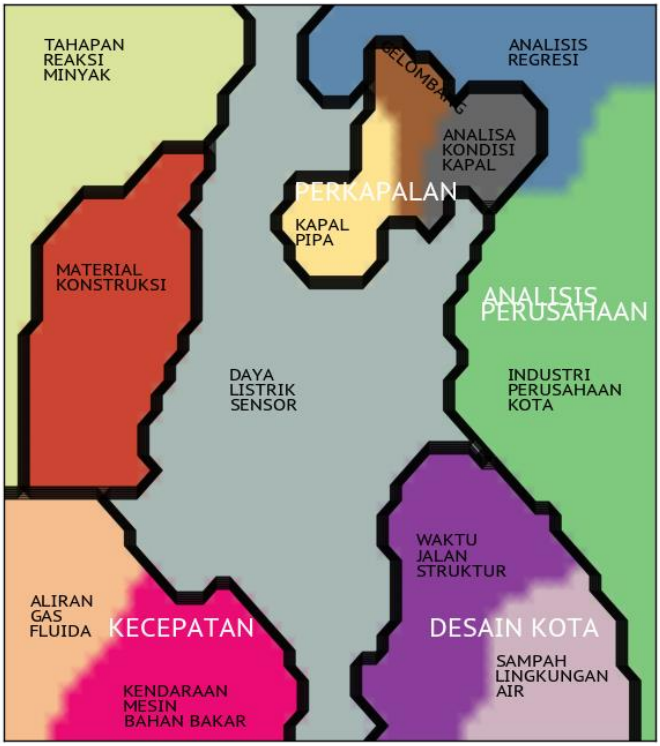
5.6.3.2 Kuisioner Skenario Kedua

Skenario kedua bertujuan untuk mengetahui preferensi pengguna mengenai tingkat kedalaman dari peta yang dihasilkan. Peta dapat memiliki beberapa level kedalaman diukur dari *Cutoff*.



Gambar 5.2 Peta dengan Nilai Cutoff Distance 4000

Distance untuk tiap *cluster* yang dihasilkan. *Cutoff Distance* ialah *threshold* jarak dari hasil metode *Hierarchical Clustering*.



Gambar 5.3 Peta dengan Nilai Cutoff Distance 6000

Responden diminta untuk memilih salah satu dari peta pada gambar 5.2 dan 5.3 guna mengetahui kedalaman level yang optimal menurut pengguna.

Tahapan-tahapan uji coba untuk mengetahui tingkat kedalaman level yang optimal ialah sebagai berikut:

A. Peta Visualisasi pada gambar 5.2 dan 5.3 ditunjukkan kepada responden.
B. Responden dijelaskan singkat mengenai maksud dari garis-garis pembatas dan level dari peta visualisasi topik penelitian.
C. Minta Responden menjawab pertanyaan nomor 1 yakni memvalidasi apakah *Hierarchical Clustering* yang dilakukan bermanfaat bagi peta yang telah terbentuk.
D. Minta Responden menjawab pertanyaan nomor 2 mengenai preferensi kedalaman level dari peta yang dihasilkan.

Hasil dari scenario ini dapat dilihat pada Tabel 5.5

Tabel 5.5 Hasil Kuisioner Skenario Kedua

No	Item	Informasi
1	Apakah peta lebih informatif setelah dikenakan Hierarchical Clustering	Ya = 33
		Tidak = 0
2	Preferensi Cutoff Distance Peta	4000 = 12
		6000 = 21
3	Total jumlah responden	33

Tabel 5.5 menunjukkan bahwa metode *Hierarchical Clustering* dinilai membantu peta visualisasi yang dihasilkan agar lebih informatif dengan 100% Responden menyetujui hal tersebut. Sedangkan untuk preferensi *Cutoff Distance* dari peta yang terbentuk, 63% Responden lebih memilih peta dengan *Cutoff Distance* 6000 dengan alasan peta yang terbentuk lebih informatif.

5.6.3.3 Kuisioner Skenario Ketiga

Skenario ketiga bertujuan untuk mengetahui kejelasan informasi dari label yang dihasilkan pada peta visualisasi topik penelitian yang terbentuk. Responden diminta untuk mengukur tingkat kejelasan label-label yang telah terbentuk pada peta.

Berikut ialah tahapan-tahapan pada scenario kuisioner ketiga:

A. Responden ditunjukkan Peta Visualisasi Similaritas Topik Penelitian dan difokuskan terhadap label dari tiap *cluster*
B. Minta Responden menjawab pertanyaan nomer 1 yang berisikan preferensi jumlah label untuk tiap *cluster.*
C. Minta Responden menjawab pertanyaan nomer 2 yang berisikan tentang kejelasan label untuk tiap *cluster* dan menggambarkan perbedaan untuk tiap *cluster*nya.

Hasil dari scenario ini dapat dilihat pada Tabel 5.6

Tabel 5.6 Hasil Kuisioner Skenario Ketiga

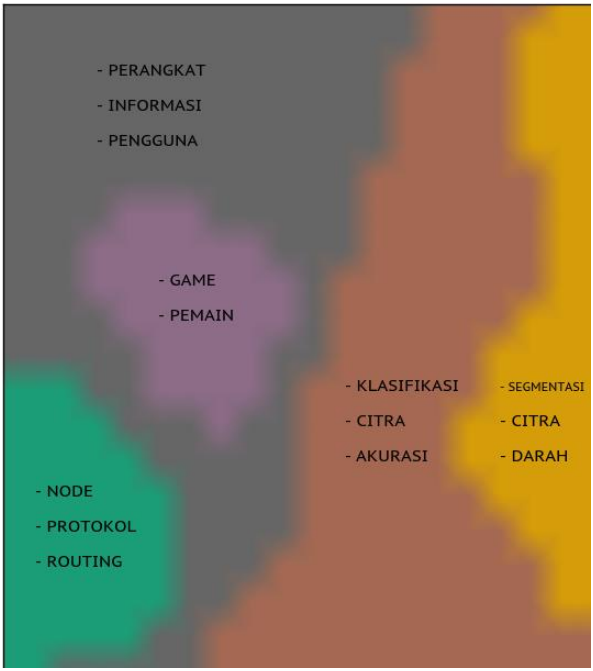
No	Item	Informasi
1	Preferensi jumlah label tiap cluster	1 = 1
		3 = 32
2	Apakah label yang dihasilkan menggambarkan perbedaaan antar *cluster*	Ya = 33
		Tidak = 0
3	Total jumlah responden	33

Tabel 5.6 menunjukkan bahwa responden sudah cukup jelas dengan label-label yang dihasilkan dengan 100% Responden

menilai bahwa label tiap *cluster* sudah membantu membedakan *cluster-cluster* dari Peta Visualisasi yang terbentuk. Responden juga menilai bahwa penggunaan 3 label untuk definisi tiap *cluster* dirasa lebih baik dan informatif.

5.6.3.4 Kuisioner Skenario Keempat

Kuisioner skenario keempat bertujuan untuk mengetahui apakah hasil peta visualisasi lebih baik jika data yang digunakan merupakan data dari rumpun yang spesifik. Responden diminta memilih dan menilai peta visualisasi similaritas yang dirasa lebih jelas dan informatif. Hasil Peta Visualisasi dengan data penelitian dari rumpun Teknik Informatika saja dapat dilihat pada Gambar 5.4



Gambar 5.4 Peta Visualisasi Similaritas Topik Teknik Informatika

Tahapan-tahapan uji coba untuk mengetahui peta yang lebih informatif dan jelas ialah sebagai berikut:

A. Responden ditunjukkan Peta Visualisasi Similaritas Topik keseluruhan dan cakupan Teknik Informatika saja.
B. Responden diberikan penjelasan singkat dari kedua peta tersebut.
C. Responden diminta memilih peta yang dirasa lebih jelas dan informatif.

Hasil dari kuisioner diatas dapat dilihat pada Tabel 5.7 di bawah

Tabel 5.7 Hasil Kuisioner Skenario Keeempat

No	Item	Informasi
1	Preferensi Hasil Peta Visualisasi	Global = 10
		Teknik Informatika = 23

Dari tabel 5.6 dapat disimpulkan bahwa Peta Visualisasi dengan data Teknik Informatika saja dinilai lebih informatif dan jelas oleh responden. Hal ini menunjukkan bahwa metode *learning* yang dipakai pada Tugas Akhir ini berperforma lebih baik untuk data dengan bervariasi lebih sedikit.

5.4 Evaluasi Umum Skenario Uji Coba

Berdasarkan skenario uji coba yang dilakukan untuk memvalidasi ekstraksi fitur dan Teknik *clustering* yang digunakan pada Tugas Akhir ini, dapat dikatakan performa Sistem yang dibuat cukup memuaskan, dengan hasil *Silhouette Score* tertinggi

sebesar 0.5215 dan *Cophenet Correlation Coefficient* sebesar 0.978.

Pada Skenario Uji Coba pertama dilakukan uji coba dalam jumlah *cluster* yang digunakan untuk algoritma *K-means Clustering*, dari hasil uji coba dapat disimpulkan bahwa pembagian ke dalam 12 *cluster* paling tepat untuk dataset *corpus* penelitian Tugas Akhir Institut Teknologi Sepuluh Nopember. Hal ini digambarkan pada hasil uji coba, dimana ketika jumlah *cluster* diturunkan, performa *Silhouette Score* menurun, sedangkan ketika jumlah *cluster* ditingkatkan, performa *Silhouette Score* juga menurun secara konstan.

Pada Skenario Uji Coba yang kedua, didapat nilai *Cophenet Correlation Coefficient* yang tinggi yaitu 0.978 dari 1, dimana hal ini menandakan bahwa ekstraksi fitur yang digunakan untuk mengukur kedekatan antar *cluster* pada *Hierarchical Clustering* sudah merepresentasikan dataset dengan baik. Begitu juga metode *linkage* yang digunakan yaitu *Ward's Method* dinilai paling cocok untuk digunakan pada dataset yang digunakan pada Tugas Akhir ini.

Pada Skenario Uji Coba yang ketiga, kuisioner untuk mengetahui performa Peta Visualisasi Similaritas Topik Penelitian terhadap pengguna dapat dikatakan bahwa pengguna menilai Peta yang terbentuk sudah memiliki fungsionalitas dan informasi yang jelas, namun dirasa masih dibutuhkan penjelasan di awal untuk mengetahui makna dari peta yang terbentuk. Pengguna juga merasa meski peta yang terbentuk terlihat menarik, skema warna untuk tiap *cluster* dirasa masih tidak berarti apa-apa selain merupakan pembatas untuk tiap *cluster*. Pengguna menyarankan agar *cluster* yang berdekatan/mirip memiliki warna yang mirip juga agar warna-warna tersebut memiliki makna.

Pada Kuisioner Skenario kedua, Pengguna menilai bahwa semakin dalam level dari peta yang disajikan lebih baik dan informatif, dan dirasa peta yang terbentuk masih perlu diperdalam levelnya agar informasi yang didapatkan pengguna lebih banyak.

Pengguna juga menilai bahwa Peta dengan data yang spesifik (Teknik Informatika) lebih menggambarkan separasi topik yang lebih jelas untuk tiap *cluster*-nya. Hal ini menunjukkan bahwa data untuk semua jurusan di Institut Teknologi Sepuluh Nopember memiliki persebaran kata-kata yang terlalu luas sehingga Peta Jaringan yang dari metode *Self-organizing Maps* terlalu melebar dan tidak optimal. Hal ini juga diakibatkan karena dataset yang digunakan tidak diperiksa kualitas datanya terlebih dahulu juga, semisal terdapat banyak kata-kata *noise* yang muncul di banyak topik sehingga membuat hasil *training* tidak optimal.

BAB VI
KESIMPULAN DAN SARAN

Bab ini membahas tentang kesimpulan yang didasari oleh hasil uji coba yang telah dilakukan pada bab sebelumnya. Kesimpulan nantinya sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, juga terdapat saran yang ditujukan untuk pengembangan penelitian lebih lanjut di masa depan.

6.1 Kesimpulan

Kesimpulan yang dapat diambil didapatkan berdasarkan hasil uji coba Visualisasi Similaritas Topik Penelitian dengan Pendekatan Kartografi menggunakan *Self-organizing Maps* (SOM) adalah sebagai berikut:

1. Metode TF-IDF dan *Self-organizing Maps* dapat digunakan sebagai metode ekstraksi fitur dan *unsupervised learning* yang baik untuk data teks ditunjukkan dengan hasil nilai *Cophenet Correlation Coefficient* yang tinggi yaitu 0.977.
2. Berdasarkan hasil uji coba, metode *K-means Clustering* pada jaringan SOM yang sudah terlatih menghasilkan nilai rata-rata terbesar untuk penggunaan 12 jumlah *cluster*. Nilai *Silhouette Score* yang dihasilkan untuk jumlah *cluster* 12 adalah 0.5215
3. Implementasi *Term Frequency* (TF) dianggap lebih cocok untuk menentukan label dari tiap *cluster* dibandingkan *Term Frequency – Inverse Document Frequency* (TF-IDF) dikarenakan label yang dihasilkan lebih merepresentasikan tiap karakteristik dari *cluster.*
4. Teknik *K-means Clustering* dan *Hierarchical Clustering* dinilai mampu menyderhanakan jaringan peta SOM yang telah terbentuk sehingga lebih mudah divisualisasikan. Uji Validasi algoritma *clustering* diatas menggunakan

Silhouette Scoring dengan nilai 0.5215 dan *Cophenet Correlation Coefficient* dengan nilai 0.977.

5. Peta Visualisasi Similaritas Topik Penelitian yang dihasilkan dinilai sudah memberikan makna dan informasi yang jelas dengan 73% Responden menyatakan demikian. Peta Visualisasi yang terbentuk juga dinilai sudah menarik untuk dilihat oleh 81% Responden.
6. Metode *Hierarchical Clustering* dinilai bermanfaat oleh 100% Responden dalam memberikan informasi tambahan pada peta yang telah terbentuk.
7. Peta Visualisasi yang terbentuk dari data penelitian rumpun yang lebih spesifik dinilai memberikan separasi antar topik lebih jelas oleh 69% dari Responden. Hal ini menunjukkan bahwa metode *Self-organizing Maps* memiliki performa yang lebih baik ketika dataset memiliki lebih sedikit variasi.

6.2 Saran

Saran yang diberikan terkait pengembangan pada Tugas Akhir ini adalah:

1. *Rapid Automatic Keyword Extraction* (RAKE) dapat digunakan untuk penentuan *keyword* dari *corpus* selain dari judul dokumen.
2. Menggunakan n-grams *counting* untuk metode penentuan label untuk tiap *cluster* yang terbentuk.
3. Menemukan cara peletakan label otomatis yang lebih baik dibanding metode *Centroid Mass* yang dipakai pada Tugas Akhir ini.
4. Skema pewarnaan diaplikasikan dengan lebih jelas dan bermakna sehingga *Cluster* yang mirip/berdekatan memiliki warna yang mirip juga.

DAFTAR PUSTAKA

[1] Ristek Dikti, "http://www1.ristek.go.id/?module=News%20News&id=8705," [Online].

[2] Ristek Dikti, "http://www.dikti.go.id/kolokium-di-australia-kerjasama-antar-penelitisemakin-dibutuhkan-di-indonesia/," [Online].

[3] A. Skupin, "A Cartographic Approach to Visualizing Conference Abstracts,," 2013.

[4] A. B. L. A. F. P. d. K. R. U. T. V. Nguyen, "SOM-based Data Visualization Methods," 1999..

[5] Lucid Chart, "https://www.lucidchart.com/pages/concept-map," [Online].

[6] S. F. a. B. Buttenfield, "Formalizing Semantic Spaces for Information Access," dalam *Annals of the Assoc. American Geographers*, June 2001, pp. vol. 91, no. 2 pp. 263-280.

[7] J. Wise, ""Visualizing the Non-Visual: Spatial Analysis and Interaction with Information from Text DocumentsProc," dalam *IEEE Symp. Information Visualization 1995 (InfoVis 95)*, Los Alamitos, California, IEEE CS Press, 1995, pp. pp. 51-58.

[8] X. Lin, "Visualization for the Document Space," dalam *Proc. IEEE Visualization 92*, Los Alamitos, California, IEEE CS Press, 1992, pp. pp. 274-281.

[9] T. Kohonen dan T. Honkela, "Kohonen network Scholarpedia," 2011. [Online]. [Diakses 24 09 2016].

[10] J. A. Bullinaria, "Self Organizing Maps: Fundamentals," 2004.

[11] A. A. Akinduko, E. M. Mirkes, dan A. N. Gorban, "SOM: Stochastic initialization versus principal components," *Science Direct*, 2015.

[12] I. Valova, G. Georgiev, N. Gueorguieva, dan J. Olson, "Initialization Issues in Self-organizing Maps," *Science Direct*, 2013.

[13] W. Natita, W. Wiboonsak, dan S. Dusadee, "Appropriate Learning Rate and Neighborhood Function of Self-organizing Map (SOM) for Specific Humidity Pattern Classification over Southern Thailand," 2016.

[14] M. Nocker, F. Morchen, dan A. Ultech, "An algorithm for fast and reliable ESOM learning," 2011.

[15] J. Ramos, "Using TF-IDF to Determine Word Relevance in Document Queries," 2007.

[16] A. K. Jain, M. N. Murty, dan P. J. Flynn, "Data Clustering: A Review," dalam ACM Computing Surveys (CSUR), Volume 31, Issue 3, ACM Press, New York, pp. 264-323

[17] M. A., Gao dan R.X, "PCA-Based Feature Selection Scheme for Machine Defect Classification," *IEEE*, vol. 53, 2004.

[18] Scikit-learn, "http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html," [Online].

[19] S. Saracli, "Comparison of Hierarchical Cluster Analysis methods by cophenetic correlation," *Journal of Inequalities and Applications*, 2013

[20] H. Couclelis, "Worlds of Information: The Geographic Metaphor in the Visualization of Complex Information," Cartography and Geographic Information Systems, vol. 25, no. 4, Oct. 1998, pp. 209-220

LAMPIRAN

A. Data yang Digunakan pada Tugas Akhir

1. Jumlah Data untuk Tiap Fakultas

FAKULTAS	JUMLAH DOKUMEN
FMIPA	2194
FTI	5360
FTSP	3078
FTK	1156
FTIF	1250

B. Hasil Validasi terhadap metode Clustering

1. **Validasi Silhouette Score untuk tiap jumlah Cluster pada K-means Clustering**

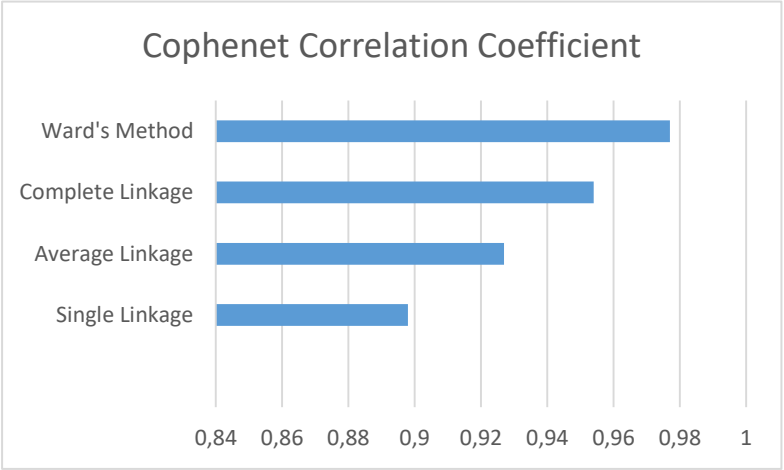
X (Jumlah Cluster)	Cluster	Silhouette Score Tiap Cluster
	1	0.554
	2	0.372
	3	0.532
	4	0.543
	5	0.522
	6	0.398
	7	0.482
	8	0.373
	9	0.523
10	10	0.498
Average		0.4797

X (Jumlah Cluster)	Cluster	Silhouette Score Tiap Cluster
	1	0.538
	2	0.539
	3	0.503
	4	0.496
	5	0.514
	6	0.507
	7	0.507
	8	0.51
	9	0.541
	10	0.476
	11	0.583
12	12	0.544
Average		0.5215

X (Jumlah Cluster)	Cluster	Silhouette Score Tiap Cluster
	1	0.409
	2	0.428
	3	0.436
	4	0.409
	5	0.456
	6	0.451
	7	0.391
	8	0.417
	9	0.415
	10	0.437
	11	0.38
	12	0.404
	13	0.398
	14	0.406
15	15	0.448
Average		0.419

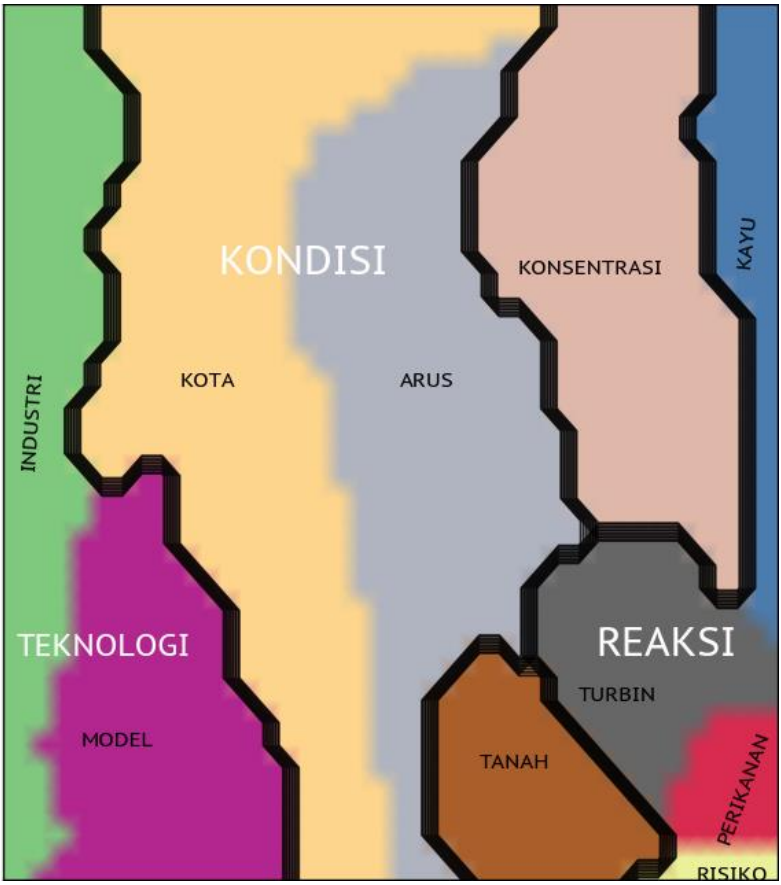
X (Jumlah Cluster)	Cluster	Silhouette Score Tiap Cluster
	1	0.03
	2	0.004
	3	0.052
	4	0.054
	5	0.022
	6	0.023
	7	0.016
	8	0.047
	9	0.029
	10	0.102
	11	0.056
	12	0.066
	13	0.053
	14	0.045
	15	0.039
	16	0.09
	17	0.145
	18	0.074
	19	0.023
20	20	0.08
Average		0.0525

2. Hasil Validasi Hierarchical Clustering

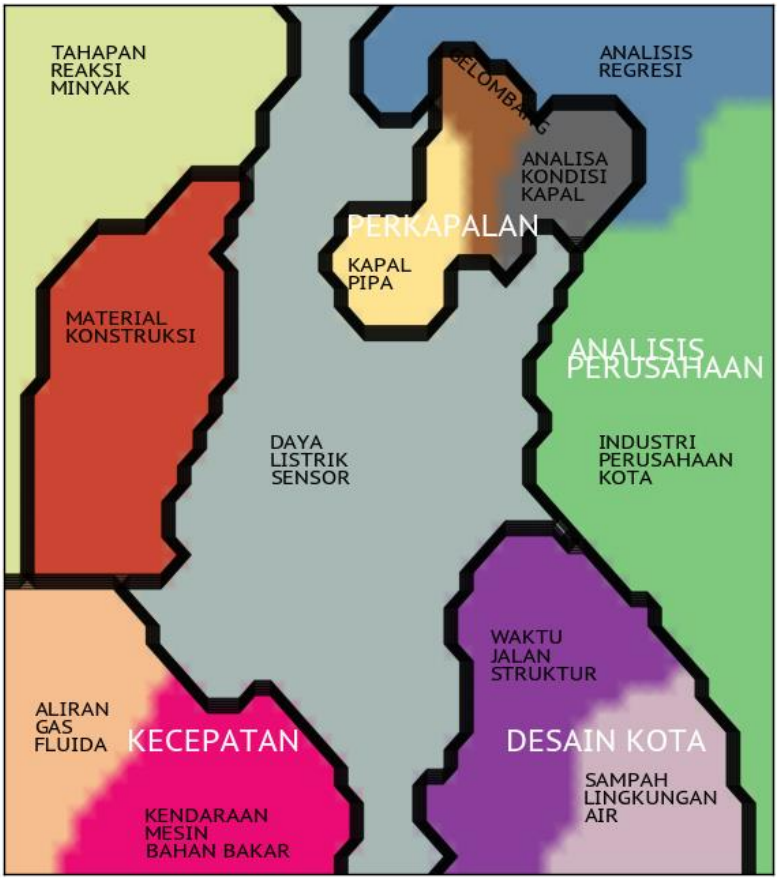


C. Hasil Uji Coba

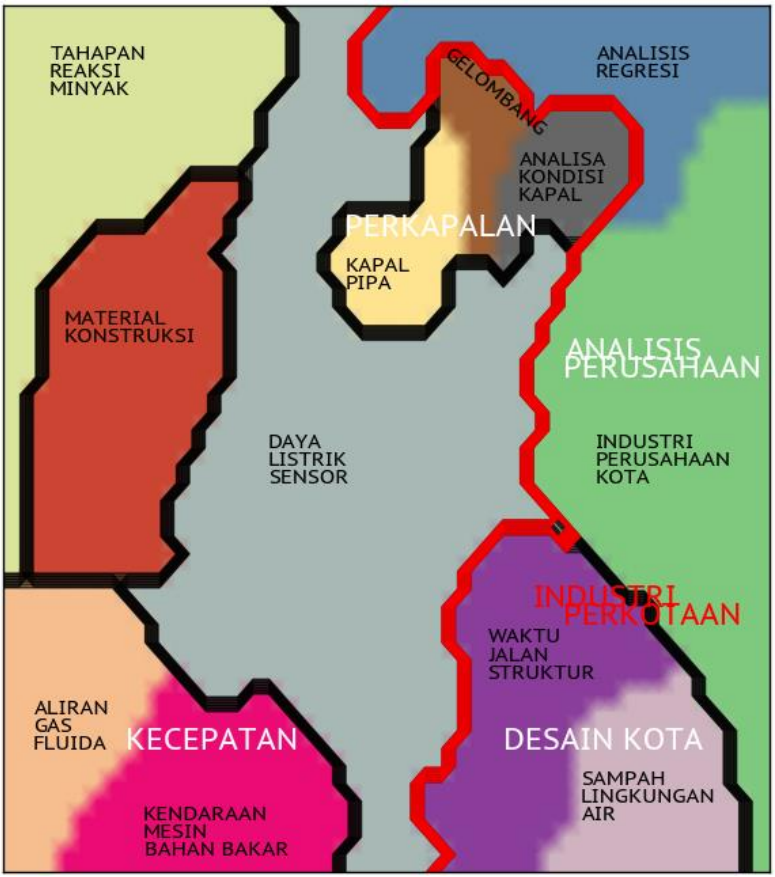
1. Hasil Visualisasi dengan 10 Cluster


KONDISI
KONSENTRASI
KAYU
INDUSTRI
KOTA
ARUS
TEKNOLOGI
REAKSI
TURBIN
MODEL
TANAH
PERIKANAN
RISIKO

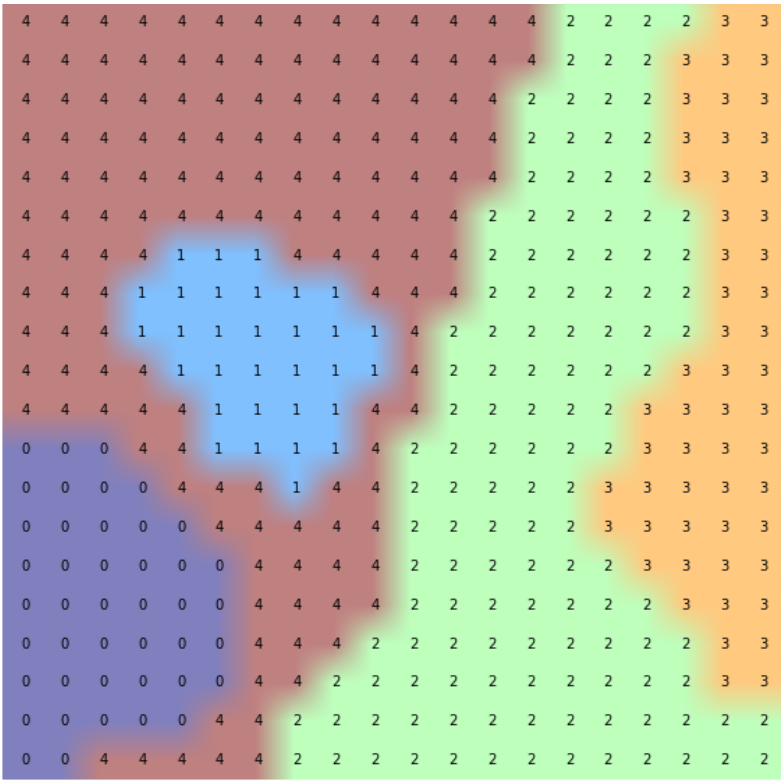
2. **Hasil Visualisasi dengan 12 Cluster dengan Cut-off Distance 4000**



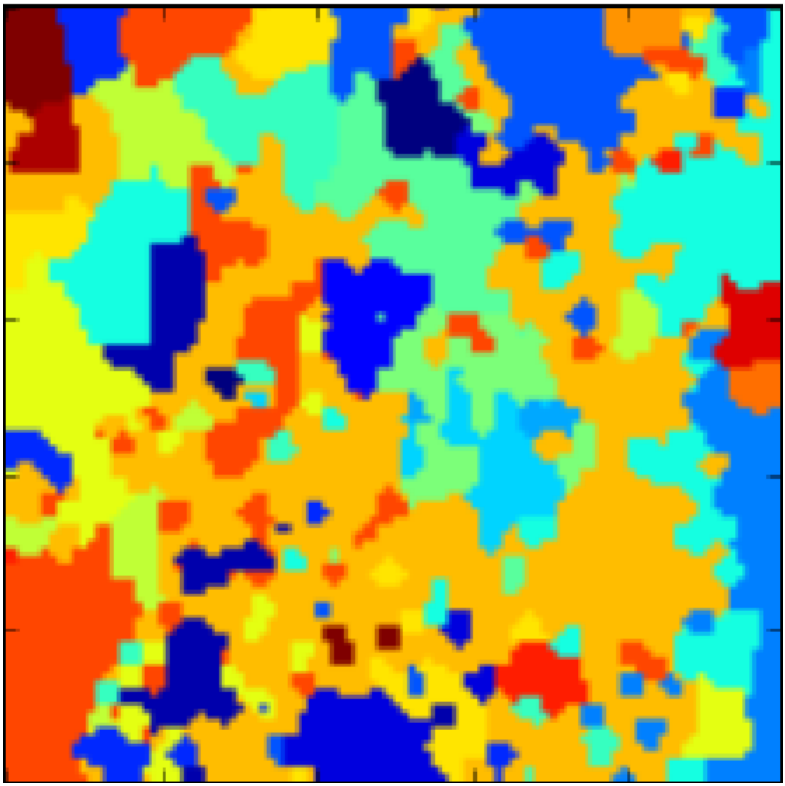
3. **Hasil Visualisasi dengan 12 Cluster dengan Cut-off Distance 6000**


TAHAPAN
REAKSI
MINYAK
GELOMBANG
ANALISIS
REGRESI
ANALISA
KONDISI
KAPAL
PERKAPALAN
KAPAL
PIPA
MATERIAL
KONSTRUKSI
ANALISIS
PERUSAHAAN
DAYA
LISTRIK
SENSOR
INDUSTRI
PERUSAHAAN
KOTA
INDUSTRI
PERKOTAAN
WAKTU
JALAN
STRUKTUR
ALIRAN
GAS
FLUIDA
KECEPATAN
DESAIN KOTA
SAMPAH
LINGKUNGAN
AIR
KENDARAAN
MESIN
BAHAN BAKAR

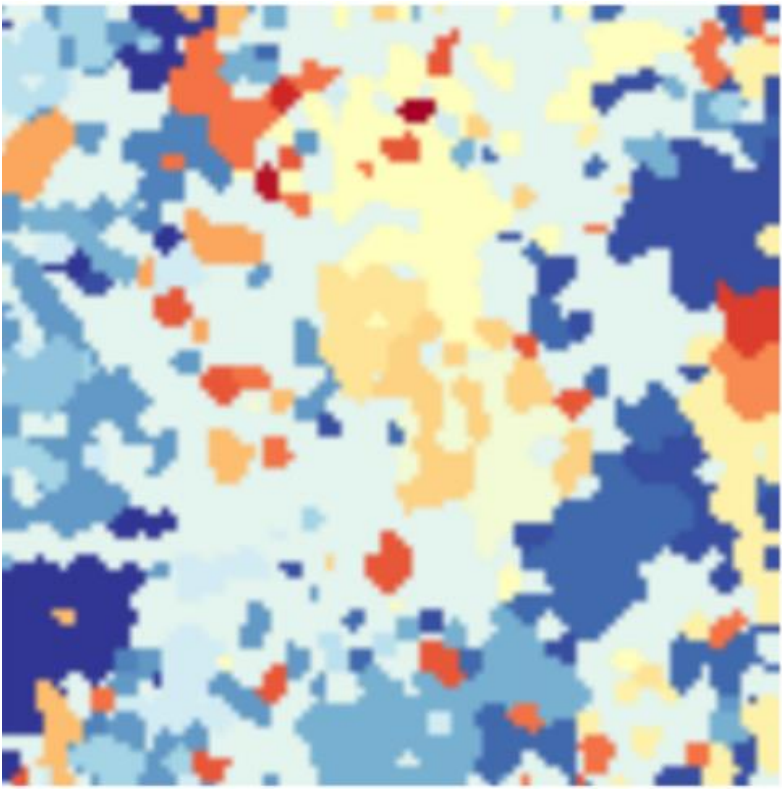
4. **Hasil Clustering dari K-means untuk 5 Cluster**



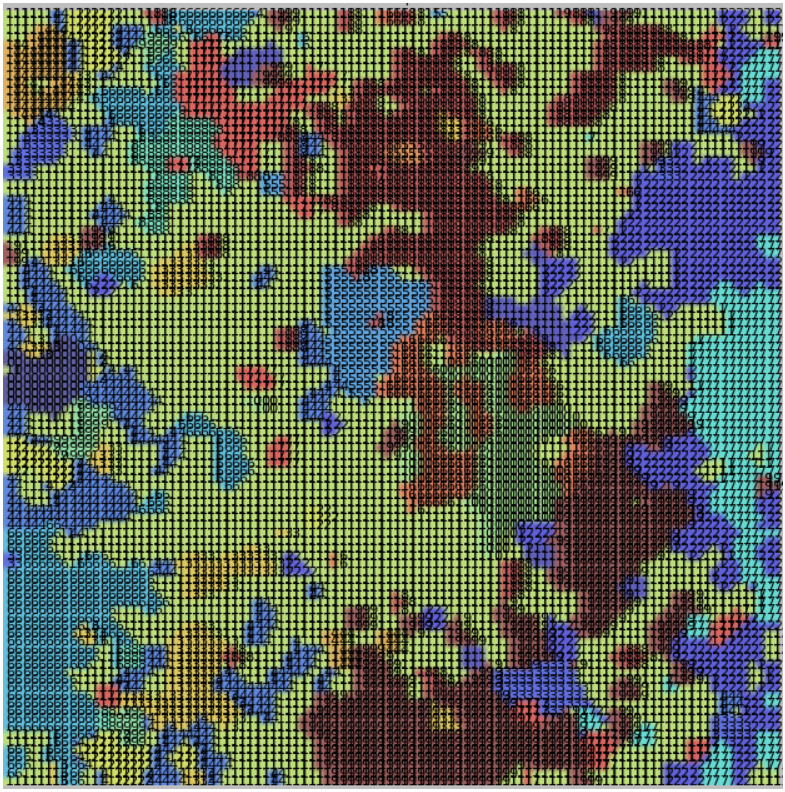
5. **Hasil SOM untuk Mapsize 100x100**



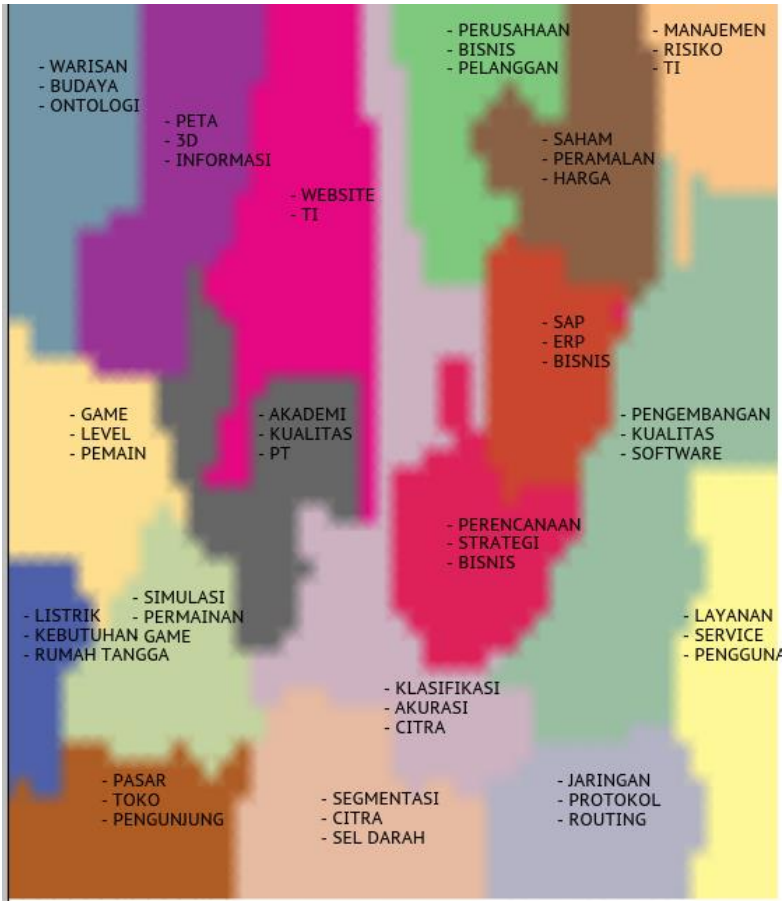
6. **Hasil SOM untuk Mapsize 40x40**



7. **Hasil SOM untuk Mapsize 75x75**



8. **Hasil Visualisasi Peta Similaritas Topik Penelitian tanpa garis batas Cluster**



(Halaman sengaja dikosongkan)

BIODATA PENULIS

Budi Pangestu Tanujaya, lahir di Surabaya pada tanggal 11 Agustus 1995. Penulis merupakan anak ketiga dari tiga bersaudara pasangan Bapak Lucky Tanujaya dan Joo Melia. Penulis menempuh pendidikan formal dimulai dari TK St. Carolus Surabaya (1999-2001), SD St. Carolus Surabaya (2001-2007), SMPK Angelus Custos Surabaya (2007-2010), SMAK Frateran Surabaya (2010-2013) dan S1 Teknik Informatika ITS (2013-2017). Bidang studi yang diambil oleh penulis pada saat berkuliah di Teknik Informatika ITS adalah Komputasi Cerdas dan Visi (KCV). Selama masa kuliah, penulis aktif dalam organisasi kemahasiswaan yaitu Himpunan Mahasiswa Teknik Computer-Informatika (2014-2015) dan juga SCHEMATICS 2014-2015. Penulis dianugerahi kepercayaan untuk menjadi salah satu dari Bukalapak Scholarship Awardee pada tahun 2016-2017. Semasa kuliah, Penulis memiliki ketertarikan pada bidang Web Development, Data Mining, dan juga Competitive Programming. Penulis sempat menjadi asisten dosen selama menjalani perkuliahan di Teknik Informatika yaitu asisten dosen mata kuliah Pemrograman Web, Analsis dan Perancangan Sisetm Informasi, Pengolahan Citra Digital, dan Manajemen Proyek Perangkat Lunak. Penulis memiliki ketertarikan dalam hal *travelling*, *gaming*, musik dan menyukai sesuatu hal yang baru. Komunikasi dengan penulis dapat melalui email: **budi.pangestu.t@gmail.com**