# BAB IV HASIL PENELTIAN DAN PEMBAHASAN

Pada bab ini penulis menguraikan hasil dari implementasi dan uji coba sistem. Bab ini terdiri dari pengumpulan data, pengembangan model, implementasi sistem, hasil implementasi, panduan penggunaan aplikasi, rancangan eksperimen, analisis dan hasil eksperimen.

## Pengumpulan Data

Penelitian ini menggunakan data historis nilai tukar mata uang asing terhadap rupiah (kurs), data klimatologi, data kualitas udara. Data-data tersebut dipilih berdasarkan jenis data tersebut yaitu *time series* dengan format waktu “mm/dd/yyyy hh:mm”, selain itu juga digunakan beberapa data dari penelitian sebelumnya, guna untuk membandingkan hasil dari sistem yang akan dikembangkan dengan sistem yang dikembangkan sebelumnya. Maka dari itu, pembahasan lebih detil akan dipaparkan pada sub-bab selanjutnya.

### Data Nilai Tukar Mata Uang Asing

Data diperoleh dari situs Kementrian Perdagangan Republik Indonesia (http://www.kemendag.go.id). Data berisikan nilai tukar mata uang asing terhadap mata uang rupiah atau kurs dengan rentang per bulan, mulai dari 1 Januari 2001 sampai dengan 1 Juli 2018 dengan jumlah data sebanyak 211 baris. Untuk mengakses data yang tersedia, penulis mengunjungi situs Kemenag RI lalu memilih menu statistik yang berada pada bagian bawah situs, lalu penulis memilih menu nilai tukar rupiah dan memilih menu export to excel, atau bisa diakses melalui alamat (www.kemendag.go.id/id/economic-profile/economic-indicators/exchange-rates).

Setelah melakukan serangkaian proses untuk mengakses data, data yang didapatkan berbentuk tabel dengan jumlah baris sebanyak 211 baris, dan jumlah atribut sebanyak 10 atribut, dengan nilai atribut mewakili nilai tukar mata uang tersebut terhadap rupiah. Berikut adalah atribut-atribut yang terdapat pada data tersebut:

* Waktu data
* Dolar Amerika (USD)
* Yen Jepang (JPY)
* Poundsterling Inggris (GBP)
* Franc Swiss (CHF)
* Dolar Singapore (SGD)
* Riggit Malaysia (MYR)
* Dolar Hongkong (HKD)
* Dolar Austrailia (AUD)
* Dolar Canada (CAD)

Untuk lebih lengkapnya data dapat dilihat pada lampiran, namun penulis mengutip beberapa data yang disajikan pada tabel 4.1, seperti yang sudah dijelaskan sebelumnya data berisikan beberapa atribut yang merepresentasikan nilai tukar mata uang asing terhadap rupiah.

Tabel 4.1 Data Nilai Tukar Mata Uang Asing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| DateTime | USD | JPY | GBP | CHF | ... | CAD |
| 01/01/2001 00:00 | 9.45000 | 8.13149 | 13.81498 | 5.74364 | ... | 6.28451 |
| 02/01/2001 00:00 | 9.83500 | 8.45297 | 14.17963 | 5.85644 | ... | 6.43274 |
| 03/01/2001 00:00 | 10.40000 | 8.37000 | 14.85227 | 6.01400 | ... | 6.60907 |
| 04/01/2001 00:00 | 11.67500 | 9.42066 | 16.74548 | 6.76774 | ... | 7.56988 |
| 05/01/2001 00:00 | 11.05800 | 9.21733 | 15.76486 | 6.21098 | ... | 7.15591 |
| ... | ... | ... | ... | ... | ... | ... |
| 03/01/2018 00:00 | 13.75600 | 12.90553 | 19.36503 | 14.38763 | ... | 10.64665 |
| 04/01/2018 00:00 | 13.87700 | 12.71895 | 19.11904 | 14.04698 | ... | 10.80554 |
| 05/01/2018 00:00 | 13.95100 | 12.83973 | 18.55344 | 14.11117 | ... | 10.83194 |
| 06/01/2018 00:00 | 14.40400 | 13.03707 | 18.83468 | 14.44230 | ... | 10.86439 |
| 07/01/2018 00:00 | 14.41300 | 12.98294 | 18.91203 | 14.59250 | ... | 11.03346 |

### Data Klimatologi

Data ini diperoleh dari stasiun pemantuan milik lembaga meteorologi dan klimatologi Galicia yaitu MeteoGalicia yang berlokasi di kota Mabegondo, Provinsi A Coruna, daerah komUnitas otonom Galicia, Spanyol. Data ini berisikan data klimatologi dengan rentang per hari mulai dari tanggal 06 Juli 2016 sampai dengan 6 Juli 2017 sebanyak 365 baris data. Data ini merupakan data yang digunakan pada penelitian sebelumna, yaitu DWP yang dibahas pada bab 2 sebelumnya. Parameter-parameter yang terdapat pada data ini diantaranya:

Waktu data

Cakupan awan rata-rata per hari dengan satuan % (Persen)

Suhu rata-rata per hari dengan satuan oC (derajat celcius)

Kecepatan angin rata-rata per hari dengan satuan km/h (Kilometer per jam)

Arah angin rata-rata per hari dengan satuan o (derajat)

Curah hujan rata-rata per hari dengan satuan L/m2 (Liter per meter kuadrat)

Parameter tersebut merupakan hasil analisis pada penelitian DWP sebelumnya (Putra dkk., 2017). Untuk lebih lengkapnya data dapat dilihat pada lampiran dan untuk kutipan datanya bisa dilihat pada tabel 4.2.

Tabel 4.2 Kutipan data klimatologi

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| DateTime | CloudCoverage | Temperature | WindSpeed | WindDirection | Rainfall |
| 07/06/2016 00:00 | 40.8 | 21.3 | 5.47 | 315 | 0 |
| 07/07/2016 00:00 | 20.9 | 20.1 | 6.41 | 315 | 0 |
| 07/08/2016 00:00 | 27.2 | 19.5 | 7.02 | 315 | 0 |
| 07/09/2016 00:00 | 23.2 | 19.1 | 5.94 | 315 | 0 |
| 07/10/2016 00:00 | 77.5 | 18.7 | 5.44 | 180 | 0 |
| ... | ... | ... | ... | ... | ... |
| 07/02/2017 00:00 | 12.6 | 18.9 | 7.34 | 315 | 0 |
| 07/03/2017 00:00 | 13.3 | 21.8 | 4.86 | 315 | 0 |
| 07/04/2017 00:00 | 18.7 | 24 | 6.59 | 225 | 0 |
| 07/05/2017 00:00 | 81.1 | 19.2 | 8.35 | 225 | 0 |
| 07/06/2017 00:00 | 58.3 | 17.9 | 6.48 | 315 | 0 |

### Data Kualitas Udara

Data ini hampir mirip dengan data klimatologi sebelumnya, data ini diperoleh dari stasiun pemantuan milik lembaga meteorologi dan klimatologi Galicia yaitu MeteoGalicia. Data ini berisikan nilai-nilai partikel yang terkandung dalam udara dengan rentang per hari mulai dari tanggal 06 Juli 2016 sampai dengan 6 Juli 2017 sebanyak 365 baris data. Parameter-parameter yang terdapat pada data ini diantaranya:

Data Karbon Monoksida (CO) per hari dengan satuan ppm (Part per Million)

Data Nitrogen Monoksida (NO) per hari dengan satuan ppm (Part per Million)

Data Nitrogen Dioksida (NO2) per hari dengan satuan ppm (Part per Million

Data Nitrogen Oksida (NOX) per hari dengan satuan ppm (Part per Million)

Data Ozone (O3) per hari dengan satuan ppm (Part per Million)

Data Particulate Matter 10 mikronmeter per hari

Data Particulate Matter 25 mikronmeter per hari

Data Sulfur Dioksida (SO2) per hari dengan satuan ppm (Part per Million)

Data ini juga digunakan pada penelitian DWP sebelumnya (Putra dkk., 2017). Tabel 4.3 merupakan kutipan data yang digunakan selama penelitian, untuk 366 baris data secara lengkap dapat dilihat pada lampiran.

Tabel 4.3 Kutipan data Kualitas Udara

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DateTime | CO | NO | NO2 | NOX | O3 | PM10 | PM25 | SO2 |
| 07/06/2016 00:00 | 0.13 | 3 | 15 | 19 | 51 | 18 | 10 | 1 |
| 07/07/2016 00:00 | 0.11 | 1 | 10 | 10 | 56 | 14 | 7 | 1 |
| 07/08/2016 00:00 | 0.1 | 1 | 8 | 8 | 59 | 12 | 8 | 1 |
| 07/09/2016 00:00 | 0.1 | 2 | 10 | 11 | 57 | 12 | 7 | 1 |
| 07/10/2016 00:00 | 0.11 | 2 | 10 | 12 | 53 | 12 | 11 | 1 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 07/02/2017 00:00 | 0.17 | 2 | 35 | 37 | 10 | 14 | 20 | 7 |
| 07/03/2017 00:00 | 0.17 | 23 | 43 | 78 | 1 | 11 | 14 | 9 |
| 07/04/2017 00:00 | 0.17 | 31 | 42 | 90 | 1 | 0 | 14 | 9 |
| 07/05/2017 00:00 | 0.18 | 32 | 41 | 90 | 1 | 0 | 12 | 7 |
| 07/06/2017 00:00 | 0.18 | 32 | 61 | 90 | 1 | 0 | 12 | 7 |

### Data Partikel Udara Kota Beijing

Data ini hampir mirip dengan data kualitas udara, data ini diperoleh dari stasiun pemantuan milik lembaga meteorologi dan klimatologi kota Beijing. Data ini berisikan nilai-nilai partikel yang terkandung dalam udara dengan rentang per jam selama 2 tahun penuh mulai dari tanggal 01 Januari 2010 00:00 sampai dengan 31 Desember 2011 23:00 sebanyak 17521 baris data. Parameter-parameter yang terdapat pada data ini diantaranya:

* Data Konsentrasi Particulate Matter 2.5 mikronmeter (ug/m^3)
* DEWP: Dew Point atau titik embun
* TEMP: Temperature atau suhu
* PRES: Pressure atau tekanan (hPa)
* cbwd: Combined wind direction atau arah angin
* Iws: Cumulated wind speed atau kecepatan angin (m/s)
* Is: Cumulated hours of snow atau akumulasi jam salju
* Ir: Cumulated hours of rain atau akumulasi jam hujan

Tabel 4.4 merupakan kutipan data yang digunakan selama penelitian, untuk 17521 baris data secara lengkap dapat dilihat pada lampiran.

Tabel 4.4 Kutipan data Partikel Udara Kota Beijing

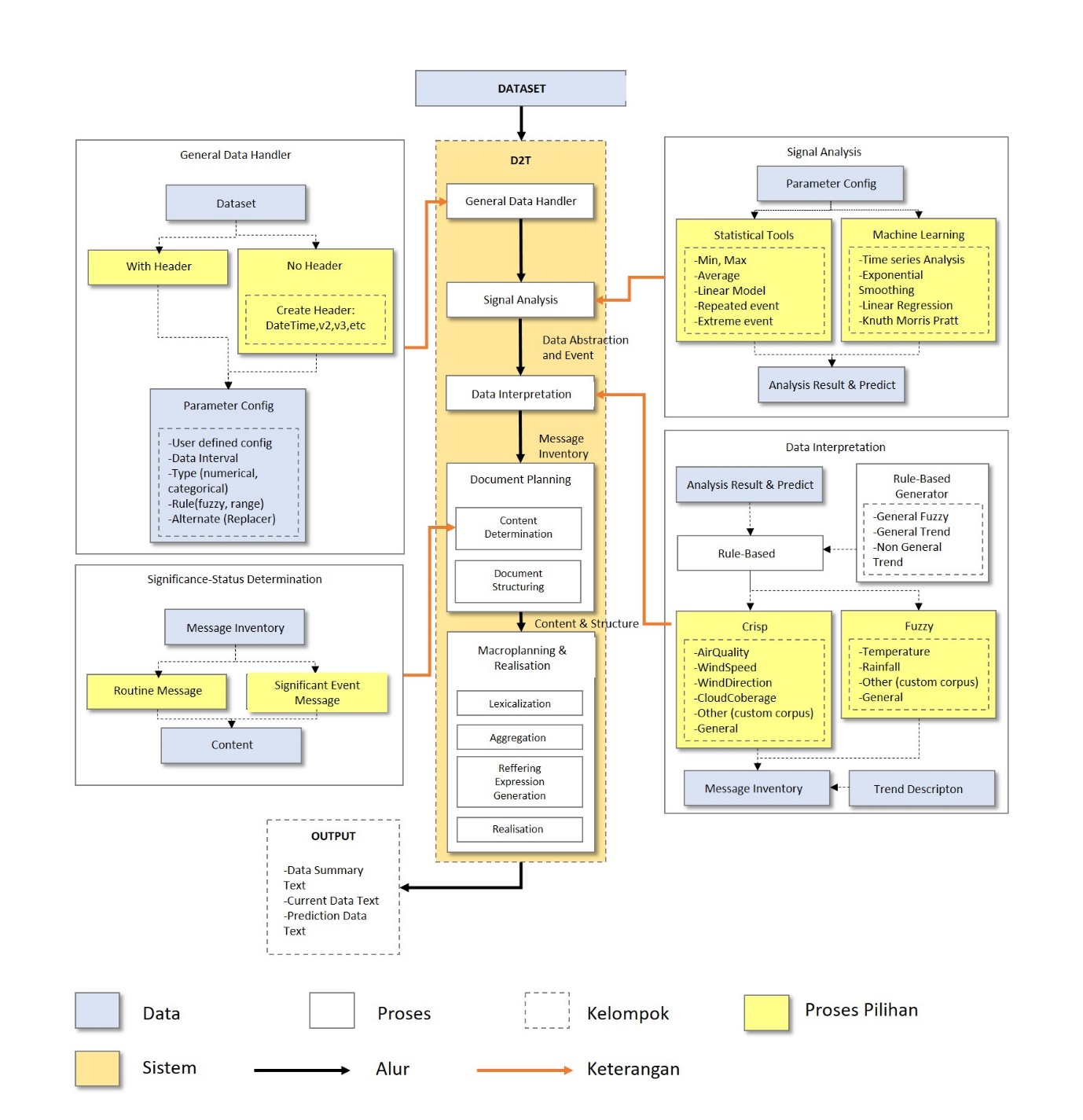
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DateTime | PM2.5 | DEWP | TEMP | PRES | CBWD | LWS | IS | IR |
| 01/01/2010 00:00 | 138 | -21 | -11 | 1021 | NW | 1.79 | 0 | 0 |
| 01/01/2010 01:00 | 125 | -21 | -12 | 1020 | NW | 4.92 | 0 | 0 |
| 01/01/2010 02:00 | 249 | -21 | -11 | 1019 | NW | 6.71 | 0 | 0 |
| 01/01/2010 03:00 | 210 | -21 | -14 | 1019 | NW | 9.84 | 0 | 0 |
| 01/01/2010 04:00 | 98 | -20 | -12 | 1018 | NW | 12.97 | 0 | 0 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 01/31/2010 19:00 | 54 | -15 | -1 | 1025 | NE | 2.68 | 0 | 0 |
| 01/31/2010 20:00 | 71 | -13 | -1 | 1025 | cv | 0.89 | 0 | 0 |
| 01/31/2010 21:00 | 129 | -8 | -1 | 1026 | SE | 5.81 | 0 | 0 |
| 01/31/2010 22:00 | 145 | -7 | -2 | 1027 | SE | 10.73 | 0 | 0 |
| 01/31/2010 23:00 | 101 | -7 | -2 | 1027 | SE | 13.86 | 0 | 0 |

## Pengembangan Model *Data-to-*text untuk Kasus Data *General* dengan *Fuzzy Rule-based* dan *Machine Learning*

Model sistem *Data-to-Text* pada penelitian ini mengacu pada arsitektur sistem *Data-to-Text* yang dikembangkan oleh Reiter (2011), dan beberapa bagian yang dikembangkan oleh Putra (2017). Dengan menggunakan konsep *Fuzzy Rule-based* dan *Machine Learning* pada bagian *Signal Analysis* dan *Data Interpretation*, sistem yang dibangun dapat menerima masukan berupa data *general* atau data yang berasal dari bidang apapun baik data tersebut mempunyai informasi berupa *header*, tipe data, *rule* maupun tidak.

Keluaran yang dihasilkan oleh sistem ini berupa berita yang terdiri dari tiga bagian, bagian pertama merupakan deskripsi dari ringkasan data selama periode tertentu, bagian ke-dua merupakan deskripsi dari baris terakhir pada *n* buah data, bagian ke-tiga merupakan informasi dari hasil prediksi untuk baris data ke *n+1.*

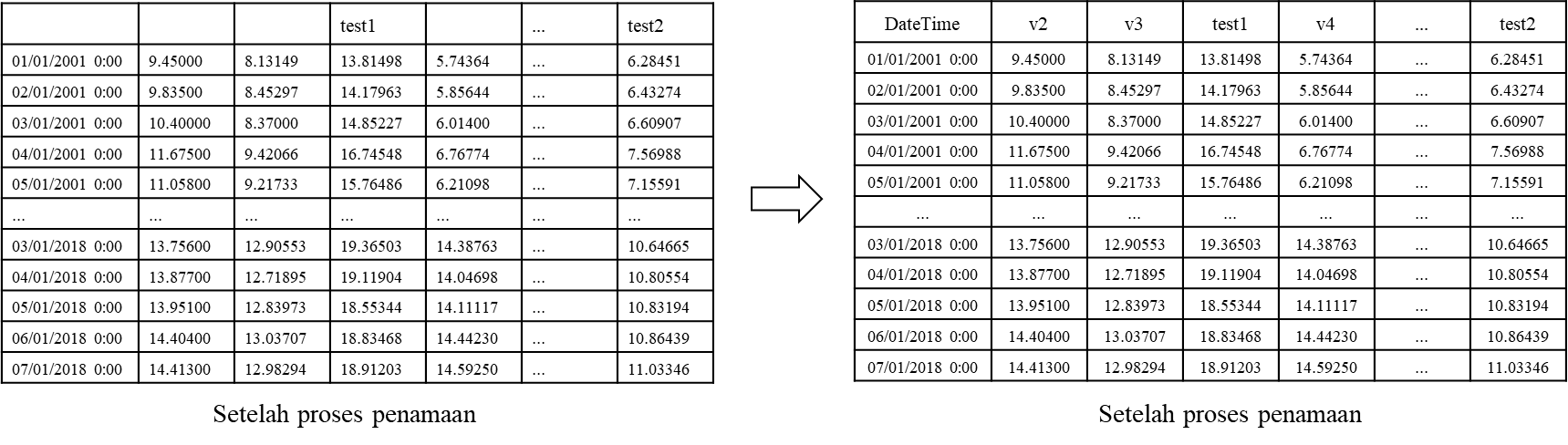
Model penelitian pada Gambar 4.1 memperlihatkan perbedaan yang terdapat pada penelitian ini yaitu terdapatnya bagian *General Data Handler* dan pengaplikasian *Machine Learning* dan *Fuzzy Rule-based* pada bagian *Signal Analysis* dan *Data Interpretation*. Saat pertama kali data masuk, data akan diproses pada bagian *General Data Handler*, bagian ini akan mengelompokan apakah data memiliki informasi berupa *header* atau tidak, lalu sistem akan secara otomatis melihat apakah sudah terdapat konfigurasi untuk setiap parameter sesuai dengan *header* parameter tersebut, jika tidak terdapat konfigurasi maka sistem akan membuat konfigurasi dengan nilai *default* sehingga sistem dapat memproses data *general*. Perbedaan lain terdapat pada bagian *Signal Analysis*, pada bagian ini digunakan *Machine Learning* dan *Time-Series Analysis* untuk mendapatkan pola diskret yang akan digunakan sebagai masukan untuk tahap *Data Interpretation*. Pola yang sudah didapatkan pada tahap *Signal Analysis* dimana pada proses *Data Interpretation* data akan diinterpretasikan sesuai tipe *Rule-based* yang terdapat pada konfigurasi parameter, dimana ada dua pilihan *Rule-based* untuk menginterpretasikan data yaitu, *Fuzzy* dan *Crisp.* Lalu dilanjutkan dengan serangkaian proses sehingga dihasilkan teks berita, serangkaian proses terebut dapat dilihat secara utuh pada kotak berwarna oranye.

Gambar 4.1 Model Data-to-text untuk Data *General*

### Model Proses General Data Handler

Proses *General Data Handler* bertujuan untuk pengolahan dan pemrosesan data, tahapan ini bertujuan agar dataset yang akan diproses menjadi *general* sebelum digunakan untuk proses selanjutnya. Jika pada penelitian sebelumnya, sistem D2T hanya mampu menerima masukan berupa data yang terbatas pada suatu bidang, seperti data klimatologi, keuangan, dan lain-lain. Maka pada penelitian inim, sistem D2T yang dibangun mampu menerima berbagai inputan data, dari bidang apapun, bahkan untuk data yang tidak dapat dikenali ataupun yang tidak memiliki *header* sekalipun.

DalamPertama-tama data yang masuk akan dilihat terlebih dahulu, apakah dataset yang masuk setiap parameternya memiliki *header* atau tidak. Jika semua atau beberapa parameter pada suatu dataset ada yang tidak memiliki *header*, sistem akan memberi *header*  pada parameter tersebut secara otomatis. Khusus untuk *header pada* parameter atau kolom pertama dari dataset, sistem akan mengubah *header* menjadi DateTime. Untuk parameter selanjutnya yang tidak memiliki *header*, sistem akan menamai *header* dengan nama *v2, v3, v4,* dan seterusnya. Untuk lebih lengkapnya, bisa dilihat pada gambar 4.2 .



Gambar 4.2 Penamaan *header* pada Proses *General Data Handler*

Setelah penamaan *header*, proses selanjutnya yaitu konfigurasi parameter. Konfigurasi parameter merupakan aturan atau cara bagaimana suatu parameter akan diolah dan diproses dalam sistem. Konfigurasi parameter disimpan pada file *mainconfig.csv* yang terdapat pada folder *Config*. Terdapat 3 konfigurasi yang disimpan, yatu:

1. *Type,* konfigurasi ini akan menentukan bagaimana suatu parameter diproses. Terdapat dua kategori yaitu, *numerical*  dan *categorical*. Untuk parameter dengan tipe *numerical*, sistem akan melakukan proses analisis data menggunakan *Statistical Tools* seperti *min, max, average,* dan penerapan *Machine Learning* seperti *time-series analysis, regression, exponential smoothing,* dan lainnya*.* Untuk parameter dengan tipe *categorical,* sistem hanya akan melakukan pemrosesan *Repeated Event* dan penerapan *Motif Discovery* mengunakan algoritma *Knuth Morris Pratt* (KMP) pada parameter tersebut.
2. *Rule,* konfigurasi ini menentukan bagaimana nantinya suatu parameter akan diinterpretasikan. Untuk parameter dengan *rule fuzzy*, parameter tersebut akan diinterpretasikan menggunakan *Fuzzy membership function*. Sama halnya untuk parameter dengan *rule crisp,* maka parameter tersebutakan diinterpretasikan menggunakan *Crisp membership function.*
3. *Alternate*, konfigurasi ini dapat digunakan untuk mengubah nama parameter pada teks keluaran nantinya.

Tabel 4.5 Konfigurasi parameter pada file *mainconfig.csv*

|  |  |  |  |
| --- | --- | --- | --- |
| **ColName** | **Type** | **Rule** | **Alternate** |
| AirQuality | numeric | crisp | Air Quality |
| WindSpeed | numeric | crisp | Wind Speed |
| WindDirection | numeric | crisp | Wind Direction |
| CloudCoverage | numeric | crisp | Cloud Coverage |
| Temperature | numeric | fuzzy | NA |
| Rainfall | categorical | fuzzy | NA |
| USD | NA | NA | U.S. Dollar |
| JPY | NA | NA | Japan Yen |
| GBP | NA | NA | Great British Pounds |
| CHF | NA | NA | Confoederatio Helvetica Franc |
| SGD | NA | NA | Singapore Dollar |
| MYR | NA | NA | Malaysian Ringgit |
| HKD | NA | NA | Hong Kong dollar |
| AUD | NA | NA | Australian Dollar |
| CAD | NA | NA | Canadian Dollar |

Konfigurasi ini bersifat opsional, pengguna dapat mengubah atau menambahkan konfigurasi untuk suatu parameter pada file *mainconfig.csv,* lebih lengkapnya dapat dilihat pada tabel 4.4*.* Namun jika tidak ditemukan konfigurasi untuk suatu parameter pada dataset, maka sistem akan secara otomatis melakukan pemrosesan data secara *general* sesuai tipe datanya, lalu diinterpretasikan menggunakan *General Fuzzy membership function.* Untuk menambahkan atau mengubah konfigurasi parameter, dapat dilakukan dengan langkah-langkah berikut ini:

1. Buka file *mainconfig.csv* pada folder Config.
2. Pengguna dapat mengubah atau menambahkan konfigurasi dengan memasukan atau mengubah nama parameter, tipe data, *rule,* dan *alternate* pada file tersebut. Jika pengguna hanya mengisi beberapa konfigurasi saja, maka pengguna harus menuliskan NA untuk konfigurasi yang kosong atau tidak diisi.
3. Untuk konfigurasi *rule,* jika pengguna memasukan konfigurasi tersebut, maka pengguna harus menambahkan file pada folder Corpus dengan nama file sesuai dengan nama parameter yang diikuti dengan *Adjective.csv.* Seperti, *AirQualityAdjective.csv*, *TemperatureAdjective.csv,* dan *RainfallAdjective.csv.*

### Model Proses *SignalAnalysis*

Proses *Signal Analysis* merupakan tahapan awal dari sistem D2T, tahapan ini bertujuan untuk mendeteksi dan menganalisis pola-pola diskret yang terdapat pada dataset. Selain itu, pada proses *Signal Analysis* ini dihasilkan juga ringkasan data seperti pada sistem *BABYTALK Family Sistem* yang dipaparkan oleh (Portet, dkk., 2009), sistem tersebut melakukan peringkasan data untuk mencari ringkasan kejadian selama 45 menit, dan juga pada sistem DWP yang menghasilkan ringkasan data klimatologi selama satu bulan dari data klimatologi selama satu tahun (Putra et al., 2017). Selain peringkasan data, pada tahap ini sistem juga melakukan analisis untuk mencari pola kejadian seperti *Motif Discovery, Repeated Event, Extreme Event,* analisis untuk data ke-*n* (data terakhir), dan prediksi data untuk data ke-*n+1*. Setelah melakukan serangkaian analisis pada tahap ini, hasil dari analasis tersebut akan disimpan dan digabungkan dalam bentuk *Data Abstraction and Event*. Untuk lebih jelasnya, input, proses, dan ouput pada tahap ini adalah sebagai berikut:

* Input : *Numerical Dataset with header*
* Proses : *Signal Analysis*
* Output : *Data Abstraction and Event*

Sub-proses pada tahap ini akan dibahas pada sub-bab berikutnya.

* 1. **Proses *Signal Analysis* untuk Ringkasan Data**

Mengacu pada penelitian sebelumnya, untuk proses *Signal Analysis* pada sistem DWP dilakukan serangkaian analisis dan peringkasan data sehingga didapatkan tiga point utama, yakni untuk setiap parameternya dilakukan analisis menggunakan *Statistical Tools* untuk mendapatkan sinyal bulanan dari masing-masing parameter (*Monthly\_Message*), khusus untuk parameter *Rainfall* dilakukan analisis pencarian perulangan suatu pola *(Repeated Event),* lalu analisis *Extreme Event* untuk parameter *Rainfall, WindSpeed,* dan  *Temperature* (Putra et al., 2017). Penelitian ini juga menggunakan beberapa analisis yang dilakukan pada sistem DWP seperti yang sudah dijelaskan pada paragraf sebelumnya, namun untuk proses *Repeated Event* dan *Extreme Event* dilakukan pada proses yang berbeda. Sehingga pada tahap peringkasan data ini, hanya dilakukan analisis sinyal menggunakan *Statistical Tools* dan ditambahkannya analisis sinyal untuk menentukan *trend* pada setiap parameter. Ringkasan ini akan digunakan sebagai dasar pemrosesan data untuk proses analisis yang lainnya. Semua analisis yang ada dalam proses peringkasan data ini hanya diterapkan pada parameter dengan tipe *numerical.*

Data statistik yang dicari pada proses ini berupa nilai minimum (nilai, tanggal, indeks data), nilai maksimum (nilai, tanggal, indeks data), jumlah nilai parameter, nilai rata-rata, dan representasi *trend* untuk setiap parameter dengan tipe *numerical* yang ada, sehingga jika ada *n* buah parameter bertipe *numerical*, maka sinyal yang dihasilkan sebanyak *n\*10* buah, dengan 1 kolom tambahan yang memuat nama parameter. Sebagai contoh pada gambar 4.3 digambarkan proses peringkasan data dengan 9 buah parameter bertipe *numerical,* setelah dilakukan proses peringkasan data didapatkan sinyal-sinyal yang kemudian disimpan dalam sebuah variabel bertipe *data frame* dengan 9buah baris sesuai jumlah parameter dengan tipe *numerical*, dan10buah kolom.

Dalam menentukan *trend* sebuah parameter, digunakan *Linear Model* dan simbol untuk merepresentasikan tingkat kenaikan atau penurunan suatu parameter sehingga sistem dapat menentukan apakah *trend*  dari parameter tersebut menaik (+) , menurun (-) , atau justru konstan (0) . Namun perlu diketahui, dalam penentuan *trend* ini diterapkan *Minimum Treshold* sebesar 5%, dimana sebuah parameter akan tetap dikatakan konstan jika selisih titik terakhir dan titik pertama model linear tidak melebihi 5% dari rentang keseluruhan data.

Berikut beberapa contoh kasus dalam penentuan *trend*:

|  |
| --- |
| Hasil ringkasan dari parameter *Rainfall* pada dataset tabel 4.2 didapatkan nilai minimum 0 dan nilai maksimum 67.6 sehingga rentang keseluruhan data yang didapatkan sebesar 67.6 . Lalu data diinterpretasikan dengan model linear, maka didapatkan nilai titik pertama 1.76 dan nilai titik terakhir 3.37, sehingga rentang dari model linear tersebut sebesar 1.61 (3.37 - 1.76 = 1.61). Karena selisih titik model linear tidak lebih besar dari 5% rentang data keseluruhan (5% dari 67.6 = 3.38) maka *trend* dari parameter *Rainfall* pada dataset tabel 4.2 adalah “0” atau konstan.  Hasil ringkasan dari parameter *CloudCoverage* pada dataset tabel 4.2 didapatkan nilai minimum 7.9 dan nilai maksimum 100 sehingga rentang keseluruhan data yang didapatkan sebesar 92.1 . Lalu data diinterpretasikan dengan model linear, maka didapatkan nilai titik pertama 43.2 dan nilai titik terakhir 49.0, sehingga rentang dari model linear tersebut sebesar 5.80 (49.0 - 43.2 = 5.80). Karena rentang model linear lebih besar dari 5% rentang data keseluruhan (5% dari 92.1 = 4.60) , dan selisih model linear bernilai positif maka *trend* dari parameter *CloudCoverage* pada dataset tabel 4.2 adalah “+” atau menaik.  Hasil ringkasan dari parameter *Temperature* pada dataset tabel 4.2 didapatkan nilai minimum 1 dan nilai maksimum 25.30 sehingga rentang keseluruhan data yang didapatkan sebesar 24.30 . Lalu data diinterpretasikan dengan model linear, maka didapatkan nilai titik pertama 15.50 dan nilai titik terakhir 12.40, sehingga rentang dari model linear tersebut sebesar -3.10 (12.40 – 15.50 = -3.10). Karena nilai mutlak rentang model linear lebih besar dari 5% rentang data keseluruhan (5% dari 92.1 = 1.21), dan selisih model linear bernilai negatif maka *trend* dari parameter *Temperature* pada dataset tabel 4.2 adalah “-” atau menurun. |

* 1. **Proses *Signal Analysis* untuk *Extreme Event***

Proses ini bertujuan untuk mencari pola-pola perubahan yang ekstrem baik berupa kenainkan ataupun penurunan pada suatu parameter dengan tipe *numerical*. Contohnya ketika suhu hari ini sangat panas, tiba-tiba sehari kemudian berubah menjadi sangat dingin, inilah perubahan ekstrem yang dimaksudkan. Untuk melakukan proses pencarian nilai ekstrem, pertama-tama setiap kenaikan dan setiap penurunan suatu parameter dijumlahkan terlebih dauhulu. Setelah didapatkan jumlah kenaikan atau penurunannya, langkah selanjutnya adalah menyimpan nilai kenaikan dan penurunan tertinggi bersama indeks data dimana kenaikan atau penurunan tersebut berawal dan berakhir (dalam bentuk numerik).

Dengan menggunakan data pada tabel 4.2 setelah dilakukan proses deteksi *Extreme Event* sehingga didapatkan hasil pada tabel 4.5, seperti yang kita lihat pada tabel tersebut, data yang disimpan berupa nilai kenaikan dan penurunan tertinggi bersama indeks-indeksnya, dimana data-data tersebut akan digunakan sebagai acuan dalam menentukan apakah kenaikan atau penurunan tersebut termasuk kondisi perubahan ekstrem atau tidak. Sebuah data akan dikatakan termasuk pada *extreme event* jika kenaikannya melebihi 65% dari interval data.

Tabel 4.6 Hasil *Extreme Event* untuk contoh kasus data klimatologi

| ColName | Inc-Value | incStart-Index | incEnd-Index | dec-Value | dec Start-Index | dec End-Index |
| --- | --- | --- | --- | --- | --- | --- |
| CloudCoverage | 90.9 | 47 | 50 | -87.6 | 245 | 247 |
| Temperature | 10.3 | 263 | 266 | -9.8 | 183 | 185 |
| WindSpeed | 20.05 | 210 | 212 | -27.03 | 213 | 217 |
| WindDirection | 270 | 23 | 23 | -270 | 7 | 8 |
| Rainfall | 67.2 | 211 | 212 | -64.1 | 213 | 214 |

* 1. **Proses *Signal Analysis* untuk *Repeated Event***

Proses ini tidak jauh beda dengan proses pendeteksian *Trend* dimana diterapkannya *Minimum Treshold* atau batas minimum, dimana jika ditemukan sebuah parameter dengan nilai sama berturut-turut lebih dari 10% jumlah data maka termasuk ke dalam kategori *Repeated Event*. Selain untuk parameter dengan tipe *numerical,* proses ini dilakukan untuk juga pada parameter dengan tipe *Categorical*. Berikut beberapa contoh kasus pada penentuan *Repeated Event*:

|  |
| --- |
| * + 1. Pada parameter X dalam data Y dengan jumlah 30 baris data (bulanan), terdapat nilai yang sama secara berturut-turut pada baris 5 sampai baris 10 (5 baris), maka sinyal tersebut merupakan sinyal *repeated event* karena 5 baris lebih besar dari 3 (10% \* 30 baris).     2. Pada parameter X dalam data Z dengan jumlah 366 baris data (tahunan), terdapat 20 baris data dengan nilai yang sama secara berturut-turut, karena 20 baris tidak lebih besar dari (10% \* 366 baris) maka sinyal tersebut bukan *repeated event*. |

* 1. **Proses *Signal Analysis* untuk Prediksi Data**

Salah satu bagian dari proses *Signal Analysis* yaitu dilakukannya proses prediksi data. Proses ini hanya diterapkan hanya untuk parameter dengan tipe data *numerical*. Proses ini bertujuan agar informasi akan disampaikan kepada pembaca lebih informatif. Selain itu juga, informasi yang disampaikan berisi kemungkinan-kemungkinan yang akan terjadi kedepannya. Dalam melakukan prediksi, diterapkan konsep prediksi *time-series* dengan algoritma *Exponential Smoothing* seperti peneliatan DWP sebelumnya (Putra et al., 2017).

Dengan menggunakan data pada tabel 4.2 setelah dilakukan proses prediksi data sehingga didapatkan hasil pada tabel 4.6 , seperti yang kita lihat pada tabel tersebut, data yang disimpan berupa hasil prediksi dengan menggunakan algoritma *Exponential Smoothing*. Data yang disimpan pada proses ini masih berbentuk numerik, yang nantinya akan diproses dan dijelaskan lebih mendetail pada bagian *Data Interpretation.*

Tabel 4.7 Hasil Prediksi data untuk contoh kasus data klimatologi

| CloudCoverage | Temperature | WindSpeed | WindDirection | Rainfall |
| --- | --- | --- | --- | --- |
| 46.9075 | 17.9001 | 7.2226 | 276.331 | 1.0607 |

* 1. **Proses *Signal Analysis* untuk *String Matching***

Proses pendeteksian *string matching* ini bertujuan untuk mencari pola yang sama pada sebuah parameter dengan tipe *categorical*, pendeteksian pola yang sama ini dilakukan dengan mengambil beberapa baris data terakhir sebagai acuan lalu mencocokannya dengan seluruh data pada dataset. Banyaknya data yang diambil, bergantung pada interval data yang menjadi masukan. Pengguna dapat menentukan sendiri berapa banyak data yang diambil untuk acuan nantinya. Namun untuk nilai *defaultnya,* jika data dengan interval per jam, maka diambil data atau pola acuan yang diambil sebanyak 6 baris atau data 6 jam terakhir. Untuk data dengan interval per hari maka data yang diambil sebanyak 7 baris atau data seminggu terakhir. Selebihnya, untuk data dengan interval bulanan atau tahunan, maka data yang diambil sebanyak 4 baris data terakhir.

Dengan menggunakan data pada tabel 4.2 setelah dilakukan proses pendeteksian *string matching* sehingga didapatkan hasil pada tabel 4.6 , seperti yang kita lihat pada tabel tersebut, data yang disimpan berupa hasil prediksi dengan menggunakan algoritma *Exponential Smoothing*. Data yang disimpan pada proses ini masih berbentuk numerik, yang nantinya akan diproses dan dijelaskan lebih mendetail pada bagian *Data Interpretation.*

* 1. **Proses *Signal Analysis* untuk Menentukan Korelasi Parameter**

Dalam menentukan korelasi antar parameter, digunakan perhitungan korelasi Pearson, dimana data akan dipetakan kedalam matriks berukuran *n,n* dimana *n* merupakan banyaknya parameter *numerical* dalam suatu data. Matriks tersebut berisikan nilai dengan rentang 0 sampai 1 yang merepresentasikan hubungan linear antara satu parameter dengan parameter lainnya. Semakin tinggi nilai yang didapat, menandakan hubungan linear parameter tersebut sangat kuat. Jika didapatkan hasil positif, maka angka tersebut merepresentasikan hubungan yang sebanding, kenaikan suatu parameter berdampak pada kenaikan parameter lannya, begitu juga sebaliknya. Namun jika hasil yang didapatkan bernilai postif maka, nilai tersebut merepresentasikan hubungan yang bertolak belakang antara parameter tersebut, kenaikan pada suatu parameter akan berdampak penurunan pada parameter lainnya, begitu juga penurunan suatu parameter akan berdampak pada kenaikan parameter lainnya.

Dengan mengunakan data partikel udara pada tabel 4.4 lalu dilakukan pendeteksian sinyal untuk mencari korelasi antar parameter maka didapatkan hasil pada tabel 4.8. Dapat kita lihat, tabel tersebut menampilkan nilai korelasi antar parameter. Seperti contohnya parameter TEMP memiliki nilai korelasi -0.8095 terhadap parameter PRES, yang berarti kedua parameter tersebut memiliki keterkaitan yang cukup kuat. Dimana ketika kenaikan terjadi pada parameter TEMP atau *Temperature* maka akan terjadi penurunan pada parameter PRESatau *Pressure*, hal ini sejalan dengan hukum alam, dimana ketika suhu semakin panas maka udara akan memuai lalu naik karena lebih ringan, akibat udara yang naik tersebut maka tekanan akan turun.

Tabel 4.8 Hasil pendeteksian sinyal menggunakan *Pearson Correlation*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | PM2.5 | DEWP | TEMP | PRES | LWS | IS | IR |
| PM2.5 | 1.0000 | 0.2776 | 0.0377 | -0.1905 | -0.2632 | -0.0130 | -0.0506 |
| DEWP | 0.2776 | 1.0000 | 0.8431 | -0.7687 | -0.3112 | -0.0410 | 0.1220 |
| TEMP | 0.0377 | 0.8431 | 1.0000 | -0.8095 | -0.1780 | -0.0997 | 0.0500 |
| PRES | -0.1905 | -0.7687 | -0.8095 | 1.0000 | 0.2120 | 0.0629 | -0.0592 |
| IWS | -0.2632 | -0.3112 | -0.1780 | 0.2120 | 1.0000 | 0.0237 | -0.0372 |
| IS | -0.0130 | -0.0410 | -0.0997 | 0.0629 | 0.0237 | 1.0000 | -0.0098 |
| IR | -0.0506 | 0.1220 | 0.0500 | -0.0592 | -0.0372 | -0.0098 | 1.0000 |

Setelah didapatkan matriks berisikan nilai korelasi tersebut, yang akan ditampilkan sebagai pesan adalah nilai korelasi parameter yang melebihi 0.7 (baik positif, maupun negatif). Jika kita kita menggunakan hasil perhitungan pada tabel 4.8, semua nilai yang lebih kecil dari 0.7 atau lebih besar dari -0.7 akan dihilangkan, sehingga didapatkan hasil seperti pada tabel 4.9. Nilai ini lah yang nantinya akan diproses sebagai keluaran nantinya.

Tabel 4.9 Nilai korelasi parameter yang akan diproses

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | PM2.5 | DEWP | TEMP | PRES | LWS | IS | IR |
| PM2.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| DEWP | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| TEMP | 0 | 0.8431 | 0 | 0 | 0 | 0 | 0 |
| PRES | 0 | -0.7687 | -0.8095 | 0 | 0 | 0 | 0 |
| IWS | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| IS | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| IR | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### Model Proses *Data Interpretation*

Pada proses ini dilakukan penerjemahan sinyal-sinyal dan *event* yang dihasilkan pada bagian sebelumnya menjadi bentuk frasa atau kata-kata yang akan disampaikan (Reiter, 2011). Proses penerjemahan ini dilakukan dengan menggunakan logika *Fuzzy* dan *Crisp* seperti yang dilakukan pada penelitian DWP sebelumnya (Putra et al., 2017). Penentuan penggunaan logika *Fuzzy* dan *Crisp* ini ditentukan sesuai dengan tipe penginterpretasian yang tersimpan pada konfigurasi parameter seperti yang sudah dijelaskan pada tahap *General Data Handler.* Dimana sinyal-sinyal baik berupa pola maupun event yang dihasilkan pada *Signal* *Analysis* akan direpresentasikan ke dalam bentuk pesan-pesan yang dapat dipahami manusia. Contohnya, seperti yang dilakukan oleh Sripada & Gao (2007) dalam kasus mengenai detak jantung, setelah mendepatkan sinyal berupa detak jantung (dalam bentuk numerik), jika data detak jantung yang didapatkan dari proses analisis bernilai dibawah 100, maka data tersebut akan didefinisikan menjadi sebuah kalimat “*heart rate is temorarily low”*. Proses penginterpretasian data ini dilakukan dengan mengacu pada nilai-nilai yang sudah ditentukan oleh *expert* sebelumnya, *Fuzzy Membership Function* digunakanuntuk kasus data dengan tipe penginterpretasian *Fuzzy*, dan *Crisp Membership Function* untuk kasus data dengan tipe penginterpretasian *Crisp*.

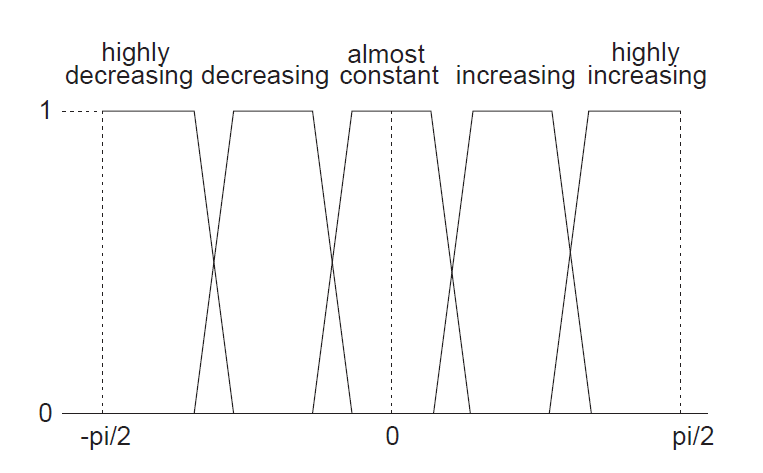
Pada penelitian ini, jika sebuah parameter tidak dikenali oleh sistem atau tidak terdapatnya cara penginterpretasian data pada konfigurasi parameter dalam file *mainconfig.csv*, maka sistem akan secara otomatis menggunakan *General Membership Function* untuk penginterpretasian datanya. Karena tidak semua pesan akan disampaikan, maka selain menerjemahkan data kedalam bentuk teks, pada tahap ini dilakukan penentuan seberapa pentingnya pesan tersebut disampaikan, hal ini serupa dengan penelitian yang dilakukan oleh (Portet et al., 2007). Penentuan ini bertujuan, agar pesan yang disampaikan tidak monoton. Contohnya dalam penginterpretasian sinyal *Repeated Event,* jika ada parameter yang memiliki nilai berulang maka sistem hanya akan menampilkan pesan untuk parameter tersebut saja, dan secara otomatis akan mengabaikan parameter lain yang tidak mempunyai nilai berulang di dalamnya. Begitu juga jika tidak terdapat parameter yang mempunyai nilai berulang di dalamnya, maka sistem hanya akan menampilkan pesan bahwa tidak ditemukannya nilai yang berulang sekali saja, dibanding dengan menampilkan pesan untuk semua parameternya. Sehingga *input*, proses, dan *output* pada tahap ini adalah sebagai berikut:

* *Input: Data Abstraction*, *and Event*
* Proses: *Data Interpretation*
* *Output: Message Inventory*

Beberapa acuan *corpus* interpretasi data pada penelitian ini adalah sebagai berikut:

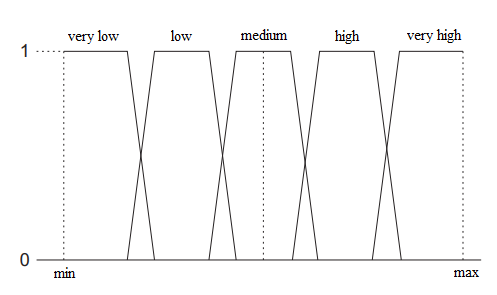
1. **Interpretasi Data Umum**

Yang membedakan penelitian ini, adalah adanya interpretasi data untuk kasus data umum atau data *general,* dimana sistem akan tetap menginterpretasikan sebuah parameter walau parameter tersebut tidak memiliki *corpus* untuk menginterpretasikannya. Sehingga ketika data jenis apapun dimasukkan, sistem akan tetap berjalan seperti meskipun yang menjadi masukan merupakan data *random* ataupun data yang tidak mempunyai *header*. Proses untuk menginterpretasikan data *general* ini menggunakan logika *fuzzy* dalam penginterpretasian datanya. Dimana logika *fuzzy* yang digunakan ini mengacu pada penelitian penentuan trend oleh Castillo-Ortega et al., (2014). Pada penelitian tersebut, Castillo membagi dan menentukan trend menjadi lima wilayah yang sama besar berdasarkan nilai pi (nilai rata-rata data) kemudian memasukan data yang didapatkan pada proses analisis sinyal kedalam *Fuzzy Membership function*, seperti pada gambar 4.3.



Gambar 4.3 *Linguistic variable for trend description* (Castillo-Ortega et al., 2014)

Jika pada penelitian DWP, Untuk kasus interpretasi data *general,* penulis memodifikasi cara pentinterpretasiannya dengan menggunakan batas *min-max* dari suatu parameter maka *Fuzzy membership function* untuk data *general* menjadi seperti pada gambar 4.4, dengan *corpus* *very low, low, medium, high, very high* (Fallah-Ghalhary et al., 2009).



Gambar 4.4 *General Fuzzy Membership Function*

Dengan menggunakan *Data Abstraction and Event* yang didapatkan dari data pada tabel 4.1 maka didapatkan hasil interpretasi data seperti pada tabel 4.7. Proses penginterpretasian dilakukan menggunakan *General Fuzzy Membership Function,* dikarenakan pada data terebut parameter-parameternya tidak memiliki konfigurasi *Rule* atau cara penginterpretasian data, kita dapat melihat konfigurasi parameter pada tabel 4.4 dimana pada konfigurasi parameter tersebut, hanya terdapat konfigurasi *Alternate* sedangkankonfigurasi *Rule* untuk parameter-parameter data nilai tukar tersebut bernilai *NA* sehingga sistem mengenali parameter-paramateter tersebut sebagai data umum atau data *general.*

Tabel 4.10 Hasil *Data Interpretation* contoh kasus data nilai tukar

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | USD | JPY | GBP | CHF | ... | CAD |
| Resume Data | low | medium | medium | medium | ... | medium |
| Current Data | very high | very high | high | very high | ... | very high |
| Predict Data | very high | very high | high | very high | ... | very high |

1. **Interpretasi Korelasi Parameter**

Untuk menginterpretasikan nilai dari *Pearson Correlation Coefficient* ini digunakan ketentuan yang ditetapkan oleh De Vaus (2002) dimana nilai yang dihasilkan saat proses pendeteksian sinyal untuk korelasi parameter dibagi menjadi 7 kategori seperti pada tabel 4.11 berikut.

Tabel 4. 11 Parameter Correlation Crisp Membership Function (de Vaus, 2002)

| No | Coefficient | Class |
| --- | --- | --- |
| 1 | 0.00 | no linear association |
| 2 | 0.01 - 0.09 | insubtantial relationship |
| 3 | 0.10 - 0.29 | low relationship |
| 4 | 0.30 - 0.49 | moderate relationship |
| 5 | 0.50 - 0.69 | strong relationship |
| 6 | 0.70 - 0.89 | very strong relationship |
| 7 | 0.9+ | almost perfect relationship |

1. **Interpretasi Kecepatan Angin**

Dalam menginterpretasikan nilai dari kecepatan angin, digunakan ketentuan yang ditetapkan oleh Rowlett (2001) yang ngelompokkan kecepatan angin menjadi 13 kategori seperti pada tabel 4.8 berikut.

Tabel 4.12 Wind Speed Crisp Membership Function (Rowlett, 2001)

| No | Speed (km/h) | Class |
| --- | --- | --- |
| 1 | x < 2 | Calm |
| 2 | 2 <= x < 5 | Light Air |
| 3 | 6 <= x < 11 | Light Breeze |
| 4 | 12 <= x < 19 | Gentle Breeze |
| 5 | 20 <= x < 29 | Moderate Breeze |
| 6 | 30 <= x < 39 | Fresh Breeze |
| 7 | 40 <= x < 50 | Strong Breeze |
| 8 | 51 <= x < 61 | Near Gale |
| 9 | 62 <= x < 74 | Gale |
| 10 | 74 <= x < 87 | Strong Gale |
| 11 | 88 <= x < 102 | Storm |
| 12 | 103 <= x < 118 | Violent Storm |
| 13 | 119 <= x < 130 | Hurricane |

1. **Interpretasi Arah Angin**

Untuk interpretasi arah angin penulis menggunakan *crisp membership function* yang dibuat oleh Zandlo, Bouley dan Ruschy dengan 16 kategori seperti pada tabel 4.9 berikut (Zandlo et al., 2001).

Tabel 4.13 *Wind Direction Crisp Membership Function* (Zandlo et al., 2001)

| No | Degree | Class |
| --- | --- | --- |
| 1 | 348.75-11.25 | North |
| 2 | 11.25-33.75 | North North East |
| 3 | 33.75-56.25 | North East |
| 4 | 56.25-78.75 | East North East |
| 5 | 78.75-101.25 | East |
| 6 | 101.25-123..75 | East South East |
| 7 | 123.75-146.25 | South East |
| 8 | 146.25-168.75 | South South East |
| 9 | 168.75-191.25 | South |
| 10 | 191.25-213.75 | South South West |
| 11 | 213.75-236.25 | South West |
| 12 | 236.25-258.75 | West South West |
| 13 | 258.75-281.25 | West |
| 14 | 281.25-303.75 | West North West |
| 15 | 303.75-326.25 | North West |
| 16 | 326.25-348.75 | North North West |

1. **Interpretasi Cakupan Awan**

Dalam proses interpretasi cakupan awan, peneliti menggunakan kategori cakupan awan yang didefiniskan oleh seorang ahli meteorologi bernama Jeff Huby yang mengelompokkan cakupan awan menjadi 7 kategori, untuk lebih lengkapnya lihat pada tabel 4.10 (Hubby, 2010).

Tabel 4.14 *Cloud Coverage Crisp Membership Function* (Hubby, 2010)

|  |  |  |
| --- | --- | --- |
| NO | Cloud Coverage (%) | Class |
| 1 | 0-10 | Clear |
| 2 | 10-20 | Foggy |
| 3 | 20-30 | Mostly sunny |
| 4 | 40-60 | Partly cloudy |
| 5 | 70-80 | Mostly cloudy |
| 6 | 80-90 | Broken |
| 7 | 90-100 | overcast |

1. **Interpretasi Kualitas Udara**

Dalam menginterpretasikan data kualitas udara, dibutuhkan nilai *Polutant Standard Index* (PSI) seperti pada penelitian DWP (Putra et al., 2017). Dimana nilai PSI ditentukan dengan membagi konsentrasi masing-masing lima polutan udara utama sehingga mendapatkan angka antara 0 dan 500, kelima pulutan tersebut adalah *Particle Matter* (PM10), partikel halus (PM2.5), *sulfur dioksida* (SO2), karbon monoksida (CO), ozon (O3) dan nitrogen dioksida (NO2) untuk jangka waktu tertentu. Setelah nilai PSI didapatkan, penulis merujuk pada kategori yang ditentukan oleh Crowder, Moore, DeRose, dan Franek (Crowder et al., 1999). Polutan yang ditetapkan oleh *National Ambient Air Quality Standard* (NAAQS) ini dapat dilihat pada tabel 4.11.

Tabel 4.15 *Air Quality* *Crisp Membership Function* (Crowder et al., 1999)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| PSI *Index* | PSI | PM25 | PM10 | SO2 | CO | O3 | NO2 |
| Good | 0-50 | 0-12 | 0-50 | 0-80 | 0-5 | 0-118 | 0 |
| Moderate | 51-100 | 13-55 | 51-15 | 81-365 | 5.1-10 | 119-157 | 0 |
| Unhealthy | 101-200 | 56-150 | 151-350 | 366-800 | 10.1-17.0 | 158-235 | 1130 |
| Very Unhealthy | 201-300 | 151-250 | 351-420 | 801-1600 | 17.1-34.0 | 236-785 | 1131-2260 |
| Hazzardous | 301-500 | 251-500 | 421-600 | 1601-2620 | 34.1-57.5 | 786-1180 | 2261-3750 |

Dalam penelitian DWP, Putra et al., (2017) menuturkan bahwa untuk menghitung PSI, digunakan fungsi linier tersegmentasi yang mengubah konsentrasi ambien ke skala yang berkisar dari 0 sampai 500. Setiap sub-indeks *i*, dihitung dengan menggunakan fungsi linier tersegmentasi yang berhubungan dengan konsentrasi polutan, *Xi* terhadap nilai sub-indeks, *Ii*. Fungsi linier tersegmentasi terdiri dari segmen garis lurus yang menggabungkan koordinat diskrit (yaitu *breakpoint* yang terdapat pada tabel 4.12). Untuk polutan *i* dan segmen *j*, koordinat titik *jth* break diwakili oleh nilai sub-indeks *Ii, j* dan konsentrasi *Xi, j* memberikan pasangan yang dipesan (*Xi, j, Ii, j*). Jika konsentrasi yang diamati adalah *Xi* nilai sub-indeks yang sesuai *Ii* perhitungan dilakukan dengan menggunakan persamaan berikut (keterangan rumus terdapat pada tabel 4.12):

(5)

(need to be updated: no persamaan)

Selain itu Putra et al., (2017) mengkonversi persamaan diatas kedalam bentuk kode program, dengan langkah-langkah sebagai berikut:

1. Definisikan tiap *breakpoint* bagi setiap variabel polutan (PM25, PM10, SO2, CO, O3, NO2. Nilai *breakpoint* dari setiap variabel ini akan digunakan saat mengimplementasikan persamaan.
2. Definisikan tiap *breakpoint* untuk *Index* PSI.
3. Hitung nilai PSI bagi tiap variabel, dengan mengimplementasikan persamaan diatas.
4. Setelah didapatkan nilai PSI dari tiap variabel, cari nilai PSI terbesar dari.
5. Setelah nilai PSI terbesar didapatkan, cari interval dari PSI *Index* yang sesuai dengan nilai ini dari setiap indeks yang tersedia.
6. Setelah nilai PSI *Index* didapatkan, lalu ubah kedalam kategori sistem.

Tabel 4.16 Keterangan Rumus PSI

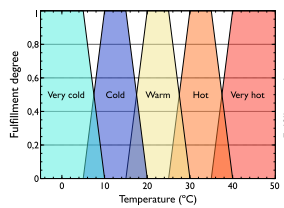
| NO | Variabel | Keterangan |
| --- | --- | --- |
| 1 | Xi | Nilai konsentrasi pollutant (yang diamati) ke-i. |
| 2 | Ii,j | Nilai PSI untuk polutan ke-i dan *breakpoint* ke-j (seperti yang diberikan pada Tabel). |
| 3 | Ii,j+1 | Nilai PSI untuk polutan ke-i dan breakpoint ke-(j + 1) (seperti yang diberikan pada Tabel). |
| 4 | Xi,j | Konsentrasi untuk polutan dan *breakpoint* seperti yang diberikan pada Tabel. |

Berikut contoh perhitungan PSI:

|  |
| --- |
| Misalkan diketahui nilai polutan PM25 adalah 25 , maka:   * + - * 1. Berdasarkan interval *breakpoint*, nilai 25 pada PM25 berada pada rentang 12-55.         2. Maka perhitungan dilakukan pada segmen pertama (j = 1). Nilai dari *X1,1* = 12, *X1,2* = 55, *I1,1* = 50, dan *I1,2* = 100.7777         3. Sehingga kita dapat menghitung dengan menggunakan persamaan (5):         4. Maka nilai dari sub-indeks PSI (*I1*) PM25 adalah , sedangkan kita misalkan nilai sub-indeks PSI(*I2*) PM10 adalah 46.1412344, (*I3*) SO2 adalah 43.135112, (*I4*) CO adalah 45.166162, (*I5*) O3 adalah 15.136222 dan (*I6*) NO2 adalah -\*.         5. Mengambil nilai maksimal dari setiap sub-indeks.         6. Diperoleh nilai 65.11627907 yang kemudian dipetakan kedalam tabel 4.9.         7. Maka didapatkanlah hasil: “Moderate”. |

1. **Interpretasi Suhu**

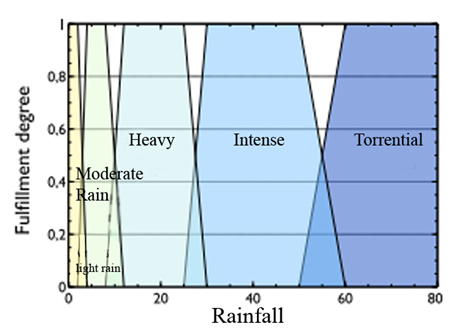
Dalam menginterpretasikan nilai suhu, digunakan *Fuzzy membership function* yang didefinisikan oleh Ramos-soto, Alejandro, Bugarin, dan Barro yang terdiri dari *Very Cold*, *Cold*, *Warm*, *Hot*, dan *Very Hot* (Ramos-Soto et al., 2016a). Untuk fungsi keanggotaannya dapat dilihat pada gambar 4.5.



Gambar 4.5 Temperature Fuzzy membership function (Ramos-Soto et al., 2016a)

1. **Interpretasi Suhu**

Sama halnya dengan penginterpretasian pada suhu, data curah hujan juga menggunakan *Fuzzy membership function* yang didefinisikan oleh Ramos-soto, Alejandro, Bugarin, dan Barro yang terdiri dari *Light Rain, Moderate Rain, Heavy Rain, Intense Rain*,dan *Torential Rain* . Fungsi keanggotaan dapat dilihat pada gambar 4.6.



Gambar 4.6 *Fuzzy membership function Rainfall* (Ramos-Soto et al., 2016a)

### Model Proses *Document Planning*

Pada proses ini dilakukan pemilihan konten (*Content Determination*), dan pembentukan struktur teks (*Document Structure*) yang akan ditampilkan (Reiter, 1996). Mengingat banyaknya pesan atau pola yang dihasilkan dari proses-proses sebelumnya, pemilihan konten akan dipilih berdasarkan *Message Inventory,* dan struktur teks yang akan dibangun berdasarkan *schema-based* yang diperkenalkan oleh Turner (Turner et al., 2008). Sehingga *input,* proses, dan *output* pada tahap ini adalah sebagai berikut:

* *Input*: *Message Inventory*
* Proses: *Document Planning*
* *Output*: *Content and Structure*

Pemilihan konten dan pembentukan struktur untuk setiap paragraf akan dijelaskan pada sub-bab berikutnya.

* + - 1. **Perencanaan Dokumen untuk Ringkasan Data**
      2. *Content Determination*

Proses pemilihan konten untuk bagian ringkasan data ini mengacu pada penelitian DWP (Putra et al., 2017). Pada penelitian tersebut, konten-konten yang akan ditampilkan dibagi menjadi dua bagian, yaitu *Routine Message* dan *Significant Event Message*. *Routine Message* merupakan pesan-pesan yang akan selalu muncul pada teks yang akan ditampilkan, sedangkan *Significant Event Message* adalah pesan yang akan ditampilkan untuk kondisi tertentu saja. Umumnya *Significant Event Message* memerlukan syarat-syarat tertentu agar pesan-pesannya bisa ditampilkan. Contohnya, dalam kasus pembangkitan berita mengenai kasus gempa bumi, pesan statu hanya akan ditampilkan jika guncangan mencapai status siaga. Pada penelitian ini, yang termasuk pada *Significant Event Message* adalah *Repeated Event, Extreme Event,* dan *String Matching* (need to be updated). Untuk gambaran *Routine Message*, terlihat pada gambar 4.7 terlihat bahwa terdapat variabel parameter[i:n], yang berarti dari parameter ke-i (awal) hingga parameter ke-n (terakhir).

*Summary – Routine Message ->*

*Construct Message {*

*Date Time*

*Parameter[i:n]*

*Parameter[i:n] Trend*

*Parameter[i:n] Comparsion*

*}*

Gambar 4.7 *Routine Message* untuk ringkasan Data

*Significant Event Message* pada gambar 4.8, terdapat variabel parameter yang diikuti dengan nama event, seperti parameter[*extreme event*] atau sejenisnya, pesan tersebut menandakan pada parameter mana saja event itu terjadi.

*Summary – Significant Event Message ->*

***IF*** *Repeated Event = “True” |*

***THEN*** *Construct Message {*

*Parameter[Repeated Event]*

*Repeated Event*

*}*

***IF*** *Extreme Event = “True” |*

***THEN*** *Construct Message {*

*Parameter[Extreme Event]*

*Extreme Event*

*}*

***IF*** *Corrleation Event = “True” |*

***THEN*** *Construct Message {*

*Parameter[Corrleation Event]*

*Corrleation Event*

*}*

***IF*** *S.Matching Event = “True” |*

***THEN*** *Construct Message {*

*Parameter[S.Matching Event]*

*S.Matching Event*

*}*

Gambar 4. 8 *Significant Event Message* untuk ringkasan data

Sebagai contoh jika kita menggunakan data pada klimatologi pada tabel 4.2 maka akan didapatkan hasil seperti pada tabel 4.13 (need to be updated)

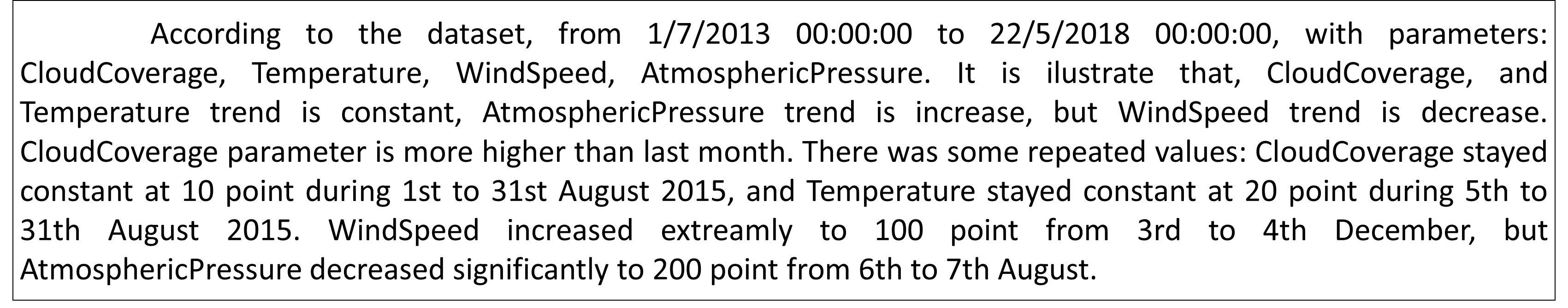
Tabel 4.17 Hasil *Content Determination Summary* untuk data klimatologi

|  |  |
| --- | --- |
| Routine Message | It can be seen that Cloud Coverage, Temperature trend is decreased but the rest is increased. |
| Significance Event  Message | Cloud Coverage decreased extremely from 14 Jan 2014 to 15 Jan 2014 (decreased 56.7 points), Temperature decreased significantly from 11 Jan 2014 to 12 Jan 2014 (decreased 2.3 points), CO fluctuate extremely (decreased 3.8 points and increased 4.1 points),There was repeated value in SO2 from 07 Jan 2014 to 13 Jan 2014 with value 4.25. |

* + - 1. *Content Struturing*

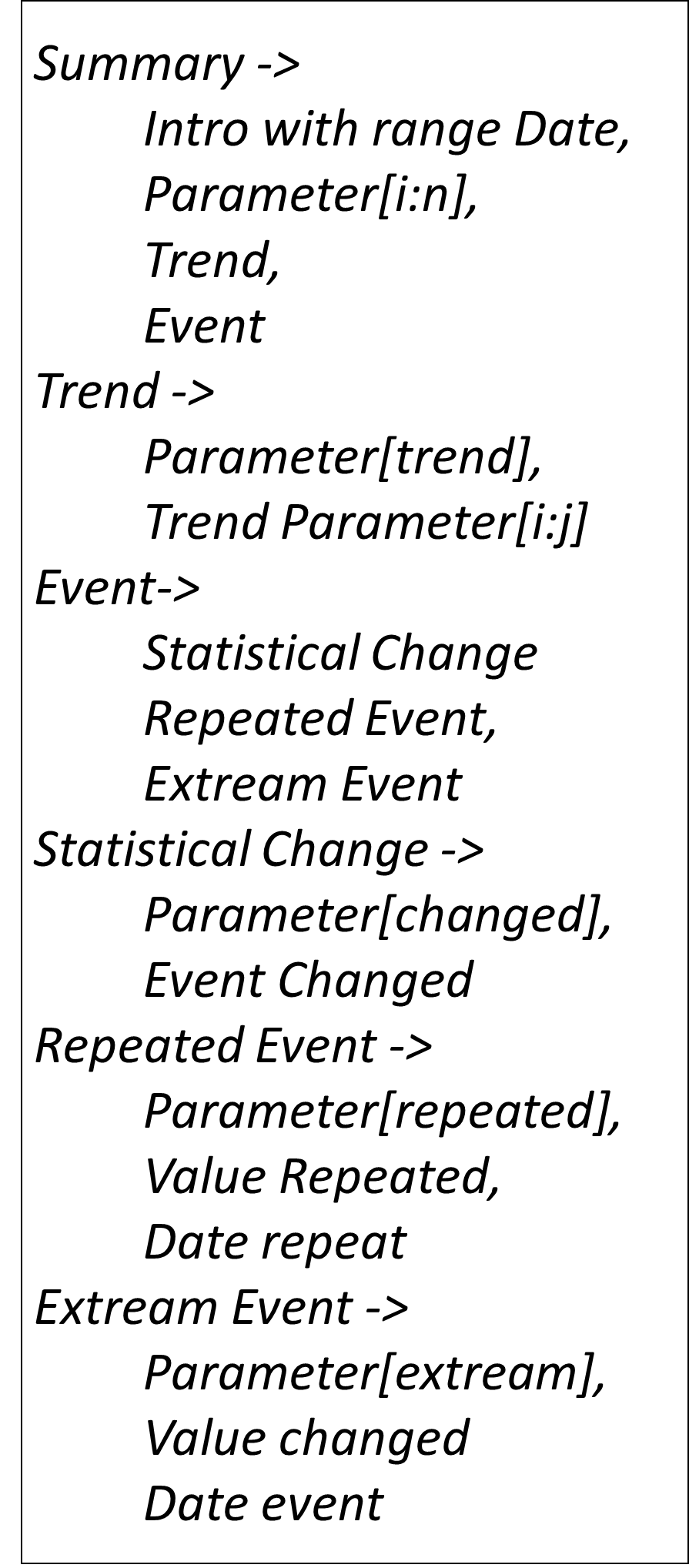
Dalam proses penentuan struktur dokumen, Reiter (1996) memaparkan bahwa penentuan struktur dokumen dapat dilakukan dengan cara membangun skema dokumen atau penalaran secara eksplisit. Teks yang dihasilkan diharapkan sesuai dengan pola umum atau konvensional, maka dari itu dalam menentukan struktur dokumen yang akan dibuat, diperlukan sebuah *Initial Corpus* sebagai target dari keluaran teks nantinya. Pembentukan *Initial Corpus* ini bertujuan untuk mendikte konten dan memastikan struktur yang koheren, sehingga teks keluaran yang akan dihasilkan lebih padat dan tidak melebar pembahasannya.

Dalam menentukan *Initial Corpus* atau *Target Text* ini, setidaknya ada 4 langkah yang harus dilakukan (Reiter, 1996), diantaranya :

* + - 1. Ambil contoh sejumlah teks dengan bidang yang sama untuk dijadikan sebagai *Target Text*.
      2. Melakukan dentifikasi terhadap pesan-pesan yang ada, lalu menentukan bagaimana setiap pesan dapat dibangun berdasarkan data.
      3. Mengusulkan aturan atau struktur yang menjelaskan mengapa pesan “x” ada dalam teks A tetapi tidak ada didalam teks B. Penentuan ini lebih mudah jika disusun dalam bentuk seperti taksonomi.
      4. Diskusikan hasil analisis bersama pakar.

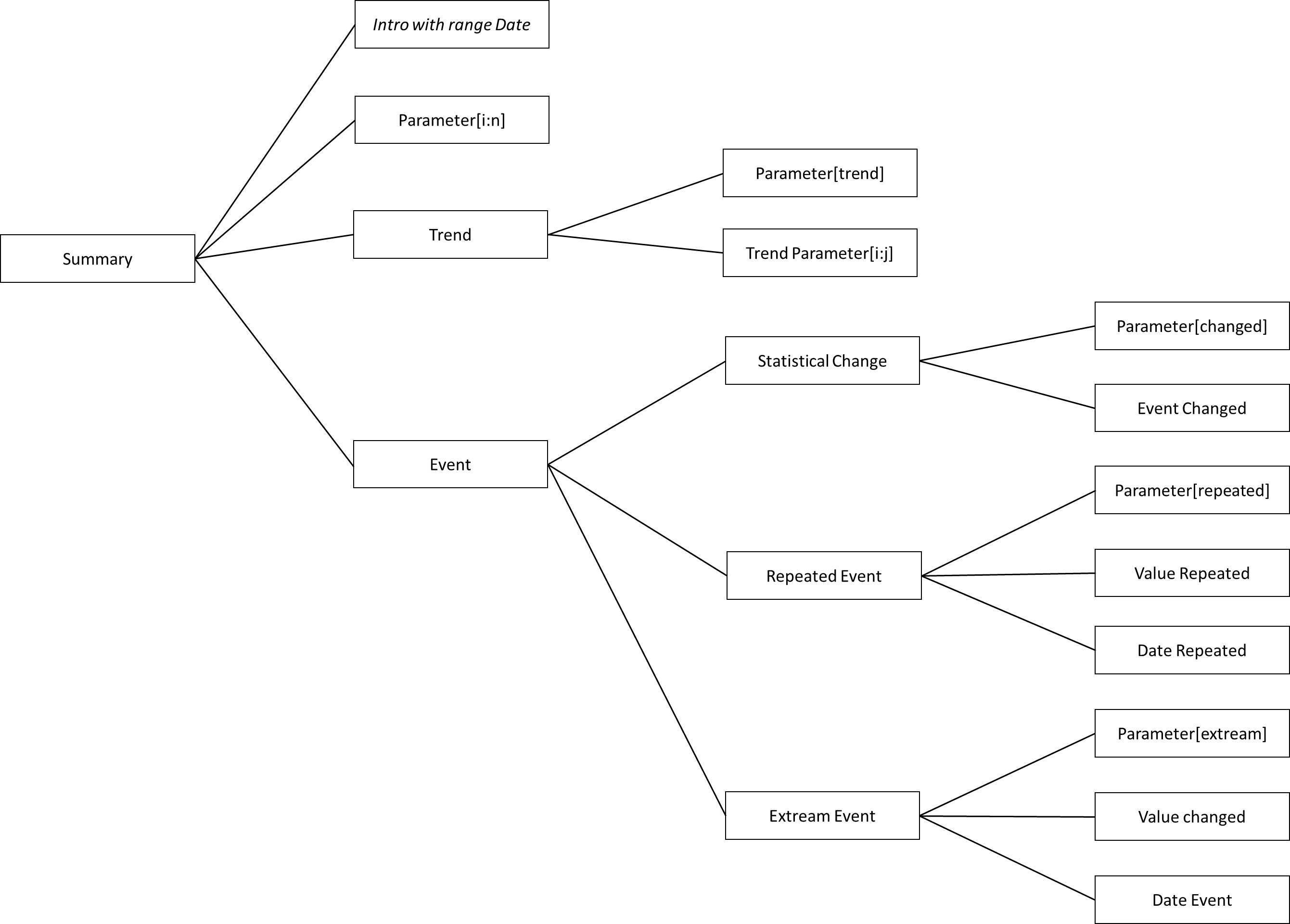
Gambar 4.9 *Initial Corpus* untuk ringkasan data (need to be updated)

Dengan mengikuti langkah-langkah dalam pembentukan *Initial Corpus* yang sudah dijelaskan sebelumnya, peneliti membuat skema yang digambarkan pada gambar 4.9 berikut ini.



Gambar 4.10 Skema teks untuk ringkasan data (need to be updated)

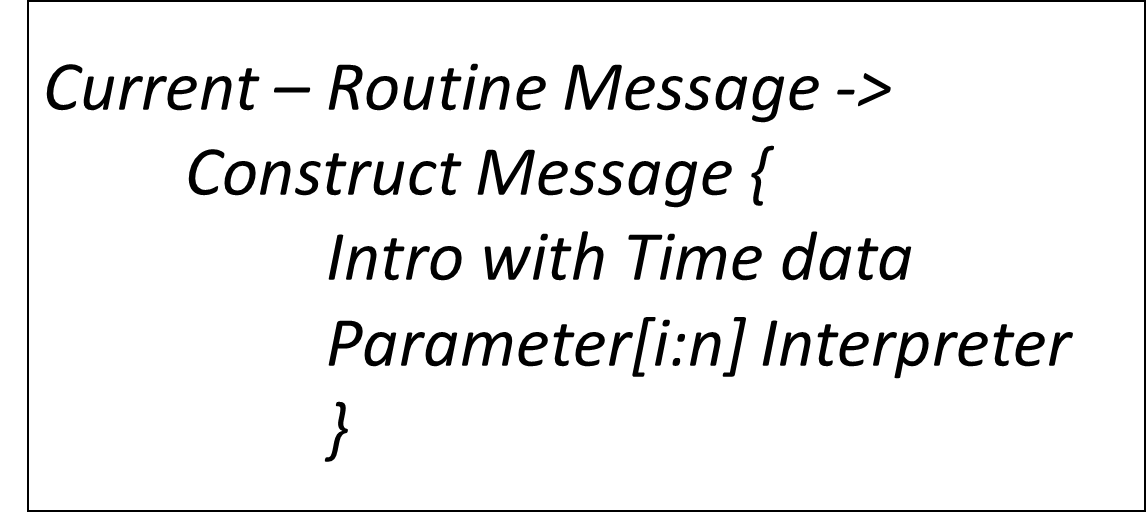
Setelah menentukan skema teks untuk ringkasan data, maka didapatkan pohon struktur dokumen seperti pada gambar 4.11. Pada struktur dokumen terdapan variabel i, j dan n, variabel i menunjukan iterasi dimulai dari i, sedangkan j dan n merupakan batas iterasi, n menunjukan jumlah keseluruhan parameter, sedangkan j jumlah keseluruhan jenis trend. Pada gambar tersebut, struktur dokumen pada ringkasan data terdiri dari *Intro with Range Date*, Parameter ke-i hingga ke-n, *Trend*, dan *Event*. Dimana *Trend* merupakan *Routine Message*, sedangkan *Event* merupakan *Significance Event Message*.



Gambar 4.11 Struktur Pohon untuk teks ringkasan data

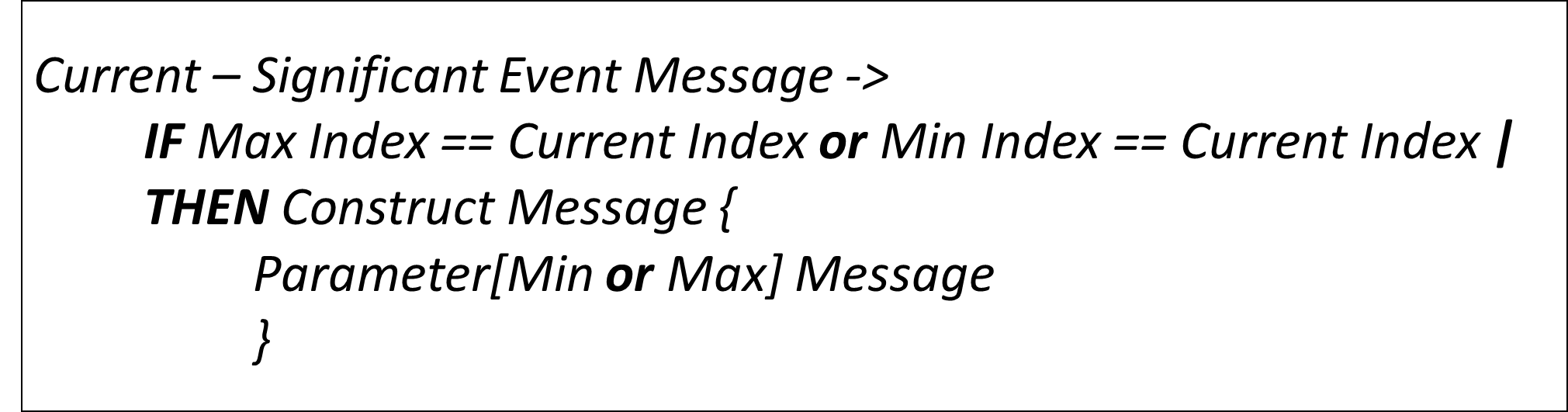
* + - 1. **Perencanaan Dokumen untuk Data Terkini**
      2. *Content Determination*

Pemilihan konten untuk data terkini atau data terkahir hampir sama dengan proses pemilihan konten untuk ringkasan data, dimana konen-konten yang akan ditampilkan terbagi menjadi dua kategori, yakni *Routine Message* dan *Significance Event Message*. Untuk konten *Routine Message* dapat dilihat pada gambar 4.12.

****

Gambar 4.12 *Routine Message* untuk data terkini

Sedangkan penentuan *Significance Event Message* untuk data terkini akan ditampilkan jika data terakhir melebihi atau sama dengan nilai statistik maksimum atau minimum data *statistical summary,* untuk lebih lengkapnya dapat dilihat pada gambar 4.13



Gambar 4.13 *Significance Event Message* untuk data terkini

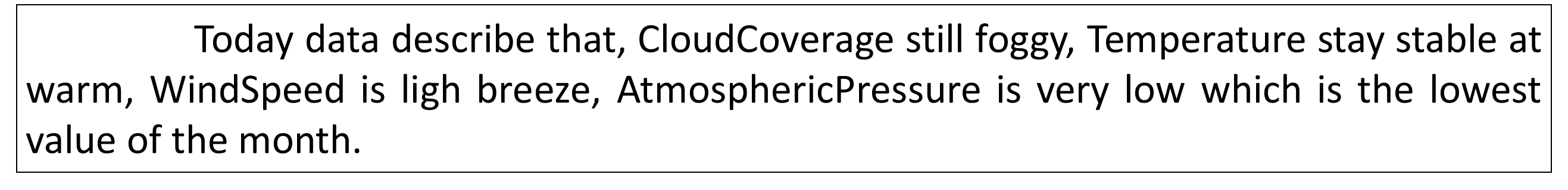
Sebagai contoh jika kita menggunakan data pada klimatologi pada tabel 4.2 maka akan didapatkan hasil *Content Determination* seperti pada tabel 4.14 (need to be updated)

Tabel 4.18 Hasil *Content Determination* untukdata terkini

|  |  |
| --- | --- |
| Routine Message | Cloud Coverage in Foggy condition. Temperature in cold condition. CO in low condition. SO2 in very high condition. |
| Significance Event Message | Cloud Coverage (at the lowest value so far) |

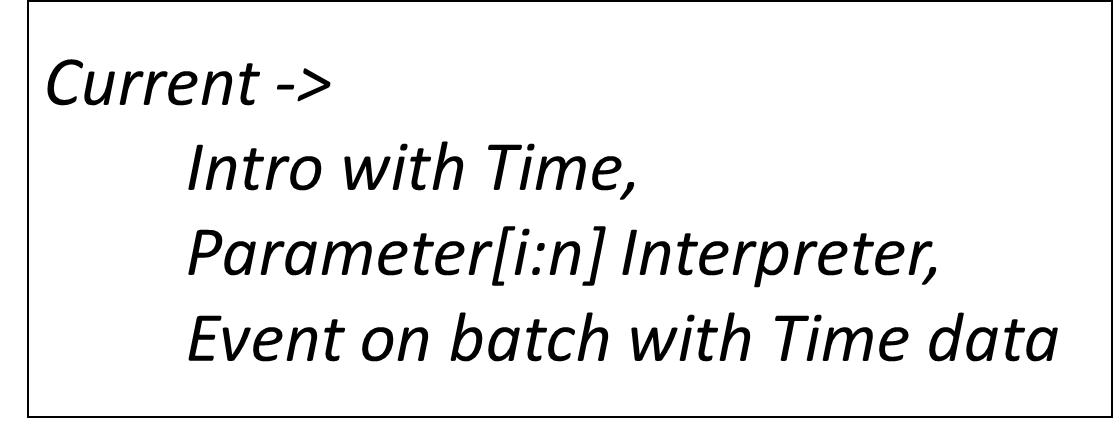
* + - 1. *Document Structuring*

*Initial Corpus* yang digunakan sebagai acuan dalam menampilkan pesan-pesan untuk data terkini dapat dilihat gambar 4.14.



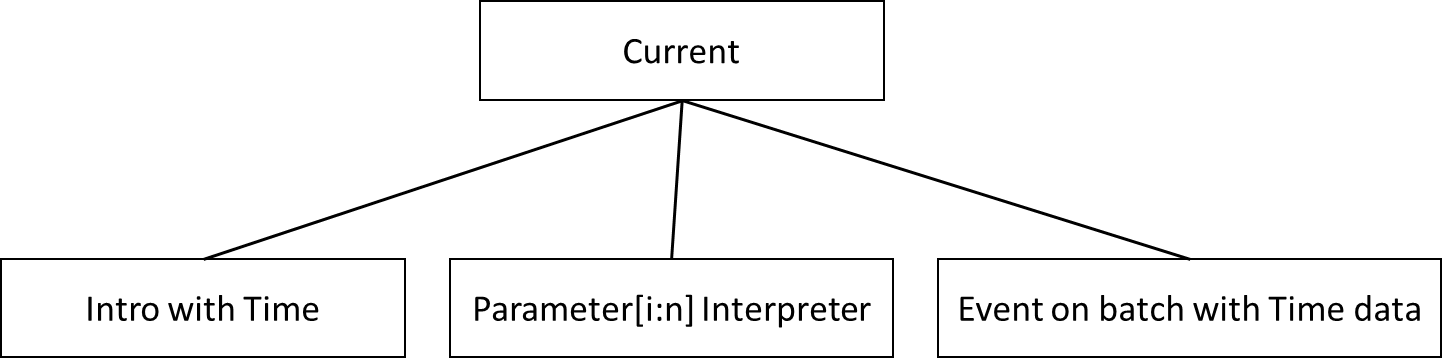
Gambar 4.14 *Initial Corpus* untuk pesan data terkini

Setelah menentukan *Initial Corpus*, maka dilakukan pembuatan skema teks seperti pada ringkasan data, sehingga didapatkan hasil seperti pada gambar 4.15.



Gambar 4.15 Struktur teks data terkini

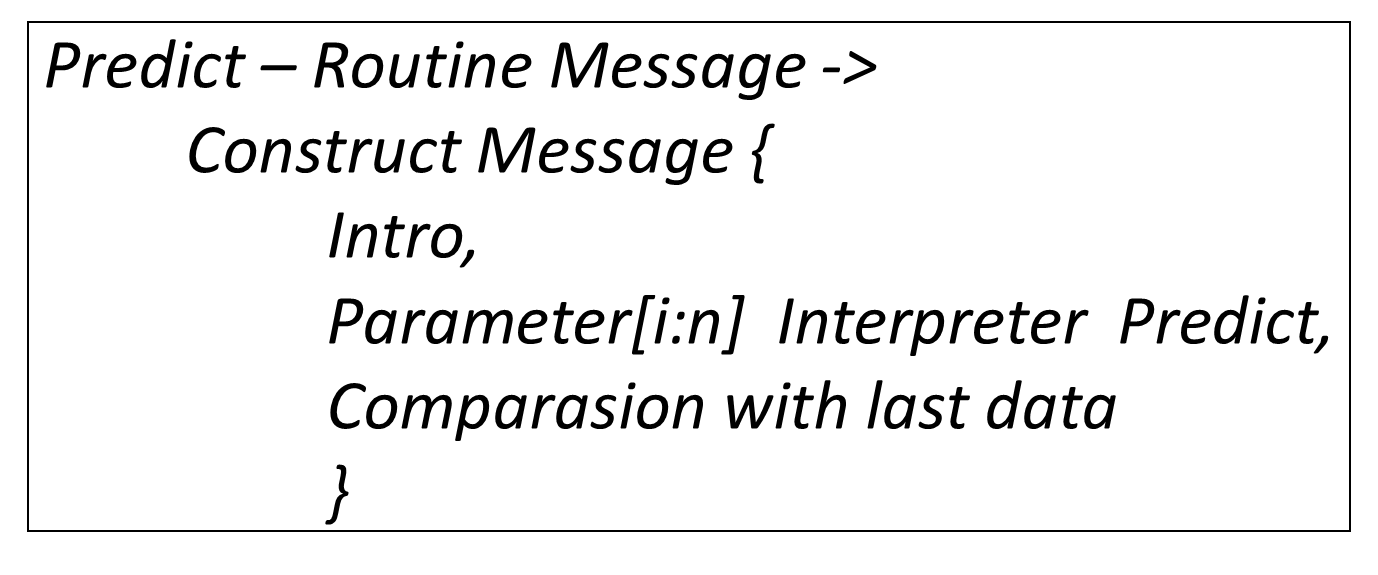
Sedangkan untuk struktur pohonnya dapat dilihat pada gambar 4.16. Pada pohon tersebut, kita bisa menemukan variabel i dan n, variabel i merupakan indeks data yang dimulai dari 1, sedangkan variabel n merupakan banyaknya parameter yang ada, sehingga i:n berarti melakukan perulangan sebanyak parameter yang ada. Struktur dokumen untuk data terkini, terdiri dari *Intro With Time* yang kemudian diikuti dengan interpretasi untuk setiap parameter yang ada, jika kondisi dari *Significant Event Message* terpenuhi, maka ditampilkan juga *Event* bersama waktu datanya.



Gambar 4.16 Pohon struktur untuk teks data terkini

* + - 1. **Perencanaan Dokumen untuk Prediksi**
      2. *Content Determination*

Pada pemilihan konten untuk prediksi, terdapat juga dua bagian yang akan ditampilkan seperti pada bagian sebelumnya, yaitu *Routine Message* dan *Significance Event Message*. Untuk *Routine Message* pada bagian prediksi, dapat dilihat pada gambar 4.17.



Gambar 4.17 *Routine Message* untuk Prediksi data

Sedangkan *Significant Event Message* pada bagian prediksi ini berupa *Special Corpus*. *Special Corpus* ini merupakan teks keluaran yang diadaptasi dari penelitian DWP yang dikembangkan oleh Putra (2017) untuk bagian prediksi. *Special Corpus* terdiri dari tiga bagian, yaitu *Sky State Message*, *Temperature Message*, dan *Air Quality* *Message*. Pesan *Sky State* *Message* akan ditampilkan jika terdapat parameter *Rainfall* dan *CloudCoverage* pada data masukan, begitu juga dengan *Temperature Message* hanya akan ditampilkan jika terdapat parameter *Temperature*, untuk *Air Quality* *Message* akan ditampilkan jika pada data masukan terdapat delapan parameter kualitas udara yang menjadi masukan sistem DWP, yakni CO, NO, NO2, NOX, O3, PM10, PM25, dan SO2. Untuk lebih lengkapnya, *Significant Event Message* ini digambarkan pada gambar 4.18.

*Predict – Significant Event Message ->*

*Construct Message {*

*Special Corpus*

*}*

*Special Corpus ->*

***IF*** *is.available(Rainfall,*

*CloudCoverage) |*

***THEN*** *Construct Message {*

*Sky State Message*

*}*

***IF*** *is.available(Temperature) |*

***THEN*** *Construct Message {*

*Temperature Message*

*}*

***IF*** *is.available(CO, NO, NO2, NOX, O3, PM10, PM25, SO2) |*

***THEN*** *Construct Message {*

*Air Quality Message*

*}*

*Sky State Message ->*

*Construct Message {*

*Rainfall Message,*

*CloudCoverage Message*

*}*

Gambar 4.18 *Significant Event Message* untuk Prediksi data

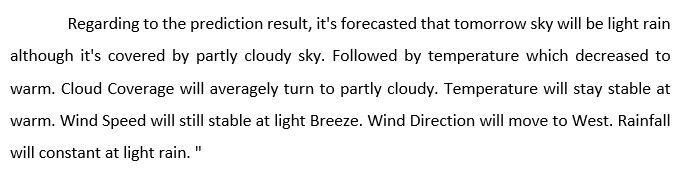
Sebagai contoh jika kita menggunakan data klimatologi pada tabel 4.2 maka akan didapatkan hasil *Content Determination* seperti pada tabel 4.15 (need to be updated)

Tabel 4.19 Hasil *Content Determination* untukPrediksi data

|  |  |
| --- | --- |
| Routine Message | SO2 will increase extremely to very low, Temperature will keep stable at cold, Cloud Coverage, and CO will increase to Partly cloudy, and high. A conclusion of the predicted result is half variable will increase slowly. |
| Significant Event Message | it's predicted that tomorrow sky will be light rain although it's covered by partly cloudy sky. Favored by temperature which decreased to warm. |

* + - 1. *Document Structuring*

*Initial Corpus* yang digunakan sebagai acuan dalam menampilkan pesan-pesan untuk prediksi datadapat dilihat gambar 4.19.



Gambar 4.19 *Initial Corpus* untuk Prediksi Data

Setelah menentukan *Initial Corpus*, maka dilakukan pembuatan skema teks seperti pada ringkasan data, sehingga didapatkan hasil seperti pada gambar 4.20

*Predict ->*

*Intro,*

*Special Corpus,*

*Parameter[i:n] interpreter with-*

*comparsion with last data*

*Special Corpus ->*

*Sky State Message****,***

*Temperature Message*

*Air Quality Message*

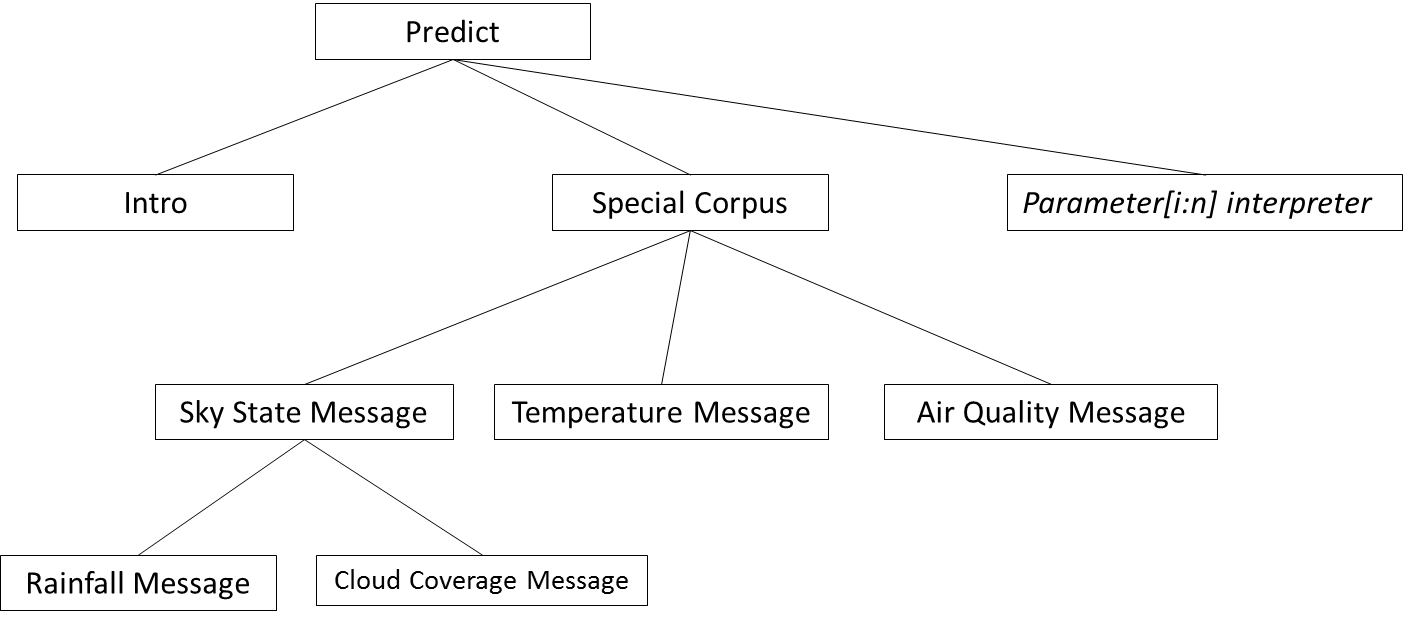
*Sky State Message ->*

*Rainfall Message,*

*CloudCoverage Message*

Gambar 4.20 Struktur teks untuk prediksi data

Sedangkan untuk struktur pohonnya dapat dilihat pada gambar 4.21, dimana struktur pohon tersebut terdiri dari intro, *special corpus*, dan prediksi untuk setiap parameter.



Gambar 4.21 Pohon struktur prediksi data

### Model Proses *Microplanning and Realisation*

Tugas utama dari tahap *Microplanning* ini adalah mengemas pesan-pesan yang sudah dibangkitkan sebelumnya sehingga dapat tersusun sebaik mungkin mengikuti struktur dokumen yang sudah dibuat pada tahap sebelumnya. Proses *Microplanning* ini terbagi menjadi tiga proses yaitu, *Lexicalisation, Aggregation,* dan *Referring Expression Generation.* Proses *lexicalisation* dalam penelitian ini menggunakan pendekatan *rule-based* seperti pada penelitian DWP, dimana setiap pilihan leksikal dibangkitkan berdasarkan rule yang diperoleh dari berbagai sumber. Misalnya, untuk merepresentasikan perubahan setiap parameter pada teks prediksi, pemilihan frase yang tepat akan melalui beberapa tahapan sehingga keluarannya adalah frase seperti “*change progressively”*, “*shifted”*, dan lain-lain. Sedangkan *aggregation* merupakan proses bagaimana pesan yang berkaitan dihubungkan, baik baik pesan yang memiliki kesamaan ataupun pesan yang bertolak belakang. Selanjutnya, *referring expression generation* merupakan bagaimana mendefiniskan subjek informasi, contohnya: “*this month, this day...”* (Putra et al., 2017)*.* Dimana untuk *input,* proses, dan *ouput* pada tahap ini adalah sebagai berikut:

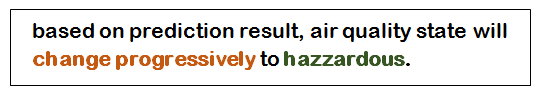
* *Input*: *Content and Structure*
* Proses: *Microplanning and Realisation*
* *Output*: *Summary Text, Current Text,* dan *Predict Text*

Berikut adalah penjelasan mengenai sub-proses dari tahapan *Microplanning and Realisation.*

* + - 1. ***Lexicalisation***

Proses *Lexicalisation* atau leksikalisasi adalah proses untuk menentukan bagaimana konsep, informasi, atau elemen pengetahuannya diungkapkan dalam bentuk kata-kata, sehingga teks yang dihasilkan dapat lebih variatif tanpa mengurangi *knowledge* dari konten yang akan disampaikan. Misalnya pada teks untuk prediksi data, banyak frasa yang dapat merepresentasikan kondisi jika perubahan ekstrim terjadi, seperti “*change progressively to”, “turn progressively to”, “move progressively to”, “shifted progressively to”,* dan lainnya. Proses ini merujuk pada representasi *Trend Description* untuk kualitas udara yang diterapkan dalam DWP (Putra et al., 2017).

Proses leksikalisasi pada sistem DWP mengacu pada proes leksikalisasi yang dilakukan oleh (Ramos-Soto et al., 2016), dimana setidaknya ada dua pesan yang direpresentasikan proses *Trend Description* ini. Yang pertama yaitu *Change Type,* dan yang kedua adalah *Change Label*. Untuk contoh hasil dari proses ini dapat dilihat pada gambar 4.22.



Gambar 4.22 Contoh dari hasil proses *Trend Description*

Frasa *“change progressively”* merupakan pesan *Change Type* atau pesan merepresentasikan perubahan apa yang terjadi pada suatu parameter, yang kemudian diikuti oleh frasa *“hazzardous”* yang merupakan *Change Label* atau frasa yang merepresentasikan hasil terakhir dari perubahan tersebut. Dalam menentukan *Trend Description*, langkah pertama yang harus dilakukan adalah membuat himpunan berisi hasil representasi data kemarin, data terkini, dan data prediksi, contohnya dapat dilihat pada tabel 4.23. Sehingga didapatkan himpunan sebagai berikut:

AQ = {*Admissible, Bad, Hazzardous*}

Tabel 4.20 Hasil interpretasi data untuk contoh kasus kualitas udara

|  |  |  |
| --- | --- | --- |
| Yesterday (aq1) | Today (aq2) | Tomorrow (aq3) |
| **Admissible** | **Bad** | **Hazzardous** |

Kemudian setiap anggota himpunan diinterpretasikan sesuai dengan nilai indeksnya. *Indeks* tersebut didapatkan berdasarkan urutan dalam partisi keanggotaan yang sudah dijelaskan pada bagian *Data Interpretation*, urutan tersebut dimulai dengan indeks 0. Urutan tersebut dapat dilihat pada tabel 4.17.

Tabel 4. 21 Indeks interpretasi kualitas udara

|  |  |
| --- | --- |
| **Partisi** | ***Index*** |
| Good | 0 |
| Admisible | 1 |
| Bad | 2 |
| Hazzardous | 3 |

Setelah dipetakan berdasarkan indeks penginterpretasiannya, kini himpunan AQ yang memuat hasil dari interpretasi beberapa data, menjadi himpunan AQI yang hanya memuat indeks-indeks penginterpretasiannya saja, maka AQI = {1,2,3}. Lalu proses selanjutnya adalah mencari *Index Variation* (IV) dengan persamaan berikut:

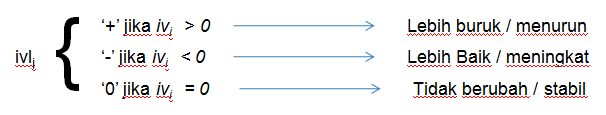
IV ={ *iv1=aqi2-aqi1 , iv2 = aqi3-aqi2, iv3 = aq3*}

Maka didapatkan nilai *Index Variation* dengan perhitungan sebagai berikut:

IV={*iv1* = 2-1, *iv2*= 3-2, *iv3* = “Hazzardous”}

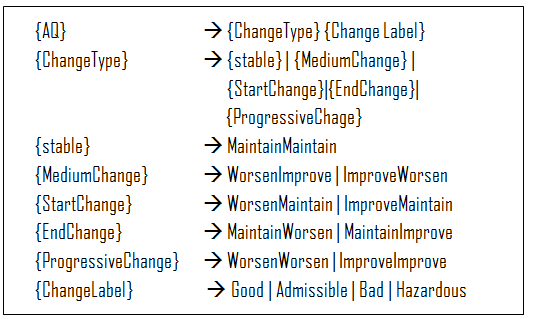
IV = {1,1,”Hazzardous”}

Seteah didapatkan nilai IV, perubahan parameter mulai terlihat, dimana pada proses selanjutnya himpunan IV akan direpresentasikan menjadi *Index Variation* *Lexicalisation* IVL yang berisikan simbol-simbol dengan menggunakan kaidah pada gambar 4.23. Dimana jika IV akan direpresentasikan menjadi simbol “+” jika bernilai positif , “-“ jika negati, dan “0” jika IV bernilai 0.



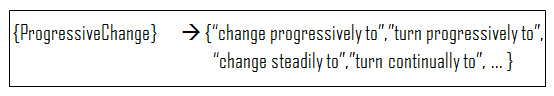
Gambar 4.23 *Rule* penginterpretasian IVL

Dari serangkaian proses diatas, maka dihasilkan nilai dari *Index* *Variation* untuk *Linguistic Description* (LD) kualitas udara adalah : LD*AirQuality = { “+”,”+”,Hazzardous }*. Inilah nilai terakhir yang kemudian nilai ini akan direpresentasikan menggunakan skema pada gambar 4.24 yang diperkenalkan oleh (Ramos-Soto et al., 2016).



Gambar 4.24 Skema untuk mendeskripsikan pesan *Trend Description*

Himpunan LD yang berisi simbol-simbol akan direpresentasikan menjadi sebuah pesan *Trend Description,* dimana simbol “+” berarti “*Improve”* atau lebih baik, lalu simbol “-“ berarti “*Worsen*” atau lebih buruk, sedangkan “0” diartikan sebagai “*Maintain*” atau tidak berubah. Dikarenakan himpunan LD yang dihasilkan pada proses sebelumnya berisi LD*AirQuality = { “+”,”+”,Hazzardous },* maka *Change* *Type* dari pesan tersebut adalah “*ImproveImprove”* yang tergolong pada kategori *ProgressiveChange*. Setelah didapatkan, *Change* *Type* dari pesan tersebut, langkah selanjutnya adalah menentukan frasa yang akan disampaikan pada teks keluaran. Dimana penentuan ini dilakukan dengan cara *random* sesuai dengan frasa yang terdapat pada *corpus* seperti yang terdapat pada gambar 4.25.



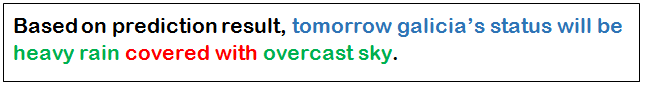
Gambar 4.25 *ProgressiveChange* *Corpus*

Pada penelitian ini, penerapan proses leksikalisasi khususnya *Trend Description* tidak hanya dilakukan untuk parameter kualitas udara saja namun proses ini dilakukan untuk setiap parameter yang terdapat pada teks prediksi data. Dimana data-data yang sudah diinterpretasikan pada proses *Data Interpretation* akan diproses sehingga dihasilkan teks yang lebih variatif dan *general*.

* + - 1. ***Aggregation***

Proses agregasi menentukan bagaimana suatu pesan digabungkan untuk menghasilkan spesifikasi frasa yang sesuai dengan kalimat kompleks (Ramos-Soto et al., 2016). Dalam proses agregasi ini digunakan teknik *Simple Conjunction,* dimana sebuah kalimat dapat digabungkan dengan frasa sederhana seperti “*and”, “but”, “altough” ,* dan lainnya sesuai dengan kebutuhan sistem (Reiter & Dale, 1997).

Salah satu contoh penerapan agregasi adalah saat pembentukan pesan *Sky State* pada *Special Corpus* yang terdiri dari pesan-pesan yang mengandung informasi mengenai hujan dan cakupan awan (Putra et al., 2017). Pada gambar 4.26 Dapat dilihat bahwa kalimat yang dibangun dihubungkan dengan kata penghubung atau *Conjunction* berupa ”covered with” yang menghubungkan dua



Gambar 4.26 Contoh Phrase Aggregation (Putra et al., 2017)

Dalam menentukan kata penghubung yang tepat, sistem perlu mengetahui bagaimana hubungan antara pesan-pesan yang akan dihubungkan nantinya. Contohnya untuk kasus pesan curah hujan dan cakupan awan pada sistem DWP, jika didapatkan pesan curah hujan berupa *“heavy rain”* atau hujan deras dan pesan cakupan awan berupa *“overcast”*  yang berarti mendung, maka pesan ini berada dalam keadaan yang sebandin,g sehingga digunakan kata penghubung *“covered with..”* yang berarti “ditutupi oleh...”. Contoh lainnya, jika pesan curah hujan yang didapatkan adalah *“no rain”* atau tidak ada hujan, namun cakupan awan adalah *“moslty cloudy”* atau umumnya berawan, maka pesan yang bertolak belakang, sehingga digunakan kata penghubung *“although it's covered by”* yang berarti “meskipun ditutupi oleh...” .

Proses selannjutnya adalah penentuan nilai kontras sehingga sistem mampu mengenali mana pesan yang sebanding atau bertolak belakang (Putra et al., 2017). Hal ini hampir sama dengan proses *Trend Description*, dimana penentuan nilai kontras menggunakan bantuan nilai indeks, jika nilai indeks dari kedua pesan yang akan disampaikan bernilai sama, maka pesan tersebut sebanding, jika nilai indeksnya berbeda maka pesan tersebut bertolak belakang. Pemetaan indeks untuk menentukan kontras pada pesan dapat dilihat pada tabel 4.18.

Tabel 4.22 Nilai kontras dalam proses agregasi dengan  
Simple Conjunction (Putra et al., 2017)

|  |  |  |  |
| --- | --- | --- | --- |
| Curah Hujan | | Cakupan Awan | |
| Partisi | Nilai Kontras | Partisi | Nilai Kontras |
| No Rain | 0 | Clear | 0 |
| Light Rain | 0 | Foggy | 0 |
| Moderate Rain | 1 | Mostly sunny | 0 |
| Intense Rain | 1 | Partly cloudy | 1 |
| Torential | 1 | Mostly cloudy | 1 |
|  |  | Broken | 1 |
|  |  | overcast | 1 |

* + - 1. ***Referring Expression Generation***

Pada tahap ini , dilakukan pemilihan kata atau ungkapan untuk merepresentasikan sebuah entitas, sehingga frasa yang digunakan dapat lebih variatif. Tahap ini relatif sederhana, dimana implementasi untuk tahap ini dapat diterapkan secara *hard code* (Reiter & Dale, 1997). Namun, peneliti menerapkan pembangkitan frasa secara *random* seperti yang dilakukan pada sistem DWP (Putra et al., 2017). Contohnya, dalam penentuan *adverb* seperti *“sharply”, “extremly”, “dramatically”, “significantly”,* dan lainnya akan dipilih secara *random* untuk merepresentasikan ketika ada kenaikan atau penurunan ekstrim terjadi.

* + - 1. ***Structure Realisation***

Pada tahap ini, teks direalisasikan kedalam bentuk aktual berdasarkan struktur yang telah dibuat pada saat prose *document planning* (Reiter, 2011)*.*  Realsasi teks dapat ditampilkan menggunakan bahasa pemrograman seperti HTML, LaTeX, RTF, atau lainnya.

Sehingga jika digunakan data nilai tukar pada tabel 4.1 akan didapatkan hasil seperti pada tabel 4.19

Tabel 4.23 Hasil akhir untuk contoh kasus data klimatologi (need to be updated)

|  |
| --- |
| Regarding the dataset (07 Jan 2014 (00:00:00) - 15 Jan 2014 (00:00:00)), with parameters: Cloud Coverage, Temperature, CO and SO2. It shows that Cloud Coverage, Temperature trend is decreased but the rest is increased. Cloud Coverage decreased extremely from 14 Jan 2014 to 15 Jan 2014 (decreased 56.7 points), Temperature decreased significantly from 11 Jan 2014 to 12 Jan 2014 (decreased 2.3 points), CO fluctuate extremely (decreased 3.8 points and increased 4.1 points),There was repeated value in SO2 from 07 Jan 2014 to 13 Jan 2014 with value 4.25.  Based on the data provided, today can be described that Cloud Coverage (at the lowest value so far) in Foggy condition. Temperature in cold condition. CO in low condition. SO2 in very high condition.  Based on prediction result SO2 will increase extremely to very low, Temperature will keep stable at cold, Cloud Coverage, and CO will increase to Partly cloudy, and high. A conclusion of the predicted result is half variable will increase slowly. |

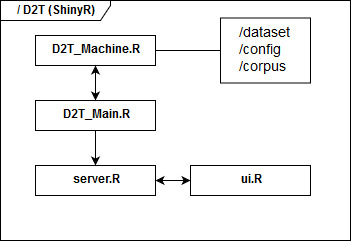
Sedangkan jika digunakan data klimatologi pada tabel 4.2 akan didapatkan hasil seperti pada tabel 4.20

Tabel 4. 24 Hasil akhir untuk contoh kasus data klimatologi

|  |
| --- |
| Regarding the dataset (07 Jan 2014 (00:00:00) - 15 Jan 2014 (00:00:00)), with parameters: Cloud Coverage, Temperature, CO and SO2. It shows that Cloud Coverage, Temperature trend is decreased but the rest is increased. Cloud Coverage decreased extremely from 14 Jan 2014 to 15 Jan 2014 (decreased 56.7 points), Temperature decreased significantly from 11 Jan 2014 to 12 Jan 2014 (decreased 2.3 points), CO fluctuate extremely (decreased 3.8 points and increased 4.1 points),There was repeated value in SO2 from 07 Jan 2014 to 13 Jan 2014 with value 4.25.  Based on the data provided, today can be described that Cloud Coverage (at the lowest value so far) in Foggy condition. Temperature in cold condition. CO in low condition. SO2 in very high condition.  Based on prediction result SO2 will increase extremely to very low, Temperature will keep stable at cold, Cloud Coverage, and CO will increase to Partly cloudy, and high. A conclusion of the predicted result is half variable will increase slowly. |

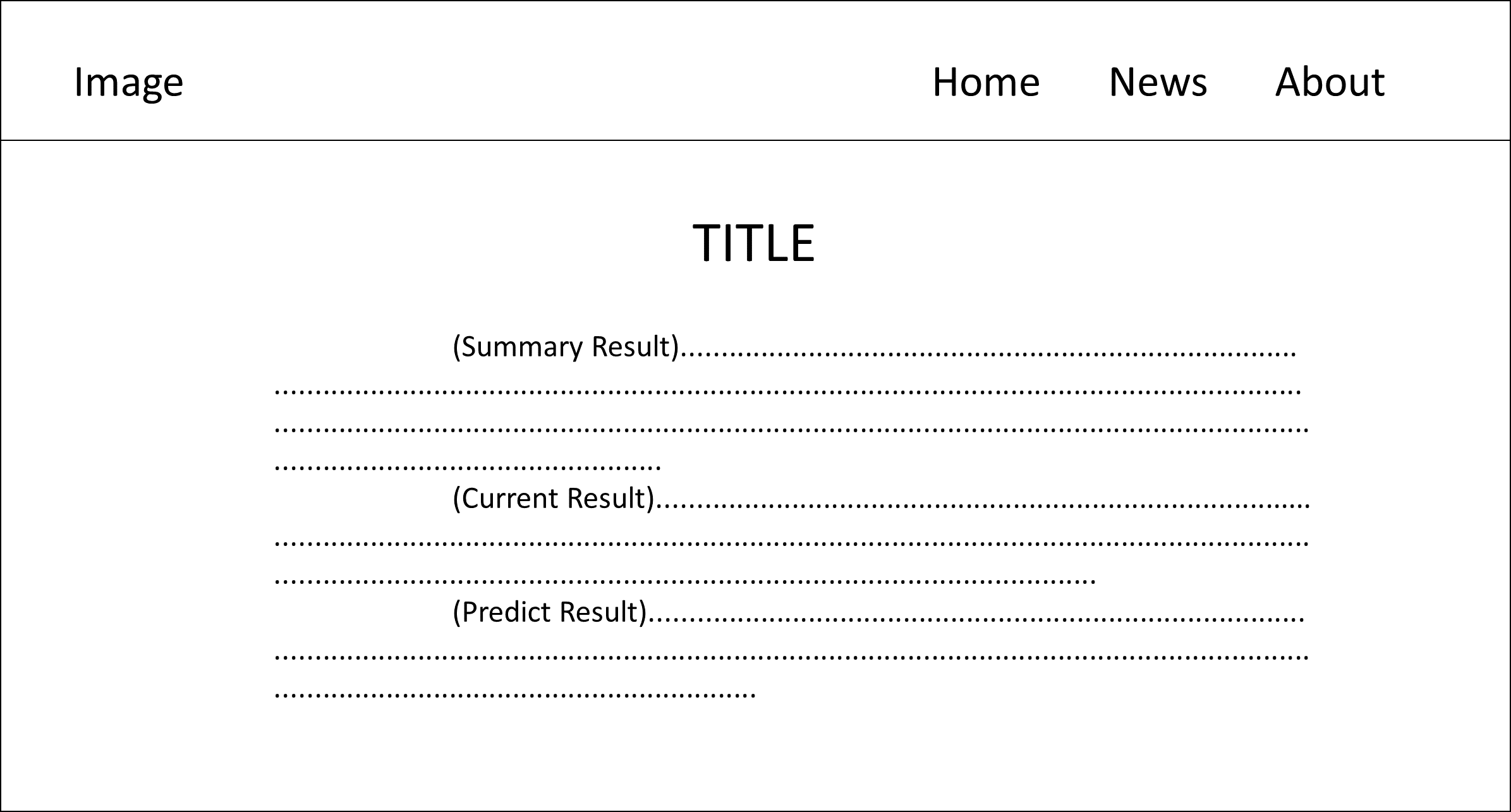
### Model Antarmuka Perangkat Lunak

Sistem dibangun berbasis web menggunakan bahasa pemrograman Javascript, R, dan HTML. Penulis menggunakan *package* yang bernama *ShinyR*, sehingga hasil pemrosesan dari bahasa R akan direpresentasikan ke dalam bentuk web menggunakan *package* tersebut.



Gambar 4.27 Struktur file aplikasi D2T (need to be updated)

Pada gambar 4.27 terlihat bahwa sistem D2T dibangun menggunakan bahasa R dan ditampilkan dalam bentuk web dengan *package* ShinyR. Semua proses implementasi dan fungsi-fungsi utama disimpan pada file *D2T\_Machine.R,* sedangkan file *D2T\_Main.R* berisikan kode program untuk memanggil fungsi-fungsi pada file *D2T\_Machine.R,* selain itu juga file *D2T\_Main.R* akann berkomunikasi langsung dengan *Server.R* sebagai perantara dalam menampilkan antarmuka sistem. Gambar 4.28 menunjukan desain antarmuka sistem yang dibangun, dengan menggunakan template *css,* dan *framework bootstrap.*



Gambar 4.28 Antarmuka sistem *General News Generator* (GNG)

## Implementasi Sistem *Data-to-*text

Pada sub-bab ini akan dipaparkan secara menyeluruh mengenai implementasi sistem *Data-to-Text* secara teknis dengan menggunakan model pada sub-bab 4.2.

### Implementasi Proses *General Data Handler*

Untuk mengimplementasikan proses ini, penulis menggunakan *package data.table* untuk mengecek apakah data masukan memiliki header atau tidak, proses pembacaan data masukan menggunakan fungsi fread() yang terdapat pada *package data.table.* Pada gambar 4.29 terlihat proses pembacaan data dilakukan dengan fungsi fread() yang kemudian dikonversikan kedalam bentuk *data frame*. Penggunaan fungsi fread() ini bertujuan agar parameter yang tidak memiliki *header* akan tetap diproses dan diberi *default header* dengan nama *v2, v3, v4* dan seterusnya, lalu mengubah *header* parameter pertama menjadi *DateTime,* sehingga sistem akan tetap berjalan walaupun data masukan tidak *memiliki* header sekalipun.

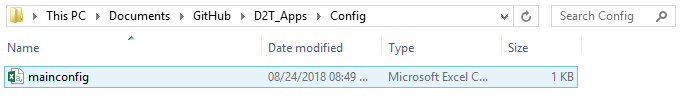
|  |
| --- |
| #-----------------------  # GENERAL DATA HANDLER |  #-----------------------  # Force read, with default parameter v2,v3,v4,etc if there's no header available  dataset **<-** as.data.frame**(**fread**(**file**=**"DatasetsExperiment/  Climatology#1.csv"**))**  # Rename first parameter header to DateTime  colnames**(**dataset**)[**1**]** **<-** "DateTime"  # Dataset with datetime Column dropped  datasetWithoutDate **<-** dataset**[** , colnames**(**dataset**)** **!=** "DateTime"**]**  # Parameter Header  columnName **<-** colnames**(**datasetWithoutDate**)**  # Parameter Config  mainConfig **<-** ReadConfig**()** |

Gambar 4.29 Prose *General Data Handler*

Setelah proses pembacaan selesai, langkah selanjutnya adalah menentukan konfigurasi parameter. Pengkonfigurasian parameter ini dilakukan dengan memanggil fungsi ReadConfig(). Pada gambar 4.30 diperlihatkan bagaimana proses pengkonfigurasian parameter ini dilakukan.

|  |
| --- |
| ReadConfig **<-** **function** **(){**  #Initializing  nullSequence **<-** rep**(NA**, length**(**columnName**))**  dfResult **<-** data.frame**(**ColName **=** columnName, Type **=** nullSequence, Rule **=** nullSequence, Alternate**=**nullSequence, stringsAsFactors**=FALSE)**    # Read config from file  mainConfig **<-** read.table**(**"Config/mainconfig.csv", header**=TRUE**, sep**=**","**)**    # Load setting from default file  i**<-**1  **for(**i **in** i**:**nrow**(**mainConfig**)){**  tempColumn **<-** as.character**(**unlist**(**mainConfig**$**ColName**[**i**]))**  dfResult**[**dfResult**$**ColName **==** tempColumn,**]** **<-** as.vector**(**unlist**(**mainConfig**[**mainConfig**$**ColName **==** tempColumn,**]))**  **}**    # Checking variable type with typeof()  headerClass **<-** ClassHeaderChecker**(**dataset**)**  headerClass **<-** headerClass**[**names**(**headerClass**)!=**"DateTime"**]**    # Merging R config with Default config  i**<-**1  **for(**i **in** i**:**length**(**headerClass**)){**  tempColumn **<-** names**(**headerClass**)[**i**]**    **if(**is.na**(**dfResult**[**dfResult**$**ColName **==** tempColumn,"Type"**])){**  dfResult**[**dfResult**$**ColName **==** tempColumn,"Type"**]** **<-** headerClass**[**names**(**headerClass**)** **==** tempColumn**]**  **}**  **}**    return**(**dfResult**)** |

Gambar 4.30 Fungsi konfigurasi parameter dalam proses *General Data Handler*



Gambar 4.31 File *mainconfig.csv* pada folder *Config*

Dalam proses penkonfigurasian parameter, yang dilakukan pertama kali adalah membaca file *mainconfig.csv* yang terdapat folder *Config* seperti yang dapat dilihat pada gambar 4.31. File ini berisikan konfigurasi berupa tipe dari suatu parameter, cara penginterpretasiannya beserta bagaimana sebuah parameter akan ditampilkan pada teks keluaran nantinya, untuk lebih lengkapnya dapat dilihat pada tabel 4.4. Setelah proses pembacaan selesai, proses selanjutnya adalah menyimpan konfigurasi tersebut ke dalam bentuk *data frame* yang sudah diinisalisasi sebelumnya yaitu variable *mainConfig*. Konfigurasi yang akan disimpan hanyalah konfigurasi untuk parameter-parameter yang menjadi masukan.

Tabel 4.25 Konfigurasi parameter pada file *mainconfig.csv*

|  |  |  |  |
| --- | --- | --- | --- |
| **ColName** | **Type** | **Rule** | **Alternate** |
| AirQuality | numeric | crisp | Air Quality |
| WindSpeed | numeric | crisp | Wind Speed |
| WindDirection | numeric | crisp | Wind Direction |
| CloudCoverage | numeric | crisp | Cloud Coverage |
| Temperature | numeric | fuzzy | NA |
| Rainfall | categorical | fuzzy | NA |
| USD | NA | NA | U.S. Dollar |
| JPY | NA | NA | Japan Yen |
| GBP | NA | NA | Great British Pounds |
| CHF | NA | NA | Confoederatio Helvetica Franc |
| SGD | NA | NA | Singapore Dollar |
| MYR | NA | NA | Malaysian Ringgit |
| HKD | NA | NA | Hong Kong dollar |
| AUD | NA | NA | Australian Dollar |
| CAD | NA | NA | Canadian Dollar |

Setelah disimpan kedalam bentuk variabel *data frame*, sistem akan melakukan pengecekan tipe parameter dengan menggunakan fungsi typeof() pada R. Jika pengguna tidak mendefinisikan tipe parameter pada file *mainconfig.csv,* maka sistem otomatis menggunakan hasil dari fungsi typeof() tersebut sebagai acuan. Namun, jika pengguna sudah mendefinisikan tipe parameter pada file *mainconfig.csv,* maka konfigurasi tersebut yang akan menjadi acuan. Misalnya jika kita menggunakan data klimatologi pada tabel 4.1 maka akan dihasilkan konfigurasi parameter seperti pada tabel 4.22.

Tabel 4.26 Konfigurasi parameter untuk contoh kasus data klimatologi.

|  |  |  |  |
| --- | --- | --- | --- |
| **ColName** | **Type** | **Rule** | **Alternate** |
| WindSpeed | numeric | crisp | Wind Speed |
| WindDirection | numeric | crisp | Wind Direction |
| CloudCoverage | numeric | crisp | Cloud Coverage |
| Temperature | numeric | fuzzy | NA |
| Rainfall | categorical | fuzzy | NA |

Untuk contoh lainnya jika kita menggunakan data nilai tukar pada tabel 4.1 , dikarenakan konfigurasi untuk parameter-parameter tersebut belum didefinisikan sebelumnya, maka akan dihasilkan konfigurasi parameter seperti pada tabel 4.23. Pada tabel 4.23 terlihat konfigurasi yang bernilai NA sebelumnya kini berubah menjadi tipe *numeric*.

Tabel 4.27 Konfigurasi parameter untuk contoh kasus data nilai tukar

|  |  |  |  |
| --- | --- | --- | --- |
| **ColName** | **Type** | **Rule** | **Alternate** |
| USD | numeric | NA | U.S. Dollar |
| JPY | numeric | NA | Japan Yen |
| GBP | numeric | NA | Great British Pounds |
| CHF | numeric | NA | Confoederatio Helvetica Franc |
| SGD | numeric | NA | Singapore Dollar |
| MYR | numeric | NA | Malaysian Ringgit |
| HKD | numeric | NA | Hong Kong dollar |
| AUD | numeric | NA | Australian Dollar |
| CAD | numeric | NA | Canadian Dollar |

### Implementasi Proses *Signal Analyisis*

1. **Ringkasan data**

Implementasi dalam proses peringkasan data, penulis menggunakan fungsi *min, max* dan *mean* yang terdapat pada R seperti pada gambar 4.32 berikut.

|  |
| --- |
| StatisticalAnalysis **<-** **function(**dataset2**){**  dataset2WithoutDate **<-** dataset2**[** , colnames**(**dataset2**)** **!=** "DateTime"**]**  ColName **<-** Average **<-** MaxValue **<-** MaxIndex **<-** MaxDate **<-** MinValue **<-** MinIndex **<-** MinDate **<-** Trend **<-** c**(**""**)**    i**=**1  n**=**length**(**dataset2WithoutDate**)**  **for(**i **in** i**:**n**){**  ColName**[**i**]** **<-** colnames**(**dataset2WithoutDate**[**i**])**  #MAX  MaxValue**[**i**]** **<-** max**(**dataset2WithoutDate**[**i**])**  max\_index2 **<-** as.integer**(**which**(**dataset2WithoutDate**[**i**]==**max**(**dataset2WithoutDate**[**i**])))**  MaxIndex**[**i**]** **<-** max\_index2**[**1**]**  max\_index0 **<-** max\_index2**[**1**]**  MaxDate**[**i**]** **<-** as.character**(**dataset2**$**DateTime**[**max\_index0**])**  #MIN  MinValue**[**i**]** **<-** min**(**dataset2WithoutDate**[**i**])**  min\_index2 **<-** as.integer**(**which**(**dataset2WithoutDate**[**i**]==**min**(**dataset2WithoutDate**[**i**])))**  MinIndex**[**i**]** **<-** min\_index2**[**1**]**  min\_index0 **<-** min\_index2**[**1**]**  MinDate**[**i**]** **<-** as.character**(**dataset2**$**DateTime**[**min\_index0**])**  #AVERAGE  Average**[**i**]** **<-** mean**(**dataset2WithoutDate**[**,i**])**  Trend**[**i**]** **<-** TrendAnalysis**(**ColName**[**i**]**,dataset2WithoutDate**[[**i**]]**,as.double**(**MaxValue**[**i**])**, as.double**(**MinValue**[**i**])**, nrowDataset**)**  **}**  dataset2Statistical **<-** data.frame**(**ColName, MaxDate, MaxValue, MaxIndex, MinDate, MinValue, MinIndex, Average, Trend**)**;  return**(**dataset2Statistical**)**  **}** |

Gambar 4.32 *Statistical Summary Function*

Dalam menentukan *trend,* penulis menggunakan pendekatan *linear model* dengan fungsi lm() yang terdapat pada R. Hasil dari *linear model* tersebut digunakan untuk menentukan kecenderungan apakah *trend* data menaik atau menurun seperti pada gambar 4.33.

|  |
| --- |
| TrendAnalysis **<-** **function(**start,dataset, min, max**){**  # Dataset is vector  plot**(**as.numeric**(**unlist**(**dataset**))**, type**=**"o", col**=**"blue"**)**  dataset **<-** dataset**[**start**:**length**(**dataset**)]**    **if(**length**(**unique**(**dataset**))** **==** 1**){**  result **<-** "0"  **}else{**  x **=** c**(**1**:**length**(**dataset**))**  reg **=** lm**(**dataset**~**x**)**    #linear model range  df **<-** reg**$**coefficients**[**2**]** **+** reg**$**coefficients**[**1**]**  dl **<-** reg**$**coefficients**[**2**]\***length**(**dataset**)** **+** reg**$**coefficients**[**1**]**  range **<-** dl**-**df    #stats  stat **<-** 0  **if(**range **<** 0**){**  range **<-** range **\*** **(-**1**)**  stat **<-** 1  **}**    #dataset range  rangeReal **<-** max**-**min    #5% minimum tershold  **if(**range **>** 0.05**\***rangeReal**){**  **if(**stat **==** 0**){**  result **<-** "+"  **}else{**  result **<-** "-"  **}**  **}else{**  result **<-** "0"  **}**  abline**(**reg,col**=**"red"**)**  **}**    return**(**result**)**  **}** |

Gambar 4.33 *Trend Analysis* *function*

Hasil dari proses ringkasan untuk data klimatologi pada tabel 4.2 dapat dilihat pada tabel 4.24 berikut ini.

Tabel 4.28 Hasil dari ringkasan data klimatologi

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ColName** | **MaxDate** | **Max**  **Value** | **Max**  **Index** | **Min**  **Date** | **Min**  **Value** | **Min**  **Index** | **Sum**  **Value** | **Average** | **Trend** |
| Cloud  Coverage | 08/25/2016 00:00 | 100 | 51 | 04/28/2017 00:00 | 7.9 | 297 | 16891.3 | 46.15 | + |
| Temperature | 06/20/2017 00:00 | 25.3 | 350 | 01/19/2017 00:00 | 1 | 198 | 5126.9 | 14.00 | - |
| WindS  peed | 02/03/2017 00:00 | 29.84 | 213 | 10/28/2016 00:00 | 0.68 | 115 | 2404.51 | 6.56 | + |
| Wind  Direction | 07/06/2016 00:00 | 315 | 1 | 07/13/2016 00:00 | 45 | 8 | 82845 | 226.352 | + |
| Rainfall | 02/03/2017 00:00 | 67.6 | 213 | 07/06/2016 00:00 | 0 | 1 | 940.9 | 2.57 | 0 |

1. ***Extreme Event***

Dalam pendeteksian *Extreme Event*, penulis menggunakan fungsi ResumeEventExtreme(), dengan *input* parameter yang bertipe *numerical*, dan *statistical summary* dari hasil peringkasan data. Kenaikan atau penurunan yang digolongkan ke dalam *extreme event* jika kenaikan atau penurunan lebih besar dari 65% dari interval data (data maksimum – data minimum). Untuk lebih lengkapnya implementasi *Extreme Event* dapat dilihat pada gambar 4.34.

|  |
| --- |
| ResumeEventExtreme **<-** **function(**datasetWithoutCatDate, statisticalResume, type**=NULL){**  # intializing  i **<-** 1  vectorGrowth **<-** c**()**  vectorStartIndex **<-** c**()**  vectorEndIndex **<-** c**()**  vectorInterpreter **<-** c**()**  **for(**i **in** i**:**length**(**datasetWithoutCatDate**)){**  listColumn **<-** datasetWithoutCatDate**[[**i**]]**    # Get highest growth/decay with their index  **if(**type **==** "Growth"**){**  listExtremeAnalysisResult **<-** ResumeHighestGrowthAnalysis**(**diff**(**listColumn**)**,"Growth"**)**  **}else** **if** **(**type **==** "Decay"**){**  listExtremeAnalysisResult **<-** ResumeHighestGrowthAnalysis**(**diff**(**listColumn**)**,"Decay"**)**  **}**    #store growth/decay value, start/end index into vector  vectorGrowth**[**i**]** **<-**listExtremeAnalysisResult**$**valueResult  vectorStartIndex**[**i**]** **<-**listExtremeAnalysisResult**$**startIndexResult  vectorEndIndex**[**i**]** **<-**listExtremeAnalysisResult**$**endIndexResult    #checking if range > 65% data range  vectorInterpreter**[**i**]** **<-** InterpreterExtremeEvent**(**vectorGrowth**[**i**]**, statisticalResume**[**i,**])**  **}**  #exception  vectorEndIndex **<-** vectorEndIndex **+** 1    #Combine all process into df  dfExtremeEvent **<-** data.frame**(**vectorGrowth, vectorStartIndex, vectorEndIndex, vectorInterpreter**)**  **if(**type **==** "Growth"**){**  colnames**(**dfExtremeEvent**)** **<-** c**(**"IncValue", "IncStartIndex", "IncEndIndex", "IncInterpreter"**)**  **}else** **if** **(**type **==** "Decay"**){**  colnames**(**dfExtremeEvent**)** **<-** c**(**"DecValue", "DecStartIndex", "DecEndIndex", "DecInterpreter"**)**  **}**    return**(**dfExtremeEvent**)**  **}** |

Gambar 4.34 *ResumeEventExtreme function*

Dengan menggunakan data nilai tukar pada tabel 4.1 didapatkan hasil seperti pada tabel 4.25, dimana terdapat *extreme event* pada beberapa parameter seperti pada parameter JPY dan CAD terjadi kenaikan yang cukup ekstrem, lalu pada parameter HKD terjadi kenaikan dan penurunan yang ekstrim.

Tabel 4.29 Hasil *Extreme Event Analysis* untuk data nilai tukar

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Column**  **Name** | **Inc**  **Value** | **IncStart**  **Index** | **IncEnd**  **Index** | **IncInterpreter** | **Dec**  **Value** | **DecStartIndex** | **DecEnd**  **Index** | **DecInterpreter** |
| USD | 3.0330 | 91 | 95 | normal | -2.57500 | 6 | 8 | normal |
| JPY | 4.3530 | 92 | 95 | extreme | -2.65900 | 141 | 149 | normal |
| GBP | 3.8430 | 149 | 153 | normal | -3.64550 | 177 | 182 | normal |
| CHF | 2.5500 | 149 | 153 | normal | -1.66092 | 177 | 179 | normal |
| SGD | 1.6080 | 92 | 95 | normal | -1.18442 | 6 | 8 | normal |
| MYR | 1.7140 | 113 | 117 | normal | -1.64500 | 112 | 113 | normal |
| HKD | 1.3619 | 189 | 191 | extreme | -1.33185 | 191 | 200 | extreme |
| AUD | 2.0930 | 86 | 89 | normal | -1.57000 | 89 | 94 | normal |
| CAD | 4.0510 | 24 | 29 | extreme | -3.65800 | 29 | 31 | normal |

1. ***Repeated Event***

Pada pendeteksian sinyal *Repeated Event* dilakukan dengan menggunakan fungsi ResumeRepeatedAnalysis(), yang didalamnya terdapat fungsi rle() atau *run length encoding* dimana hasil dari fungsi rle() adalah sebuah vektor boolean yang menunjukan terjadinya perulangan data, jika ditemukan nilai yang berulang, maka boolean akan bernilai *true*. Perulangan yang termasuk kedalam kategori *Repeated Event* adalah jika perulangan tersebut melebihi 10% dari jumlah data. Gambar 4.35 menunjukan bagaimana proses ini dimplementasikan.

|  |
| --- |
| #Repeated Value analysis  ResumeRepeatedAnalysis **<-** **function(**dataset**){**    lengthEncoding **<-** rle**(**dataset**)**  #limit  n **<-** length**(**dataset**)** **\*** 0.1    repeatedSequence **<-** rep**(**lengthEncoding**$**lengths **>=** n, times**=**lengthEncoding**$**lengths**)**  #Example repeatedSequence  #[1] FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE    RepValue **<-** as.numeric**(**table**(**repeatedSequence**)[**"TRUE"**])**  **if(!**is.na**(**RepValue**)){**  # Get  dt **<-** data.frame**(**number **=** rle**(**repeatedSequence**)$**values, lengths **=** rle**(**repeatedSequence**)$**lengths**)**  # Get the end  dt**$**end **<-** cumsum**(**dt**$**lengths**)**  # Get the start  dt**$**start **<-** dt**$**end **-** dt**$**lengths **+** 1  # Selecting column  dt **<-** dt**[**dt**$**number **==** **TRUE**, c**(**"number", "start", "end"**)]**  result **<-** list**(**RepValue **=** nrow**(**dt**)**, Start **=** dt**$**start, End **=** dt**$**end**)**  **}else{**  result **<-** list**(**RepValue **=** 0, Start **=** 0, End **=** 0**)**  **}**    return**(**result**)**  **}** |

Gambar 4.35 *ResumeRepeatedAnalysis* *function*

Dengan menggunakan data klimatologi pada tabel 4.2 maka didapatkan hasil pada tabel 4. , dimana terdapat perulangan pada parameter *CloudCoverage* dimulai dari indeks ke-1 sampai dengan indeks ke-44 dengan 44 kali perulangan.

Tabel 4.30 Hasil dari *repeated event analysis*

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Rep Value** | **Start** | **End** |
| CloudCoverage | 0 | 0 | 0 |
| Temperature | 0 | 0 | 0 |
| Windspeed | 0 | 0 | 0 |
| WindDirection | 0 | 0 | 0 |
| Rainfall | 44 | 1 | 44 |

1. **Prediksi Data**

Dalam proses prediksi data, penulis menggunakan *package xts* dan *smooth*. Package xts digunakan untuk mengkonversi data masukan kedalam bentuk *time-series* yang kemudian diurutkan berdasarkan *DateTime* secara *ascending.* Sedangkan package *smooth* digunakan untuk melakukan prediksi data dengan *exponential smoothing*. Pada gambar 4.36 proses prediksi data dilakukan didalam fungsi ResumeEventRepeat(), setelah mengkonversi data menjadi bentuk *time-series* dengan menggunakan fungsi xts(), langkah selanjutnya adalah prediksi data dengan fungsi forecast() yang terdapat pada *package smooth*.

|  |
| --- |
| PredictDataset**<-function(**dataset, format**=**"%m/%d/%Y %H:%M"**){**  result **<-** c**()**  lengthWithoutDate **<-** length**(**dataset**[**,**-**which**(**colnames**(**dataset**)** **==** "DateTime"**)])**  # Convert to TS data  dataSeries **<-** xts**(**dataset**[**,**-**which**(**colnames**(**dataset**)** **==** "DateTime"**)]**, order.by**=**as.Date**(**dataset**[**,"DateTime"**]**, format**))**    # Forecasting with Ex.Smooth  i**<-**1  **for(**i **in** i**:**lengthWithoutDate**){**  result**[**i**]** **<-** forecast**(**dataSeries**[**,i**]**, h**=**1**)$**mean  **}**      names**(**result**)** **<-** colnames**(**dataset**[** , colnames**(**dataset**)** **!=** "DateTime"**])**    # print(result)  return**(**result**)**  **}** |

Gambar 4.36 *PredictDataset function*

Setelah dilakukan prediksi data, dengan menggunakan data klimatologi pada tabel 4.2 maka diperoleh hasil prediksi seperti yang ditampilkan pada tabel 4.27.

Tabel 4.31 Hasil dari prediksi data

| CloudCoverage | Temperature | WindSpeed | WindDirection | Rainfall |
| --- | --- | --- | --- | --- |
| 46.9075 | 17.9001 | 7.2226 | 276.331 | 1.0607 |

1. ***String Matching***

Pendeteksian sinyal *string matching* ini hanya diterapkan untuk parameter dengan tipe *categorical.* Sehingga meskipun data masukan memiliki parameter dengan tipe *categorical*, sistem akan tetap bekerja seperti biasa. Proses pendeteksian sinyal *string matching*  ini menggunakan algoritma *Knuth Morris Prat* (KMP). Dimana pola yang akan cari adalah sebanyak *n*  data terakhir, nilai *n* ditentukan berdasarkan dengan interval data, jika data dengan interval per jam, maka diambil data atau pola acuan yang diambil sebanyak 6 baris atau data 6 jam terakhir. Untuk data dengan interval per hari maka data yang diambil sebanyak 7 baris atau data seminggu terakhir. Selebihnya, untuk data dengan interval bulanan atau tahunan, maka data yang diambil sebanyak 4 baris data terakhir. Implementasi proses *string matching* ini terdapat pada fungsi MotifDiscoveryAnalysis() yang dapat dilihat pada gambar 4.37

|  |
| --- |
| MotifDiscoveryAnalysis **<-** **function(**colName, dataset, datasetIntervalValue**){**  #Splitting n last data  n **<-** DataInterpreterInterval**(**datasetIntervalValue, type **=** "limit"**)**  index **<-** length**(**dataset**)+**1 **-** n    #pattern  pattern **<-** dataset**[**index**:**length**(**dataset**)]**    #dataest  dataset **<-** dataset**[**1**:**index**]**  print**(**dataset**)**  print**(**pattern**)**    #KMP Process  result **<-** list**()**  **if(!**is.null**(**KMP**(**dataset, pattern**))){**  result**$**total **<-** length**(**KMP**(**dataset,pattern**))**  result**$**pattern **<-** KMP**(**dataset,pattern**)**  **}else{**  result**$**total **<-** 0  result**$**pattern **<-** **NA**  **}**    return**(**result**)**  **}** |

Gambar 4.37 *MotifDiscoveryAnalysis function*

Fungsi KMP() yang terdapat pada gambar 4.37 mengacu pada implementasi algoritma KMP yang pada bahasa R yang dibuat oleh (Rahman, 2017) pada penelitiannya. Untuk lebih lengkapnya, implementasi algoritma KMP ini dapat dilihat pada gambar 4.38.

|  |
| --- |
| #Function to seacrh the pattern in the string  KMP **<-** **function(**string, pattern**){**    #inisiasi variabel  prefix **<-** KMP\_Prefix**(**pattern**)**  n\_string **<-** length**(**string**)**  n\_pattern **<-** length**(**pattern**)**  index **<-** c**()**  total **<-** 0  i **<-** 0    #Perulangan sesuai dengan jumlah string  **for(**j **in** 1**:**n\_string**){**  **while(**i **>** 0 **&&** pattern**[**i**+**1**]** **!=** string**[**j**]){**  i **<-** prefix**[**i**]**  **}**  **if(**pattern**[**i**+**1**]** **==** string**[**j**]){**  i **<-** i**+**1  **}**  **if(**i **==** n\_pattern**){**  index **<-** c**(**index, j**-**n\_pattern**+**1**)**  total **<-** total**+**1  i **<-** prefix**[**i**]**  **}**  **}**  return**(**index**)**  **}** |

Gambar 4.38 Implementasi algoritma KMP (Rahman, 2017)

Sebelum proses pencocokan pola menggunakan algoritma KMP, diperlukan penentuan *prefix* sebagai acuan dalam proses pencocokan pola nantinya, dimana proses penentuan *prefix* dilakukan pada fungsi KMP\_Prefix dapat dilihat pada gambar 4.39

|  |
| --- |
| #Function for get the prefix from the pattern  KMP\_Prefix **<-** **function(**pattern**){**    #declare variable  n\_pattern **<-** length**(**pattern**)**  prefix **<-** c**(**0**)**  a **<-** 0    #pattern making  **for(**b **in** 2**:**n\_pattern**){**  **while(**a **>** 0 **&&** pattern**[**a**+**1**]** **!=** pattern**[**b**]){**  a **<-** prefix**[**a**]**  **}**  **if(**pattern**[**a**+**1**]** **==** pattern**[**b**]){**  a **<-** a**+**1  **}**  prefix**[**b**]** **<-** a  **}**    #return the result  return**(**prefix**)**    **}** |

Gambar 4.39 Penentuan prefix pada algoritma KMP (Rahman, 2017)

Setelah proses pendeteksian *string matching* selesai, dengan menggunakan data klimatologi pada tabel 4.4 didapatkan hasil seperti pada tabel 4.29

Tabel 4.32 Hasil pendeteksian sinyal *string matching* untuk  
contoh kasus data partikel udara kota beijing

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Total** | **Pattern Index** |
| CBWD | 1 | 477 |

1. **Korelasi antar parameter**

Pendeteksian korelasi parameter ini menggunakan *Pearson Correlation Coefficient*, yang dalam implementasinya menggunakan fungsi cor() pada R. Implementasi proses ini dapat dilihat pada gambar 4.40 ini.

|  |
| --- |
| CorrelationAnalysis **<-** **function(**data**){**  #using pearson correlation coefficient  return**(**cor**(**data**))**  **}** |

Gambar 4.40 *CorrelationAnalysis function.*

Dimana hasil dari fungsi tersebut berupa matriks dengan ukuran *n,n* dimana *n* merupakan jumlah parameter *numerical* pada data masukan. Nilai yang dihasilkan merepresentasikan seberapa kuatnya hubungan linear antar paramater. Dengan menggunakan data partikel udara Kota Beijing pada tabel 4.4 maka didapatkan hasil seperti pada tabel 4.32.

Tabel 4.33 Hasil implementasi *Pearson Correlation*  menggunakan fungsi cor()

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | PM2.5 | DEWP | TEMP | PRES | LWS | IS | IR |
| PM2.5 | 1.0000 | 0.2776 | 0.0377 | -0.1905 | -0.2632 | -0.0130 | -0.0506 |
| DEWP | 0.2776 | 1.0000 | 0.8431 | -0.7687 | -0.3112 | -0.0410 | 0.1220 |
| TEMP | 0.0377 | 0.8431 | 1.0000 | -0.8095 | -0.1780 | -0.0997 | 0.0500 |
| PRES | -0.1905 | -0.7687 | -0.8095 | 1.0000 | 0.2120 | 0.0629 | -0.0592 |
| IWS | -0.2632 | -0.3112 | -0.1780 | 0.2120 | 1.0000 | 0.0237 | -0.0372 |
| IS | -0.0130 | -0.0410 | -0.0997 | 0.0629 | 0.0237 | 1.0000 | -0.0098 |
| IR | -0.0506 | 0.1220 | 0.0500 | -0.0592 | -0.0372 | -0.0098 | 1.0000 |

Setelah didapatkan nilai korelasi pada tabel 4.32, proses selanjutnya adalah menentukan parameter yang mana yang memiliki dampak paling besar terhadap parameter lain. Cara ini dilakukan dengan mengubah semua data pada tabel 4.32 ke dalam bentuk positif terlebih dauhulu menggunakan fungsi abs(), lalu dicari nilai rata-ratanya untuk setiap parameter, proses ini terdapat pada fungsi CorrelationRoutineMessage() yang dapat dilihat pada gambar 4.41.

|  |
| --- |
| CorrelationRoutineMessage **<-** **function(**corMatrix**){**  #Correlation Routine Message Analysis  #Mean with absolute value  coreMean **<-** apply**(**abs**(**corMatrix**)**, 2, mean**)**  #get the highest mean with their index  highestMean **<-** max**(**coreMean**)**  highestIndex **<-** which.max**(**coreMean**)**    message **<-** correlationRoutineDocPlan**(**highestMean, names**(**highestIndex**))**  return**(**message**)**  **}** |

Gambar 4.41 *CorrelationRoutineMessage function*

Setelah dilakukan proses pada gambar 4.41, maka didapatkan hasil seperti pda tabel 4.33 dimana nilai tertinggi ada pada parameter DEWP yang nantinya akan diproses menjadi pesan *routine* untuk korelasi parameter.

Tabel 4.34 Nilai-rata rata korelasi tiap parameter

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| PM2.5 | DEWP | TEMP | PRES | LWS | IS | IR |
| 0.2618 | 0.4805 | 0.4311 | 0.4433 | 0.2893 | 0.1786 | 0.1898 |

Setelah didapatkan pesan *routine*, maka dilakukan proses analisis sinyal untuk pesan *significant*, analisis dilakukan dengan memproses data pada tabel 4.32 lalu mengambil nilai yang lebih besar dari 0.7 untuk nilai positif, dan lebih kecil dari -0.7 untuk nilai negatif. Proses ini dilakukan pada fungsi CorrelationSignificantMsgContentDetermination() yang dapat dlihat pada gambar 4.42

|  |
| --- |
| CorrelationSignificantMsgContentDetermination **<-** **function(**matrix**){**  # ContentDetermination  # Only showing var more than 0.7    matrix**[!**lower.tri**(**matrix**)]** **<-** 0  matrix **<-** as.data.frame**(**matrix**)**  matrix**[**matrix **<** 0.7 **&** matrix **>=** 0 **|** matrix **>** **-**0.7 **&** matrix **<=** 0**]** **<-** 0    return**(**matrix**)**  **}** |

Gambar 4. 42 CorrelationSignificantMsgContentDetermination function

Setelah dilakukan proses pada gambar 4.42 maka didapatkan hasil seperti pada tabel 4.34. Nilai inilah yang akan diambil untuk ditampilkan sebagai *significant message* untuk korelasi parameter. Jika tidak didapatkan hasil pada proses ini, maka *significant message* untuk proses korelasi parameter tidak akan ditampilkan.

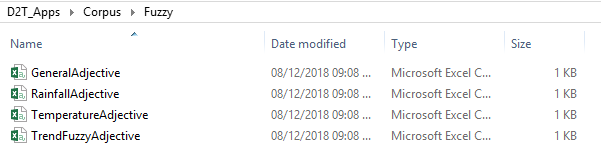
Tabel 4.35 Nilai yang akan diproses sebagai *significant message*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | PM2.5 | DEWP | TEMP | PRES | LWS | IS | IR |
| PM2.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| DEWP | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| TEMP | 0 | 0.8431 | 0 | 0 | 0 | 0 | 0 |
| PRES | 0 | -0.7687 | -0.8095 | 0 | 0 | 0 | 0 |
| IWS | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| IS | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| IR | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### Implementasi Proses *Data Interpretation*

Pada tahap ini implementasi dilakukan dengan cara menginterpretasikan setiap parameter sesuai cara penginterpretasian yang terdapat pada konfigurasi parameter. Jika tipe penginterpretasiannya menggunakan *fuzzy* maka akan digunakan *Fuzzy Membership Function*, jika pentinterpretasiannya adalah *crisp* maka digunakan *Crisp Membership Function* dalam proses penginterpretasiannya. Namun, jika tidak terdapat cara penginterpretasian atau konfigurasi *Rule* pada konfigurasi parameter, maka digunakan *General Membership Function*.

Fungsi keanggotaan bagi parameter yang sudah didefinisikan cara penginterpretasiannya disimpan dalam file berbentuk *csv* (separator koma) dengan format *file* [parameter]*Adjective.csv* lalu disimpan pada folder *Corpus/Fuzzy* atau *Corpus/Crisp* sesuai dengan tipe penginterpretasiannya. Seperti pada gambar 4.43, sehingga sistem dapat bekerja secara *general* untuk data apapun sesuai kebutuhan pengguna.



Gambar 4.43 *Fuzzy Corpus for Data Interpretation*

Proses interpretasi ini dilakukan dengan memanggil fungsi DataInterpreterAdjective(), fungsi ini bekerja dengan cara menyamakan nama kolom pada data dengan daftar parameter yang ada, jika terdapat dalam konfigurasi parameter maka lakukan data interpretasi sesuai fungsi keanggotannya, jika tidak terdapat cara penginterpretasiannya maka proses ini menggunakan cara *general*. Fungsi DataInterpreterAdjective() ini dapat dilihat pada gambar 4.44.

|  |
| --- |
| DataInterpreterAdjective **<-** **function(**value, type**=**"General",statisticalResume**=NULL){**  **if(!**is.na**(**mainConfig**[**mainConfig**$**ColName **==** type,**]$**Rule**)){**  **if(**mainConfig**[**mainConfig**$**ColName **==** type,**]$**Rule **==** "fuzzy"**){**  corpus **<-** read.table**(**file**=**paste0**(**"Corpus/Fuzzy/",type,"Adjective.csv"**)**, sep**=**",", header**=TRUE)**  **if(**type **==** "Rainfall" **&&** value **==** 0**){**  result **<-** list**(**InterpreterResult **=** as.character**(**"no rain"**)**, InterpreterIndex **=** 0**)**  **}else{**  result **<-** MembershipFuzzy**(**value, corpus**)**;  **}**  **}else{**  corpus **<-** read.table**(**file**=**paste0**(**"Corpus/Range/",type,"Adjective.csv"**)**, sep**=**",", header**=TRUE)**  result **<-** MembershipClassifier**(**value, corpus**)**;  **}**  **}else{**  corpus **<-** GeneralFuzzyGenerator**(**type, statisticalResume**)**  result **<-** MembershipFuzzy**(**value, corpus**)**;  **}**    # print(result)  return**(**result**)**  **}** |

Gambar 4. 44 *DataInterpreterAdjective function*

Jika tipe penginterperetasian suatu parameter adalah *fuzzy* maka akan digunakan *Fuzzy Membership Function* yang diimplementasikan pada fungsi MembershipFuzzy() seperti pada gambar 4.45.

|  |
| --- |
| MembershipFuzzy **<-** **function(**value, corpus**){**  **if(**is.null**(**corpus**)){**  return **(**list**(**InterpreterResult **=** as.character**(**"Constant"**)**, InterpreterIndex **=** 0**))**  **}**  i **<-** 1;  n **<-** nrow**(**corpus**)**;  m **<-** length**(**corpus**)**;    membershipValue **<-** c**()**  **for(**i **in** i**:**n**){**  v1**<-**corpus**[**i, "v1"**]**;  v2**<-**corpus**[**i, "v2"**]**;  v3**<-**corpus**[**i, "v3"**]**;  v4**<-**corpus**[**i, "v4"**]**;    ##/ ¯ \ <- 1st area, 2nd area, 3rd area  #first area  **if((**value**>=**v1**)&&(**value**<=**v2**)){**  membershipValue**[**i**]** **<-** **(** **(**value**-**v1**)** **/** **(**v2**-**v1**)** **)**;  #second area (optimum)  **}else** **if((**value**>**v2**)&&(**value**<=**v3**)){**  membershipValue**[**i**]** **<-** 1;  #third area  **}else** **if((**value**>**v3**)&&(**value**<=**v4**)){**  membershipValue**[**i**]** **<-** **(** **(**v4**-**value**)** **/** **(**v4**-**v3**)** **)**;  #fourth, default condition (outside)  **}else{**  membershipValue**[**i**]** **<-** 0;  **}**      **if(**is.nan**(**as.numeric**(**membershipValue**[**i**]))){**  membershipValue**[**i**]** **<-** 9999  **}**  **}**    print**(**membershipValue**)**  #check highest membership result  interpreterResult **<-** corpus**[**which.max**(**membershipValue**)**, "Category"**]**  interpreterIndex **<-** which**(**interpreterResult **==** corpus**$**Category**)**  return **(**list**(**InterpreterResult **=** as.character**(**interpreterResult**)**, InterpreterIndex **=** interpreterIndex**))**  **}** |

Gambar 4.45 *Fuzzy Membership Function*

Namun jika penginterpretasian suatu data adalah *crisp* maka akan digunakan *Crsip Membershi Function* yang diimplementasikan pada fungsi MembershipClassifier() yang dapat dilihat pada gambar 4.46

|  |
| --- |
| MembershipClassifier **<-** **function(**value, corpus**){**    interpreterResult **<-** sapply**(**value, **function(**v**)** corpus**[**v **>=** corpus**[**"Lower"**]** **&** v **<** corpus**[**"Upper"**]**,"Category"**])**  interpreterIndex **<-** which**(**interpreterResult **==** corpus**$**Category**)**  return **(**list**(**InterpreterResult **=** as.character**(**interpreterResult**)**, InterpreterIndex **=** interpreterIndex**))**  **}** |

Gambar 4.46 *Crisp Membership Function*

Untuk parameter yang tidak memiliki cara penginterpretasian maka akan digunakan *General Memership Function* yang diimplementasikan pada fungsi GeneralFuzzyGenerator() seperti pada gambar 4.46 yang kemudian diterapkan fungsi keanggotaan *Fuzzy Membership Function* pada gambar 4.45 sehingga didapatkan hasil penginterpretasiannya.

|  |
| --- |
| GeneralFuzzyGenerator **<-function(**type, statisticalResume**){**  corpus **<-** read.table**(**file**=**paste0**(**"Corpus/Fuzzy/GeneralAdjective.csv"**)**, sep**=**",", header**=TRUE)**  maxRange **<-** as.character**(**statisticalResume**[**statisticalResume**$**ColName **==** type,"MaxValue"**])**  minRange **<-** as.character**(**statisticalResume**[**statisticalResume**$**ColName **==** type,"MinValue"**])**    listGeneralPartition **<-** list**()**  **if(**minRange **==** maxRange**){**  return**(NULL)**  **}else{**  n **=** nrow**(**corpus**)**  node **=** **(**2**\***n**)+**n**-**1    maxRange **<-** as.double**(**maxRange**)**  minRange **<-** as.double**(**minRange**)**  rangenode **=** **(**maxRange**-**minRange**)/**node    i**=**1  j**=**0  membershipValue **<-** c**()**  **for** **(**i **in** i**:**n**)** **{**  **if(**i **==** 1**){**  v1**<-**minRange;  v2**<-**minRange;  v3**<-**minRange**+(**2**\***rangenode**)**;  v4**<-**minRange**+(**3**\***rangenode**)**;    j **<-** i**+**1  listGeneralPartition**[[**i**]]** **<-** c**(**v1,v2,v3,v4**)**  **}else{**  v1**<-**minRange**+(**j**)\***rangenode;  v2**<-**minRange**+(**j**+**1**)\***rangenode;  v3**<-**minRange**+(**j**+**3**)\***rangenode;  v4**<-**minRange**+(**j**+**4**)\***rangenode;      listGeneralPartition**[[**i**]]** **<-** c**(**v1,v2,v3,v4**)**  **}**  **}**  **}**    v1 **<-**unlist**(**lapply**(**listGeneralPartition, `**[[**`, 1**))**  v2 **<-**unlist**(**lapply**(**listGeneralPartition, `**[[**`, 2**))**  v3 **<-**unlist**(**lapply**(**listGeneralPartition, `**[[**`, 3**))**  v4 **<-**unlist**(**lapply**(**listGeneralPartition, `**[[**`, 4**))**    result **<-** data.frame**(**Category**=**corpus**$**Category, v1, v2, v3, v4**)**    return**(**result**)**  **}** |

Gambar 4.47 *GeneralFuzzyGenerator Function*

* + - 1. **Data Umum**

Implementasi *rule-based* untuk data umum atau *general* didapatkan dari hasil perhitungan menggunakan fungsi GeneralFuzzyGenerator() pada gambar 4.47 sehingga jika digunakan parameter USD untuk data nilai tukar pada tabel 4.1 akan didapatkan hasil seperti pada tabel 4.36 dimana batas atas dan bawahnya merupakan nilai maksimum dan nilai minimum dari parameter USD tersebut.

Tabel 4.36 *General Membership value* untuk parameter USD

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **v1** | **v2** | **v3** | **v4** |
| very low | 8.27900 | 8.27900 | 9.190143 | 9.645714 |
| low | 9.190143 | 9.645714 | 10.556857 | 11.012429 |
| medium | 10.556857 | 11.012429 | 11.923571 | 12.379143 |
| high | 11.923571 | 12.379143 | 13.290286 | 13.745857 |
| very high | 13.290286 | 13.745857 | 14.657 | 15.112571 |

* + - 1. **Korelasi Parameter**

Implementasi *rule-based* untuk korelasi parameter seperti pada tabel 4.37 yang disimpan dalam file *CorrelationAdjective.csv* pada folder *Corpus/Crisp* dengan tipe Crisp.

Tabel 4.37 *Correlation Crisp Membership Value*

| **Category** | **Lower** | **Upper** |
| --- | --- | --- |
| no linear association | 0 | 0 |
| insubtantial relationship | 0 | 0.1 |
| low relationship | 0.1 | 0.3 |
| moderate relationship | 0.3 | 0.5 |
| strong relationship | 0.5 | 0.7 |
| very strong relationship | 0.7 | 0.9 |
| almost perfect relationship | 0.9 | 1 |

* + - 1. **Kecepatan Angin**

Implementasi *rule-based* untuk kecepatan angin seperti pada tabel 4.38 yang disimpan dalam file *WindSpeedAdjective.csv* pada folder *Corpus/Crisp* dengan tipe Crisp.

Tabel 4.38 *Wind Speed Crisp Membership Value*

| **Category** | **Lower** | **Upper** |
| --- | --- | --- |
| Calm | 0 | 3 |
| Light Air | 3 | 6 |
| Light Breeze | 6 | 12 |
| Gentle Breeze | 12 | 20 |
| Moderate Breeze | 20 | 30 |
| Fresh Breeze | 30 | 40 |
| Strong Breeze | 40 | 51 |
| Near Gale | 51 | 62 |
| Gale | 62 | 74 |
| Strong Gale | 74 | 88 |
| Storm | 88 | 103 |
| Violent Storm | 103 | 119 |
| Hurricane | 119 | 130 |

* + - 1. **Arah Angin**

Implementasi *rule-based* untuk kecepatan angin seperti pada tabel 4.39 yang disimpan dalam file *WindDirectionAdjective.csv* pada folder *Corpus/Crisp* dengan tipe Crisp.

Tabel 4.39 Wind Direction Crisp Membership Value

| **Category** | **Lower** | **Upper** |
| --- | --- | --- |
| North | 0 | 11.25 |
| North North East | 11.25 | 33.75 |
| North East | 33.75 | 56.25 |
| East North East | 56.25 | 78.75 |
| East | 78.75 | 101.25 |
| East South East | 101.25 | 123.75 |
| South East | 123.75 | 146.25 |
| South South East | 146.25 | 168.75 |
| South | 168.75 | 191.25 |
| South South West | 191.25 | 213.75 |
| South West | 213.75 | 236.25 |
| West South West | 236.25 | 258.75 |
| West | 258.75 | 281.25 |
| West North West | 281.25 | 303.75 |
| North West | 303.75 | 326.25 |
| North North West | 326.25 | 348.75 |
| North | 348.75 | 360 |

* + - 1. **Cakupan Awan**

Implementasi *rule-based* untuk kecepatan angin seperti pada tabel 4.40 yang disimpan dalam file *CloudCoverageAdjective.csv* pada folder *Corpus/Crisp* dengan tipe Crisp.

Tabel 4.40 Cloud Coverage Crisp Membership Value

| **Category** | **Lower** | **Upper** |
| --- | --- | --- |
| Clear | 0 | 11 |
| Foggy | 11 | 21 |
| Mostly sunny | 21 | 31 |
| Partly cloudy | 31 | 51 |
| Mostly cloudy | 51 | 71 |
| Broken | 71 | 81 |
| Overcast | 81 | 100 |

* + - 1. **Kualitas Udara**

Dalam melakukan implementasi *rule-based* untuk kualitas udara ini hanya akan dilakukan jika semua parameter kualitas udara ada dalam data masukan. Interpretasi kualitas udara ini dilakukan dengan menghitung nilai PSI untuk kualitas udara yang ada pada fungsi AirQualityCalculation() (Putra et al., 2017) seperti pada gambar 4.41 dengan modifikasi pengecekan parameter yang ada, sehingga intepretasi kualitas udara tidak akan dilakukan jika parameter PM25, PM10, CO, SO2, dan O3 tidak terdapat pada dataset.

|  |
| --- |
| AirQualityCalculation **<-** **function** **(**dataset**){**  #menghitung sub-index value dari variabel PM25  #b2 = polutant concentration breakpoint 2, a2 = PSI Index breakpoint 2  #b1 = polutant concentration breakpoint 1, a1 = PSI Index breakpoint 1  a2**<-**a1**<-**b2**<-**b1**<-**PM25\_PSI\_value**<**-PM10\_PSI\_value**<-**O\_PSI\_value<- NO2\_PSI\_value**<-**SO2\_PSI\_value**<-**O3\_PSI\_value **<-** 0;    #CHECKING PROCES  ...  PM25\_PSI\_value **<-** **((**a2**-**a1**)/(**b2**-**b1**))\*(**PM25**-**b1**)+**a1  PM10\_PSI\_value **<-** **((**a2**-**a1**)/(**b2**-**b1**))\*(**PM10**-**b1**)+**a1  SO2\_PSI\_value **<-** **((**a2**-**a1**)/(**b2**-**b1**))\*(**SO2**-**b1**)+**a1  CO\_PSI\_value **<-** **(((**a2**-**a1**)/(**b2**-**b1**))\*(**CO**-**b1**)+**a1**)**  O3\_PSI\_value **<-** **(((**a2**-**a1**)/(**b2**-**b1**))\*(**O3**-**b1**)+**a1**)**    PSI\_data **<-** c**(**PM25\_PSI\_value,PM10\_PSI\_value,SO2\_PSI\_value,CO\_PSI\_value,O3\_PSI\_value**)**  PSI\_value **<-** as.integer**(**max**(**PSI\_data**))**  return**(**PSI\_value**)**;  **}** |

Gambar 4.48 Implementasi perhitungan PSI untuk kualitas udara (Putra et al., 2017)

Sehingga setelah implementasi pada gambar 4.48 dilakukan, maka didapatkan implementasi *rule-based* untuk kualitas udara seperti pada tabel 4.41 yang disimpan dalam file *AirQualityAdjective.csv* pada folder *Corpus/Crisp* dengan tipe Crisp.

Tabel 4.41 *Air Quality Crisp Membership Value*

|  |  |  |
| --- | --- | --- |
| **Category** | **Lower** | **Upper** |
| Good | 0 | 51 |
| Moderate | 51 | 101 |
| Unhealthy | 101 | 201 |
| Very Unhealthy | 201 | 301 |
| Hazzardous | 301 | 501 |

* + - 1. **Suhu**

Implementasi *rule-based* untuk kecepatan angin seperti pada tabel 4.42 yang disimpan dalam file *TemperatureAdjective.csv* pada folder *Corpus/Fuzzy* dengan tipe Fuzzy.

Tabel 4.42 *Temperature Fuzzy Membership Value*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **v1** | **v2** | **v3** | **v4** |
| very cold | 0 | 0 | 5 | 10 |
| Cold | 5 | 10 | 15 | 20 |
| Warm | 15 | 20 | 25 | 30 |
| Hot | 25 | 30 | 35 | 40 |
| very hot | 35 | 40 | 45 | 50 |

* + - 1. **Curah Hjan**

Implementasi *rule-based* untuk kecepatan angin seperti pada tabel 4.43 yang disimpan dalam file *RainfallAdjective.csv* pada folder *Corpus/Fuzzy* dengan tipe Fuzzy.

Tabel 4.43 *Rainfall Fuzzy Membership Value*

| **Category** | **v1** | **v2** | **v3** | **v4** |
| --- | --- | --- | --- | --- |
| light | 0 | 0 | 2.5 | 2.75 |
| moderate | 2.5 | 2.75 | 7.5 | 11.25 |
| heavy | 7.5 | 11.25 | 15 | 30 |
| intense | 15 | 30 | 50 | 60 |
| torential | 50 | 60 | 70 | 80 |

### Implementasi Proses *Document Planning*

Berdasarkan penjelasan pada sub-bab model proses *Document* *Planning* yang telah dipaparkan sebelumnya, implementasi pada proses ini berupa *Content Determination* atau pemilihan konten yang akan ditampilkan pada teks keluaran nantinya. Karena banyaknya sinyal dan pesan yang dihasilkan pada proses sebelumnya, maka sinyal-sinyal dan pesan-pesan tersebut perlu diseleksi terlebih dahulu sehingga pesan yang ditampilkan tetap relevan.

Pada pendeteksian sinyal *Repeated Event*, konten yang akan ditampilkan adalah ketika ada nilai perulangan pada suatu parameter. Dapat dilihat pada gambar 4.49, dimana ada proses pengecekan variabel *maxValue*, variabel ini berisikan jumlah perulangan terbesar pada suatu adata, jika variabel *maxValue* ini bernilai 0, maka pesan *Repeated Event* tidak akan dimunculkan, dan diganti dengan pesan default *“There were no repeating values within [limit] [interval] or more, every value changed from time to time.”* yang menandakan bahwa tidak terdapat nilai yang berulang yang melebihi *limit* sebesar 10% dari range data, dengan *interval* data perjam, perhari, perminggu, atau pun yang lainnya. Namun, jika terdapat perulangan pada suatu parameter, akan ditampilkan pesan seperti ini *“There were some repeating value more than 876 hours: IR stayed constant at very low during 1 Jan 2010 00:00 to 4 Mar 2010 15:00, 24 Oct 2010 14:00 to 31 Dec 2010 23:00.”*.

|  |
| --- |
| resumeRepeatedLimit **<-** as.integer**(**nrow**(**dataset**)** **\*** 0.1**)**  #Content Determination  **if(**maxValue **!=** 0**){**  resumeRepeated **<-** ResumeRepeated2**(**columnName**[[**maxIndex**]]**, dataset, vectorRepeatedInterpretResult, listRepeated**[[**maxIndex**]]$**Start, listRepeated**[[**maxIndex**]]$**End**)**  resumeRepeated **<-** paste**(**"There were some repeating value more than @limit @interval: ", resumeRepeated**)**  **}else{**  resumeRepeated **<-** "There were no repeating values within @limit @interval or more, every value changed from time to time."  **}** |

Gambar 4.49 Implementasi *Content Determination* untuk *Repeated Event*

Implementasi *Content Determination* untuk *Extreme Event* dilakukan dengan memilih kenaikan atau penurunan yang mempunyai nilai interpreter *extreme*. Seperti yang digambarkan pada gambar 4.50 dimana jika suatu kenaikan dan penurunan dua-duanya merupakan sinyal *extreme* maka konten *fluctuate extremely* akan ditampilkan. Namun jika hanya kenaikannya saja yang merupakan sinyal *extreme* maka sinyal tersebut akan menghasilkan pesan *increase extremely,* begitu juga jika penurunan suatu parameter merupakan sinyal *extreme* maka sinyal tersebut akan menghasilkan pesan berupa *decrease extremely*. Contoh pesan *extreme* yang akan ditampilkan adalah *“O3 increased significantly from 29-30 Sep 2016 (increased 95 points)”.*

|  |
| --- |
| #Fluctuated Extremely  **if(**dfGrowth**$**IncInterpreter**[**i**]** **==** "extreme" **&&** dfGrowth**$**DecInterpreter**[**i**]** **==** "extreme"**){**  event **<-** "fluctuated"  adverb **<-** "significantly"  phrase **<-** paste**(**event, adverb**)**    sentence **<-** paste0**(**dfGrowth**$**columnNameNumerical**[**i**]**, " ", phrase, " (increased ", dfGrowth**$**IncValue**[**i**]**," points ","and decreased ", abs**(**dfGrowth**$**DecValue**[**i**])**," points)"**)**    flResult**[**flIndex**]** **<-** as.character**(**sentence**)**  flIndex **<-** flIndex **+**1  #Increased Extremely  **}else** **if(**dfGrowth**$**IncInterpreter**[**i**]** **==** "extreme"**){**  event **<-** "increased"  adverb **<-** "significantly"  phrase **<-** paste**(**event, adverb**)**    dateRange **<-** LexicalDateRange**(**as.character**(**dateTime**[**dfGrowth**$**IncStartIndex**[**i**]])**,  as.character**(**dateTime**[**dfGrowth**$**IncEndIndex**[**i**]]))**    sentence **<-** paste0**(**dfGrowth**$**columnNameNumerical**[**i**]**, "  phrase, " from ", dateRange, " ","(increased ", dfGrowth**$**IncValue**[**i**]**," points)" **)**    incResult**[**incIndex**]** **<-** as.character**(**sentence**)**  incIndex **<-** incIndex **+**1  #Increased Extremely  **}else** **if(**dfGrowth**$**DecInterpreter**[**i**]** **==** "extreme"**){**  event **<-** "decreased"  adverb **<-** "significantly"  phrase **<-** paste**(**event, adverb**)**    dateRange **<-** LexicalDateRange**(**as.character**(**dateTime**[**dfGrowth**$**DecStartIndex**[**i**]])**,  as.character**(**dateTime**[**dfGrowth**$**DecEndIndex**[**i**]]))**    sentence **<-** paste0**(**dfGrowth**$**columnNameNumerical**[**i**]**, " ", phrase, " from ", dateRange, " ", "(decreased ", abs**(**dfGrowth**$**DecValue**[**i**])**," points)" **)**    decResult**[**decIndex**]** **<-** as.character**(**sentence**)**  decIndex **<-** decIndex **+**1  **}** |

Gambar 4.50 Implementasi *Content Determination* untuk *Extreme Event*

Dalam pendeteksian sinyal *String Matching*, implementasi *Content Determination* dilakukan pada fungsi MotifDiscoveryDocPlan() dengan melakukan pengecekan hanya pada parameter dengan tipe *categorical*. Jika tidak terdapat pola yang sama pada parameter tersebut, maka variabel *listColumn* pada gambar 4.51 akan bernilai *NA,* dan ditampilkan pesan *“For the past 7 days , no same patterns were found for each categorical parameters”.* Namun jika suatu parameter memiliki pola yang sama, maka variable *listColumn* pada gambar 4.51akan berisikan nilai sesuai indeks dari parameter tersebut. Sehingga akan ditampilkan pesan seperti berikut *“There are a duplicate CBWD data pattern in the last 6 hours (31 Dec 2010 17:00 to 31 Dec 2010 23:00) with data patterns from 3 May 2010 20:00 to 4 May 2010 02:00, and 1 Oct 2010 10:00 to 1 Oct 2010 16:00.”*

|  |
| --- |
| #Content Determination  MotifDiscoveryDocPlan **<-** **function(**listMD**){**  #For categorical parameter only  listCategorical **<-** mainConfig**[**which**(**mainConfig**$**Type **==** **(**"character"**)** **|** mainConfig**$**Type **==** **(**"categorical"**)|** mainConfig**$**Type **==** **(**"factor"**))**,**]**      **if(**nrow**(**listCategorical**)** **!=** 0**){**    listColumn **<-** rownames**(**listCategorical**)**    i**<-**1  **for(**i **in** i**:**length**(**listColumn**)){**  #If there’s no pattern in data, value = NA  **if(**is.na**(**listMD**[[**as.numeric**(**listColumn**[**i**])]]$**pattern**[**1**]))** **{**  listColumn**[**i**]** **<-** **NA**  **}**  **}**  **}else{**  listColumn **<-** **NULL**  **}**    return**(**listColumn**)**  **}** |

Gambar 4.51 Implementasi *Content Determination* untuk pendeteksian *String Matching*

Implementasi *Content Determination* pada korelasi parameter dilakukan pada fungsi CorrelationSignificantMsgContentDetermination() yang dapat dilihat pada gambar 4.52. Implementasi *Content Determination* untuk korelasi parameter terbilang sederhana, dimana nilai yang akan diproses sebagai pesan keluaran adalah nilai korelasi yang lebih besar dari 0.7 untuk korelasi positif, dan nilai korelasi yang lebih kecil dari -0.7 untuk korelasi negatif. Sehingga didapatkan pesan keluaran seperti ini *“An increase in TEMP resulted a growth in DEWP. As a result of the decay in PRES, it can be seen that DEWP, and TEMP is increasing”.*

|  |
| --- |
| CorrelationSignificantMsgContentDetermination **<-** **function(**matrix**){**  # ContentDetermination    # Removing top triangle  matrix**[!**lower.tri**(**matrix**)]** **<-** 0  matrix **<-** as.data.frame**(**matrix**)**  # Only showing var more than 0.7  matrix**[**matrix **<** 0.7 **&** matrix **>=** 0 **|** matrix **>** **-**0.7 **&** matrix **<=** 0**]** **<-** 0    return**(**matrix**)**  **}** |

Gambar 4.52 Implementasi *Content Determination* untuk korelasi antar parameter

Implementasi *Content Determination* pada *Current Text* dilakukan pada fungi yang digambarkan pada gambar 4.53 dimana pesan hanya akan ditampilkan jika suatu parameter mendapatkan nilai tertinggi atau nilai terendah pada data terakhir. Sehingg jika data terakhir merupakan nilai tertinggi atau nilai terendah, maka akan ditampilkan pesan seperti berikut *“LWS reached their highest value on this hour”.*

|  |
| --- |
| CurrentHighest **<-** **function(**data, statSummary, interval**){**  analysisResult **<-** c**()**  analysisIndex **<-** c**()**    i **<-** 1  result **<-** ""  **for(**i **in** i**:**length**(**data**)){**  #get highgest/lowest value from statsum  maxVal **<-** as.numeric**(**as.character**(**statSummary**[**statSummary**$**ColName **==** names**(**data**)[**i**]**,**]$**MaxValue**))**  minVal **<-** as.numeric**(**as.character**(**statSummary**[**statSummary**$**ColName **==** names**(**data**)[**i**]**,**]$**MinValue**))**    sentence **<-** ""  analysisResult**[**i**]** **<-** **NA**  analysisIndex**[**i**]** **<-** **NA**  #if last data is highest data  **if(**data**[**i**]** **>=** maxVal**){**  analysisResult**[**i**]** **<-** "+"  analysisIndex**[**i**]** **<-** i  #if last data is lowest data  **}else** **if(**data**[**i**]** **<=** minVal**){**  analysisResult**[**i**]** **<-** "-"  analysisIndex**[**i**]** **<-** i  **}**  result **<-** paste0**(**result, sentence**)**  **}**    result **<-** CurrentHighestAggregation**(**data, analysisResult, analysisIndex, interval**)**  return**(**result**)**  **}** |

Gambar 4.53 Implementasi *Content Determination* untuk *Current Text*

Implementasi *Document Structuring* dilakukan dalam hal analisa struktur dokumen pada pembahasan sebelumnya, yang selanjutnya struktur dokumen tersebut akan direalisasikan pada sub-bab *Structure Realisation.*

### Implementasi Proses *Microplanning and Realisation*

Pada tahap ini implementasi ini dilakukan pada setiap perbandingan data, contoh implementasi pada tahap *Lexicalisation* adalah ketika menentukan frasa untuk mendefinisikan rentang tanggal. Misalnya, format rentang *“2-3 Dec 2018”* digunakan untuk mendefinisikan rentang tanggal yang memiliki nilai bulan dan tahun yang sama, lalu format *“2 Oct -3 Dec 2018”* digunakan untuk mendefinisikan rentang tanggal dengan tahun yang sama, sedangkan format *“20 Jan 2017 - 08 Jan 2018”* digunakan untuk mendefinisikan rentang tanggal yang tahunnya berbeda. Proses penentuan frasa ini dilakukan pada fungsi LexicalDateRange() yang dapat dilihat pada gambar 4.54.

|  |
| --- |
| LexicalDateRange **<-** **function(**dateStart, dateEnd**){**  #FORMAT INPUT: mm/dd/yyyy -> "07/01/2018"    startMonth **<-** as.numeric**(**substr**(**dateStart,1,2**))**  endMonth **<-** as.numeric**(**substr**(**dateEnd,1,2**))**    startDate **<-** as.numeric**(**substr**(**dateStart,4,5**))**  endDate **<-** as.numeric**(**substr**(**dateEnd,4,5**))**    startYear **<-** as.numeric**(**substr**(**dateStart,7,10**))**  endYear **<-** as.numeric**(**substr**(**dateEnd,7,10**))**  #Exception for hourly data  #Hourly data default: 1 June 2018 00:00 to 1 June 2018 12:00  **if(**datasetIntervalValue **==** 1**){**  result **<-** paste0**(**startDate, " ", month.abb**[**startMonth**]**, " ", startYear, " ", SubstrRight**(**dateStart,5**)**,  " to ", endDate, " ", month.abb**[**endMonth**]**, " ", endYear, " ", SubstrRight**(**dateEnd,5**))**  return**(**result**)**  **}**    timeRepeated **<-** ""  **if(**startYear **==** endYear**){**  **if(**startMonth **==** endMonth**){**  #Example: 1 - 8 June 2018  timeRepeated **<-**paste0**(**startDate, "-", endDate, " ", month.abb**[**startMonth**]**, " ", startYear**)**    #if month different  **}else{**  #Example: 1 Mar - 8 June 2018  timeRepeated **<-**paste**(**startDate, month.abb**[**startMonth**]**, "-", endDate, month.abb**[**endMonth**]**, startYear**)**  **}**  **}else{**  #Example: 20 Jan 2017 - 08 Jan 2018  timeRepeated **<-**paste**(**startDate, month.abb**[**startMonth**]**, startYear, "-", endDate, month.abb**[**endMonth**]**, endYear**)**  **}**    return**(**timeRepeated**)**  **}** |

Gambar 4.54 Implementasi *Lexicalisation* untuk format range tanggal

Implementasi *Aggregation* pada umumnya diterapkan pada proses penggabungan sinyal-sinyal yang memiliki jenis kontras yang sama. Misalnya, proses penggabungan sinyal untuk proses perbandigan parameter pada *Resume Text*, dimana sinyal-sinyal perbandingan tersebut akan dikelompokan menjadi tiga kelompok menggunakan *simple conjunction* seperti *but* dan *and*. Contoh implementasi proses *Aggregation*  ini dapat dilihat pada gambar 4.55.

|  |
| --- |
| ComparsionAggregation **<-** **function(**group**){**  **if(**is.null**(**group**)){**  return**(**group**)**  **}else** **if(**length**(**group**)** **==** 1**){**  return**(**group**)**  **}else{**  i **<-** 1  message **<-** ""  **for(**i **in** i**:**length**(**group**)){**  #aggregation process for last data with same contrast  **if(**i **==** length**(**group**)){**  message **<-** paste0**(**message, "and ", group**[**i**])**  **}else{**  message **<-** paste0**(**message, group**[**i**]**, ", "**)**  **}**  **}**    return**(**message**)**  **}**  **}** |

Gambar 4.55 Implementasi *Aggregation* untuk mengelompokan sinyal   
pada perbandingan data

Selain itu, implementasi proses *Aggregation* ini dilakukan pada proses-proses yang membutuhkan proses *grouping* didalamnya, seperti proses *grouping* pada pesan *trend,* lalu proses *grouping* untuk perbandingan parameter, dan juga proses *grouping* pada *Current text* dan dan *Predict Text*. Ciri khas dari prses *Aggregation*  dengan *Simple Conjunction* ini adalah parameter-parameter yang ditampilkan sekaligus. Contohnya, pesan pada *Current Text* berikut ini*“PM2.5, DEWP, IS, and IR in very low condition. TEMP in low condition. IS, and IR reached their lowest value on this hour.”*. Dimana pada pesan tersebut, parameter-parameter yang memiliki hasil interpretasi *very low* dan parameter-parameter yang merupakan *lowest* value dikelompokan dan ditampilkan sekaligus menggunakan *simple conjunction* berupa *and*.

Sedangkan contoh implementasi *Referring Expression Generation*  terdapat pada gambar 4.56 dimana dilakukan pemillihan *intro* secara *random* berdasarkan *corpus* yang ada.

|  |
| --- |
| ReadIntro **<-** **function(**source**=**"Data", type**=**"General"**){**  type  **...**  as.matrix**(**read.table**(**file**=**paste0**(**"Corpus/",type,"Intro.csv"**)**, header**=FALSE**, sep**=**';'**))**  # print(corpus)  n **<-** length**(**corpus**)**  random\_value **<-** as.integer**(**runif**(**1,1,n**+**0.5**))**  ...  # return("Woops no data intro!");  **}**  ReadResumeIntro **<-** **function(**dataset2, ColName, source**=**"dataset2"**){**    corpus **<-** as.matrix**(**read.table**(**file**=**paste0**(**"Corpus/", "ResumeIntro.csv"**)**, header**=FALSE**, sep**=**';'**))**    #Randoming corpus  n **<-** length**(**corpus**)**  random\_value **<-** as.integer**(**runif**(**1,1,n**))**  result **<-** corpus**[**random\_value**]**  .....  return **(**result**)**  **}** |

Gambar 4. 56 Implementasi *Referring Expression Generation* untuk  
menentukan intro secara *random*

Selain itu, penentuan interval waktu data dalam bahasa manusia terdapat pada gambar 4.57 dimana pertama-tama diambil sampel data yang kemudian ditentukan interval datanya dalam jam, yang kemudian direpresentasikan kedalam bentuk manusia menggunakan fungsi DataInterpreterInterval() yang terdapat pada gambar 4.57.

|  |
| --- |
| DataInterpreterInterval **<-** **function** **(**interval, type **=** "default"**){**  result**<-**""    #Interval data  **if(**type **==** "interval"**){**  **if(**interval **==** 1**){**  result **<-** "hourly"  **}else** **if(**interval **==** 24**){**  result **<-** "daily"  **}else** **if(**interval **==** 168**){**  result **<-** "weekly"  **}else** **if(**interval **==** 672 **||** interval **==** 696 **||** interval **==** 720 **||** interval **==** 744**){**  result **<-** "monthly"  **}else** **if(**interval **==** 8760 **||** interval **==** 8736**){**  result **<-** "yearly"  **}else{**  result **<-** ""  **}**  #Intro with interval data  **}else** **if(**type **==** "intro"**){**  **if(**interval **==** 1**){**  result **<-** "This hour"  **}else** **if(**interval **==** 24**){**  result **<-** "Today"  **}else** **if(**interval **==** 168**){**  result **<-** "This week"  **}else** **if(**interval **==** 720 **||** interval **==** 744**){**  result **<-** "This month"  **}else** **if(**interval **==** 8760 **||** interval **==** 8736**){**  result **<-** "This year"  **}else{**  result **<-** ""  **}**  **}**  **....**  return**(**result**)**  **}** |

Gambar 4. 57 Implementasi *Referring Expression Generation*   
untuk *Time Descritption*

Sedangkan implementasi *Structure Realisation* dilakukan seperti pada gambar 4.58

|  |
| --- |
| #Resume Text  resumeResult **<-** paste**(**resumeIntro, resumeTrend, resumeComparsion, resumeRepeated, resumeExtremeEvent, resumeMotifDiscovery, resumeCorrelation**)**  #Current Text  currentResult **<-** paste**(**currentIntro, currentDesc, currentHighest**)**  #Predict Text  predictResult **<-** paste**(**predictIntro, predictContent**)** |

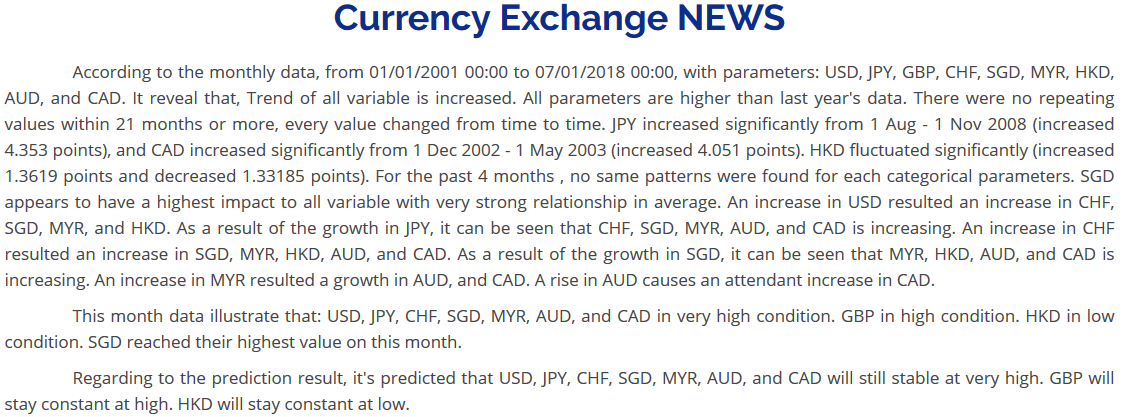
Gambar 4. 58 Implementasi *Structure Realisation*

## Hasil Implementasi

Setelah melalui berbagai proses pada sub-bab sebelumnya, maka hasil dari sistem D2T yang dibangun berupa teks keluaran yang ditampilkan ke dalam platform web dengan menggunakan *package* ShinyR. Teks keluaran yang dihasilkan terdiri dari tiga paragraf, paragraf pertama berisikan ringkasan data, dan *event* atau kejadian-kejadian yang terjadi, paragraf ke-dua menjelaskan kondisi terkini, dan paragraf ke-tiga berisikan prediksi untuk data yang akan datang mengugnakan *exponential smoothing*. Pada sub-bab ini akan dipaparkan mengenai hasil keluaran sistem, perbandingan hasil keluaran dengan penelitian terkain, dan validasi keluaran sistem.

### *Output* Sistem

Sistem D2T yang dikembangkan pada penelitian ini diharapkan tetap mampu memproses data dengan jenis apapun, sehingga sistem akan tetap berjalan meskipun data masukannya belum didefinisikan sebelumnya. Selain itu, sistem yang D2T dikembangkan mampu menghasilkan berita dengan *domain* umum atau *general*. Hasil keluaran dari serangkaian proses pada sistem untuk contoh kasus data nilai tukar mata uang asing pada tabel 4.1 dapat dilihat pada gambar 4.59.



Gambar 4.59 Hasil keluaran dari sistem untuk contoh kasus data nilai tukar

Dari berita pada gambar 4.59 dijelaskan bahwa berdasarkan pada dataset nilai tukar yang terdapat pada tabel 4.1 dengan interval data perbulan, *trend* pada semua parameter adalah menaik. Kata *“According to the mothly data”* dan *“This month data illustrate that”* pada awal paragraf pertama dan kedua merupakan implementasi dari proses *Referring Expression Generation* pada tahap *Microplanning,* dimana interval dari data akan didefinisikan terlebih dahulu lalu dilakukan pemilihan intro pada *corpus* secara random. Dikarenakan rentang data masukan adalah bulanan maka dilakukan perbandingan data terakhir dengan data 1 tahun terakhir, dimana semua data menunjukan kenaikan sehingga dihasilkan pesan *“All parameters are higher than last year's data.”*. Dikarenakan pada setiap parameter tidak memiliki nilai yang berulang melebihi 10% data atau 21 kali perulangan, maka ditampilkan *“There were no repeating values within 21 months or more, every value changed from time to time.”* yang merupakan hasil dari tahap *Content Determination* jika tidak ditemukan nilai yang berulang pada setiap parameter. Selain itu, pemilihan frasa untuk menentukan format range tanggal seperti *“from 1 Aug - 1 Nov 2008”* pada kasus *Extreme* *Event* merupakan hasil dari proses *Lexicalisation*. Proses *Lexicalisation* ini juga diimplementasikan pada paragraf ke-tiga dimana frasa seperti *“will still stable”*, *“will stay constant”* dihasilkan dari penentuan trend dari beberapa data terakhir, proses ini dijalankan pada fungsi *Trend Description*. Hubungan parameter didapatkan dari proses signal analysis untuk korelasi parameter, dimana pesan pesan yang disebutkan adalah pesan yang memiliki nilai korelasi lebih dari 0.7 baik untuk nilai positif maupun negatif.

Pengelompokan parameter pada paragraf ke-dua dan ke-tiga dihasilkan melalui proses *Aggregation* dimana parameter yang memiliki sinyal dengan kontras yang sama akan dimunculkan bersamaan, dalam kasus ini terdapat beberapa parameter dengan kontras yang sama yakni USD, JPY, CHF, SGD, MYR, AUD, dan CAD dengan persamaan kontras sinyal *Trend Description* yang menunjukakan *“will still stable”* dan juga hasil prediksi yang menunjukan parameter tersebut akan tetap berada pada kondisi *Very High.* Frasa seperti, *very high, high, low, very low* didapatkan dari hasil fungsi keanggotaan untuk data *general* yang kemudian interpretasikan menggunakan *Fuzzy Membership Function* sehingga didapatkanlah frasa tersebut.

### Perbandingan *Output* Sistem dengan Penelitian Terkait

Dalam proses perbandingan ini, digunakan data klimatologi dari stasiun MeteoGalicia yang ada pada tabel 4.2, dimana hasil dari penelitian ini akan dibandingkan dengan hasil dari beberapa penelitan sebelumnya khusunya pada bidang klimatologi, hasil *output* sistem akan dibandingkan dengan hasil dari penelitian DWP (Putra et al., 2017), penelitian oleh Ramos-soto (2015), penelitian Gkatzia et al., (2016), dan penelitian yang dilakukan oleh (Kittredge & Driedger, 1994). Perbandingan *output* dapat dilihat pada tabel 4.44.

Tabel 4.44 Perbandingan *output* dengan penelitian terkait

| Penelitian | Output |
| --- | --- |
| (Kittredge & Driedger, 1994)  *Output* | Winds northwest 15 diminishingto light monday afternoon. Cloudy with occasional light snow. Fog patches. Visibilities 2 to 5 nm in snow. Belle isle. Northeast gulf northeast coast. Gale warning in belle isle and northeast gulf issued. Gale warning in northeast coast continued. Freezing spray warning continued. Winds southwest 15 to 20 knots increasing to west gales 35 |
| (Ramos-Soto et al., 2015) *Output* | With respect to the air quality state, it will be variable although is expected to improve to good, favored by the wind during the coming days |
| (Gkatzia et al., 2016)  *Output* | -Light rian showers are likely  -Sunny intervals with rain being possible – less likely than not.  -Sunny with rain being unlikely |
| DWP (Putra, 2017) *Output* | Regarding to the prediction result, tomorrow sky state will be light rain although its covered by partly cloudy sky. Followed by temperature which decreased to warm. According to the air quality state, it will start to change to good.  According to the monthly summary result, this month was cooler and wetter than average. With average number of rain days, accordingly the total rain so far is well below the average. There was rain on everyday for 7 days from 02nd to 08th and intense rain was dropped in 06th. The wind for the month was light breeze in average. Average air quality was admissible. Average temperature was increased but 05 th was the coldest day of the month with 13.3 celcius degree temperature. |
| GNG Output  #*With*  *Header* | Regarding to the daily data, from 07/06/2016 00:00 to 07/06/2017 00:00, with parameters: Cloud Coverage, Temperature, Wind Speed, Wind Direction, and Rainfall. It represented that, Temperature trend is decreased and Rainfall trend is constant but the rest is increased. Temperature, and Wind Direction parameters are higher than last week's data, but the rest parameters are lower than last week's data. There were some repeating value more than 36 days: Rainfall stayed constant at no rain during 6 Jul - 18 Aug 2016. Cloud Coverage fluctuated significantly (increased 90.9 points and decreased 87.6 points), Wind Speed fluctuated significantly (increased 20.05 points and decreased 27.03 points), Wind Direction fluctuated significantly (increased 270 points and decreased 270 points), and Rainfall fluctuated significantly (increased 67.2 points and decreased 64.1 points). For the past 7 days , no equivalent patterns were found for each categorical parameters. Rainfall appears to have a highest impact to all variable with moderate relationship in average.  Today data represent that: Cloud Coverage in mostly cloudy condition. Temperature in warm condition. Wind Speed in light Breeze condition. Wind Direction in North West condition. Rainfall in no rain condition. Rainfall reached their lowest value on this day. Wind Direction reached their highest value on this day.  From the prediction result, it's expected that tomorrow sky will be light rain although it's covered by partly cloudy sky. Favored by temperature which decreased to warm. Cloud Coverage will change progressively to partly cloudy. Temperature will still stable at warm. Wind Speed will stay constant at light Breeze. Wind Direction will moderately turn to West. Rainfall will shifted to light rain. |
| GNG Output  #*Without*  *Header* | According to the daily data (07/06/2016 00:00 - 07/06/2017 00:00), with parameters: V2, V3, V4, V5, and V6. It is clear that, V3 trend is decreased and V6 trend is constant but the rest is increased. V3, and V5 parameters are higher than last week's data, but the rest parameters are lower than last week's data. There were some repeating value more than 36 days: V6 stayed constant at very low during 6 Jul - 18 Aug 2016. V2 fluctuated significantly (increased 90.9 points and decreased 87.6 points), V4 fluctuated significantly (increased 20.05 points and decreased 27.03 points), V5 fluctuated significantly (increased 270 points and decreased 270 points), and V6 fluctuated significantly (increased 67.2 points and decreased 64.1 points). For the past 7 days , no equivalent patterns were found for each categorical parameters. V6 appears to have a highest impact to all variable with moderate relationship in average.  Today data indicate that: V2 in medium condition. V3 in high condition. V4 in low condition. V5 in very high condition. V6 in very low condition. V6 reached their lowest value on this day. V5 reached their highest value on this day.  According to the prediction result, it's predicted that V2 will move to medium. V3 will still stable at high. V4 will still stable at low. V5 will begin to turn to very high. V6 will stay constant at very low. |

Berdasarkan hasil pada tabel 4.44 dapat disimpulkan bahwa *output* dan konten-konten yang dihasilkan dari penelitian ini lebih banyak dibanding dengan yang lainnya, namun jika dilihat dari penyampaian konten pada teks keluaran, sistem DWP lebih unggul, dikarenakan sistem DWP dikembangkan untuk membangkitkan berita dengan domain cuaca (Putra et al., 2017). Pada penelitian DWP, hasil keluaran dibagi menjadi dua paragraf, dimana paragraf pertama bersikan prediksi cuaca, dan paragraf kedua berisi tentang rangkuman data selama satu bulan. Pada sistem DWP terdapat dua data masukan yaitu data klimatologi dan kualitas udara, sehingga konten yang muncul terbagi ke dalam dua bagian. Sedangkan pada penelitian Ramos (2016), hanya ada dua konten yang ditampilkan yaitu konten kualitas udara dan kecepatan angin. Jika pada penelitian Kittredge dan Driedger (1994) dihasilkan konten mengenai kondisi angin dan salju secara terpisah-pisah, sedangkan pada penelitian Gkatzia (2016) konten yang disampaikan hanya terkait keadaan langit saja. Hal ini menunjukan bahwa sistem D2T yang dibangun sebelumnya terikat pada satu domain spesifik saja. Berbeda dengan sistem D2T yang dikembangkan pada penelitan ini, dimana sistem akan tetap menghasilkan meskipun yang menjadi masukan adalah data *general* ataupun data yang tidak memiliki header. Dimana data-data tersebut akan diberi header *default* yang secara otomatis dihasilkan oleh sistem dan lalu diinterpretasikan nilainya menggunakan fungsi keanggotaan secara *general*.

## Panduan Penggunaan Aplikasi

Berikut tahapan untuk menggukanan aplikasi D2T dalam penelitian ini:

* + - * 1. *Extract* semua file D2T.rar kedalam sebuah folder
        2. Buka file D2T\_Machine.R menggunakan RStudio.
        3. Install *package* yang diperlukan
        4. Buka file apps.R
        5. Lalu tekan tombol RunApp pada RStudio
        6. Jendela browser akan terbuka, lalu masukkan judul berita dan tekan tombol enter, tunggu sebentar hingga pesan keluaran muncul. Direkomendasikan untuk menggunakan browser eksternal seperti *Firefox* maupu *Goole Chrome* agar aplikasi berjalan optimal dan responsif.