BAB II

TINJAUAN PUSTAKA

2.1 Peramalan Beban Listrik

Metode peramalan merupakan cara memperkirakan apa yang akan terjadi pada masa depan secara sistematis dan pragmatis atas dasar data yang relevan pada masa yang lalu, sehingga dengan demikian metode peramalan diharapkan dapat memberikan objektivitas yang lebih besar. Metode peramalan dapat memberikan cara pengerjaan yang teratur dan terarah, dengan demikian dapat dimungkinkannya penggunaan teknik penganalisaan yang lebih maju. Penggunaan teknik-teknik tersebut maka diharapkan dapat memberikan tingkat kepercayaan dan keyakinan yang lebih besar karena dapat diuji penyimpangan atau *deviasi* yang terjadi secara ilmiah [16]. Peramalan dapat dibagi menjadi tiga kategori, berdasarkan jangka waktunya jangka pendek, jangka menengah, dan jangka panjang.

Beban listrik merupakan pemakaian tenaga listrik dari para pelanggan. Besar kecilnya beban listrik beserta perubahannya tergantung kepada kebutuhan para pelanggan listrik pada suatu saat tertentu. Peramalan beban listrik menjamin kualitas pelayanan para pelanggan tersebut. Peramalan di bidang tenaga listrik dimaksudkan pada perkiraan kebutuhan beban listrik di masa yang akan datang. Beban yang diramalkan mempunyai jangka waktu tertentu yang disesuaikan dengan kebutuhan peramalan. Peramalan beban jangka pendek untuk meramalkan kebutuhan beban harian umumnya tersaji pada beban listrik bertujuan untuk mengenali pola beban dengan mengolah data historis beban listrik yang direpresentasikan dalam kurva beban harian. Model peramalan beban yang akurat sangat penting dalam perencanaan dan pengoperasian sistem tenaga listrik. Pemodelan yang sesuai berdasarkan pada pola beban yang dilakukan sehingga diperoleh parameter-parameter yang diperlukan [8].

Peramalan dibagi menjadi beberapa kategori jangka waktu yaitu jangka pendek, jangka menengah, dan jangka panjang sebagi berikut [7]:

1. Peramalan jangka pendek (*short term forecasting*), yaitu peramalan penggunaan daya pada beban dengan jangka waktu beberapa jam sampai dengan 1 minggu ke depan.
2. Peramalan jangka menengah (*mid term forecasting*), yaitu peramalan penggunaan daya beban dengan jangka waktu 1 bulan sampai 1 tahun ke depan.
3. Peramalan jangka panjang (*long term forecasting*), yaitu peramalan penggunaan daya beban untuk jangka waktu di atas 1 tahun.

2.2 JST (Jaringan Syaraf Tiruan)

JST (Jaringan Syaraf Tiruan) merupakan sistem komputasi dengan arsitektur dan operasi yang diilhami dari pengetahuan tentang sel syaraf di dalam otak manausia. Prinsip kerja tersebut menjadikan JST sangat sesuai untuk menyelesaikan berbagai masalah yang mempunyai tipe sama seperti otak manusia dalam menyelesaikan masalah. Keberhasilan penggunaan metode JST ditentukan oleh tiga faktor, yaitu [12]:

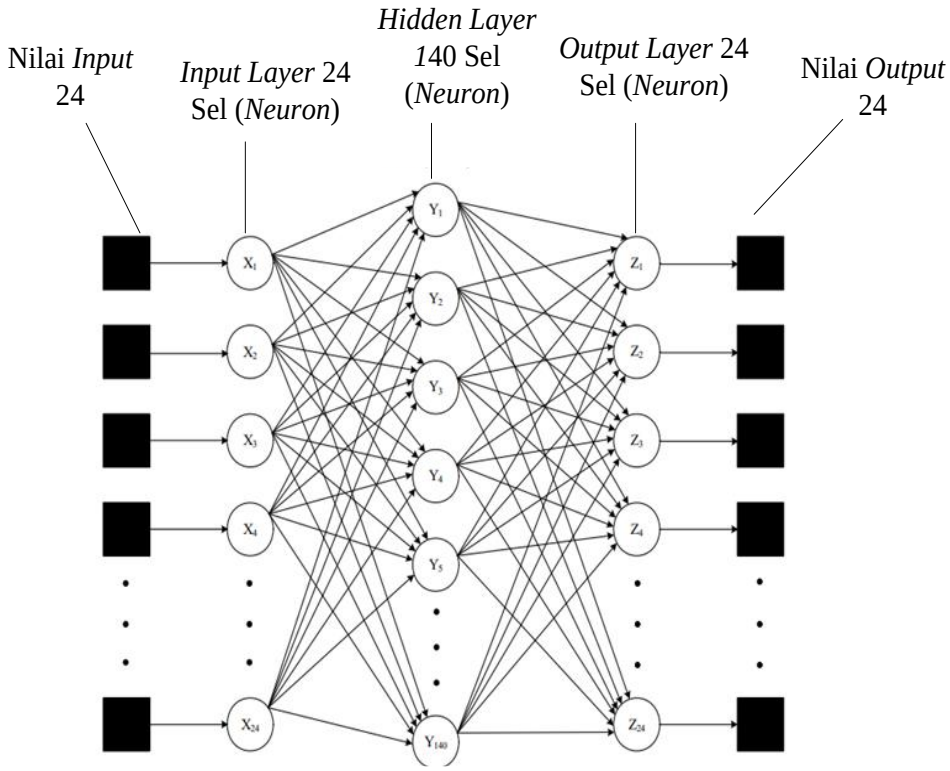
1. Pola-pola hubungan antar *neuron* yang disebut arsitekstur jaringan.
2. Metode penentuan bobot penghubung yang disebut metode pembelajaran (*training* atau *learning*).
3. Fungsi aktifasi yang digunakan.

JST mampu mengenali kegiatan dengan berbasis masa lalu. Data masa lalu dipelajari oleh JST sehingga mempunyai kemampuan untuk memberi keputusan terhadap data yang belum pernah dipelajari [17].

2.3 Arsitektur JST

Berdasarkan jumah lapis (l*ayer*), arsitektur JST dapat mengklasifikasikan menjadi dua kategori yaitu jaringan lapis tunggal (*single layer network*), dimana semua unit *input* dalam jaringan ini dihubungkan dengan semua unit *output*, meskipun dengan bobot yang berbeda-beda dan jaringan lapis jamak (*multi layer*

network) yang merupakan perluasan dari *single layer*. Arsitektur JST dapat dilihat pada Gambar 2.1.

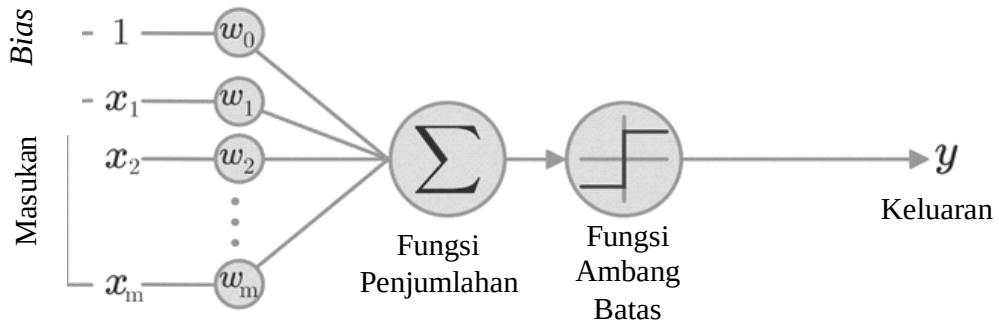


Gambar 2.1 Arsitektur JST [18]

Gambar 2.1 merupakan arsitektur JST jaringan layar jamak, memperkenalkan satu atau lebih *layer* tersembunyi (*hidden layer*) yang mempunyai simpul yang disebut *neuron* tersembunyi. Arah aliran sinyal masukan, arsitektur JST dapat menglasifikasikan menjadi dua kelas yang berbeda, yaitu jaringan umpan maju (*feedforward network*) dan jaringan dengan umpan balik (*recurrent network*) [12].

2.4 Model JST

Model JST memiliki banyak bagian. Bagian-bagian pada model JST dapat dilihat pada Gambar 2.2.



Gambar 2.2 Model JST

Gambar 2.2 merupakan model JST. Satu sel syaraf terdiri dari 3 bagian yaitu, fungsi penjumlahan (*summing function*), fungsi aktifasi (*activation function*), dan keluaran (*output*) [17].

2.5 Regresi Linier

Analisis regresi adalah studi mengenai ketergantungan variabel *dependent* (terikat) dengan satu atau lebih *variabel independent* dengan tujuan untuk mengestimasi atau memprediksi rata-rata populasi atau nilai rata-rata variabel *dependent* berdasarkan nilai *variabel independent* yang diketahui [19]. Analisis regresi dapat dibedakan menjadi dua berdasarkan banyak dan jenis data [20]:

1. Regresi linier, yaitu regresi yang membuat diagram pencar membentuk garis lurus. Regresi linier terdiri atas regresi linier sederhana (1 variabel bebas) dan regresi linier berganda (lebih dari 1 variabel bebas).
2. Regresi tidak linier, regresi yang membuat *plot* diagram tidak membentuk garis lurus tetapi membentuk pola tertentu, meliputi parabolik, eksponen, geometrik, logistik, dan hiperbolik.

Persamaan dari regresi linier sederhana dinyatakan seperti persamaan yang ada pada persamaan (2.1).

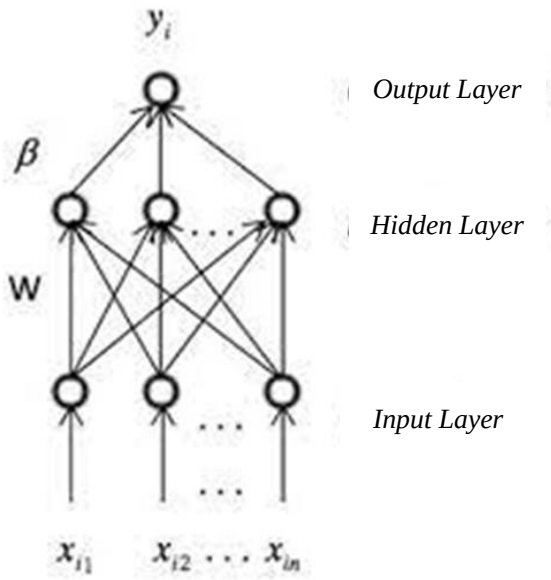
$$Y = a + bX \tag{2.1}$$

Persamaan (2.1) merupakan regresi linier sederhana, dimana Y adalah variabel akibat (*dependent*), a adalah konstanta, b adalah besaran respon yang ditimbulkan oleh *predictor*, dan X adalah variabel penyebab (*independent*).

2.6 ELM (*Extreme Learning Machine*)

ELM (*Extreme Learning* Machine) merupakan JST *feed-forward* dengan satu *hidden layer* atau lebih dikenal dengan istilah SLFN (*Single Layer Feedforward Neural*). Metode ELM mempunyai kelebihan dalam *learning speed*, serta mempunyai tingkat akurasi yang lebih baik dibandingkan dibandingkan dengan metode konvensional seperti m*oving average* dan *exponential smoothing* [21].

Jaringan *feed-forward* menggunakan parameter-parameter yang ditentukan secara manual seperti *input weight* dan *bias*. *Input weight* dan *bias* ini dibangkitkan secara acak dalam suatu rentang tertentu. Nilai yang diacak tersebut, bisa menghindari hasil prediksi yang tidak stabil. Struktur metode ELM secara umum dapat dilihat pada Gambar 2.3



Gambar 2.2 Struktur Metode ELM

Gambar 2.3 menampilkan langkah-langkah perhitungan dengan metode ELM yaitu adalah normalisasi data kemudian data diproses pada langkah proses *training* dan proses *testing*.

2.7 Normalisasi Data

Normalisasi data dilakukan karena *range* nilai *input* tidak sama, yaitu bernilai puluhan hingga ribuan. *Input* diproses ke nilai *output* yang kecil sehingga data yang

digunakan harus disesuaikan agar dapat diproses untuk mendapatkan nilai normalisasi yang kecil. Proses normalisasi data menggunakan metode *min-max Normalization* dapat dilihat pada persamaan (2.2).

$$d' = \frac{d-min}{max-min} \quad (2.2)$$

Persamaan (2.2) merupakan proses normalisasi data menggunakan *min-max normalization*, dimana d' adalah nilai dari hasil normalisasi data, d adalah nilai asli data, min adalah nilai minimum pada data *set* fitur X, dan max adalah nilai maksimal pada data *set* fitur X.

2.8 Proses *Training*

Proses *training* harus dilalui sebelum melakukan proses prediksi. Tujuannya adalah untuk mendapatkan nilai *output weight.* Langkah-langkah proses *training* yaitu sebagai berikut [22]:

1. Langkah pertama adalah menginisialisasi *input weight* dan *bias*. Nilai ini diinisialisasi secara acak dengan *range* antara -1 hingga 1.
2. Keluaran di *hidden layer* dihitung menggunakan fungsi aktifasi. Langkah pertama adalah menghitung keluaran *hidden layer* (H_{init}), setelah nilai H_{init} didapatkan kemudian dihitung menggunakan fungsi aktifasi *sigmoid.* Persamaan untuk menghitung keluaran di *hidden layer* dapat dilihat pada persamaan (2.3).

   $$H_{init\ i\ j} = (\sum_{k=1}^{n} w_{jk}.x_{ik}) + b_j \quad (2.3)$$

   Persamaan (2.3) merupakan persamaan untuk menghitung keluaran *hidden layer*, dimana H_{init} adalah matrik keluaran *hidden layer*, i adalah [1,2,…N], dimana N adalah keseluruhan jumlah data, j adalah [1,2,…N], dimana N adalah keseluruhan jumlah *hidden neuron*, n adalah jumlah *input neuron*, w adalah bobot *input*, x adalah *input* data yang digunakan, dan b adalah nilai *bias*.
3. Menghitung *output weight* dapat dilakukan dengan cara, langkah pertama yang harus dilakukan adalah melakukan *transpose* matrik hasil keluaran *hidden layer* dengan fungsi aktifasi. Setelah dilakukan t*ranspose*, matrik *transpose* tersebut dikalikan dengan matrik hasil keluaran *hidden layer*

dengan fungsi aktifasi *bias*a disebut matrik H. Langkah selanjutnya adalah menghitung nilai invers dari matrik H tersebut. Setelah itu menghitung matrik *Moore-Penrose Generalized Invers* dari hasil keluaran *hidden layer* dengan fungsi aktifasi. Persamaan untuk menghitung nilai *output weight* dapat dilihat pada persamaan (2.4).

$$\beta = H^{+}T \tag{2.4}$$

Persamaan (2.4) merupakan persamaan untuk menghitung nilai *output weight,* dimana β adalah matrik *output weight*, H^{+} adalah matrik *Moore-Penrose Generalized Invers* dari matrik H, dan T adalah matrik target.

2.9 Proses *Testing*

Proses ini bertujuan untuk mengevaluasi metode ELM dari hasil proses *training* sebelumnya. Proses *testing* digunakan menggunakan *input weight*, *bias* dan *output weight* yang didapatkan dari proses *training*. Langkah-langkah proses *testing* adalah sebagai berikut [22]:

1. Langkah pertama adalah menginisialisasi *input weight* dan *bias* yang telah didapatkan dari proses *training*.
2. Keluaran di *hidden layer* dihitung menggunakan fungsi aktifasi. Pilih salah satu fungsi aktifasi yang digunakan yaitu fungsi aktifasi *sigmoid, sin, hardlim.*
3. Nilai *output weight* yang telah didapatkan pada proses *training* digunakan untuk menghitung keluaran *output layer* yang merupakan hasil prediksi. Persamaan untuk menghitung nilai *output layer* dapat dilihat pada persamaan (2.5).

   $$y = H\beta \tag{2.5}$$

   Persamaan (2.5) merupakan persamaan untuk menghitung nilai *output layer*, dimana y adalah *output layer* yang merupakan hasil prediksi, H adalah keluaran di *hidden layer* dihitung dengan fungsi aktifasi, dan β nilai *output weight* didapatkan dari proses *training*.
4. Langkah terakhir adalah menghitung nilai *error* semua *output layer*. Nilai *error* ini menunjukkan nilai kesalahan dari hasil prediksi yang didapatkan.

2.10 Proses Denormalisasi Data

Proses ini berfungsi untuk membangkitkan nilai yang telah dinormalisasi menjadi nilai asli. Persamaan untuk denormalisasi data dapat dilihat pada persamaan (2.6).

$$d = d'(max - min) + min \tag{2.6}$$

Persamaan (2.6) merupakan persamaan untuk denormalisasi data, dimana d' adalah nilai hasil prediksi sebelum di denormalisasi, d adalah nilai hasil asli setelah di denormalisasi, min adalah nilai minimum pada data *set* fitur X dan max adalah nilai maksimum pada data *set* fitur X.

2.11 MSE (*Mean Square Error*)

MSE (*Mean Square Error*) digunakan untuk mengevaluasi hasil prediksi [22]. Persamaan untuk menghitung nilai *error* pada hasil prediksi dapat dilihat pada persamaan (2.7).

$$MSE = \frac{\sum_{i=1}^{n} e_i^2}{n} = \frac{\sum_{i=1}^{n} (y_i - t_i)^2}{n} \tag{2.7}$$

Persamaan (2.7) merupakan persamaan untuk menghitung nilai *error* pada hasil prediksi, dimana n adalah jumlah data, e adalah *error*, y_i adalah nilai *output* (prediksi), dan t_i adalah nilai aktual.

2.12 OPELM (*Optimally Pruned Extreme Learning Machine*)

OPELM (*Optimally Pruned Extreme Learning Machine*) merupakan sebuah metode yang berdasarkan pada ELM aslinya. OPELM ditunjukkan untuk mengatasi kelemahan yang ada pada ELM ketika terdapat variabel yang tidak relevan atau tidak berkorelasi. Alasan tersebut, diperkenalkan metode OPELM untuk pemangkasan variabel yang tidak relevan dengan memangkas *neuron* tidak penting dari SLFN (*Single Hidden Layer Feedforwar Neural* Network) yang dibangun oleh ELM. Langkah pertama dari metode OPELM adalah membangun struktur SLFN menggunakan algoritma ELM. Perangkingan *neuron* dilakukan pada lapisan tersembunyi dengan algoritma MRSR (*Multi Response Spare Regression*) dan akhirnya penentu banyaknya *neuron* yang dipangkas dibuat berdasarkan metode estimasi *error* LOO (*Leave One Out*). Algoritma OPELM menggunakan kombinasi

tiga jenis *kerne*l, *linear, sigmoid,* dan *gaussian*. ELM hanya menggunakan satu *kernel* saja, misalnya *sigmoid* [8].

2.13 Pembentukan SLFN dengan ELM dan Perangkingan *Hidden Neuron* dengan MRSR

Langkah ini dilakukan menggunakan algoritma standar ELM dengan sejumlah *neuron* yang cukup besar. ELM aslinya menggunakan *kernel sigmoid* dan model ELM ini *bias*anya dirumuskan berdasarkan satu jenis fungsi aktifasi atau *kernel* saja. Metode OPELM digunakan kombinasi dari tiga jenis *kernel* yaitu, *linear, sigmoid* dan *gaussian*. *Input weight* dan *bias* ditentukan secara acak. Matrik *output hidden layer* didapat dari kombinasi inisialisasi parameter secara acak dari ketiga fungsi tersebut.

Langkah kedua dalam metode OPELM, MRSR diterapkan untuk perangkingan *hidden neuron* berdasarkan keakurasiannya. Ide utama dari algoritma ini adalah menambahkan setiap kolom dari matrik *regressor* satu per satu ke dalam model $Y^k = X\,W^k$, dimana $Y^k = \left[y_1^k \ldots y_p^k\right]$ adalah pendekatan target model T, dimana $X = [x_{1\ldots\ldots}x_m]$ merupakan $n\,x\,m$ matrik *regressor*,$T = \left[t_{1\ldots\ldots}t_p\right]$ $n\,x\,p$ matrik target dan W^k *weight matrix* memiliki k baris tidak nol dan sebuah kolom baru pada matrik *regressor* ditambahkan ke model [8].

2.14 Seleksi *Neuron* dengan LOO

Perangkingan *neuron* dari *hidden layer* telah diperoleh jumlah *neuron* terbaik untuk model yang dipilih, digunakan LOO untuk memvalidasi. Menghitung LOO *error* bisa sangat memakan waktu ketika kumpulan data cenderung memiliki sampel *neuron* penting. PRESS (*PREdiction Sum of Squares*) memberikan formula langsung dan tepat untuk perhitungan kesalahan ini pada persamaan model linier [8]. Persamaan model linier dapat dilihat pada persamaan (2.8).

$$\mathcal{E}^{\text{PRESS}} = \frac{y_i - h_i b}{1 - h_i P h_i^t} \tag{2.8}$$

Persamaan (2.8) merupakan perhitungan kesalahan pada mode linier, dimana i dinotasikan sebagai *hidden node* ke-i, P di definisikan sebagai $P = (H^T H)^{-1}$, H adalah *hidden layer output matrix* yang didefinisikan sebelumnya, dan h_i adalah

kolom pada keluaran matrik lapisan tersembunyi. Jumlah *neuron* yang optimal didapat dari estimasi LOO *error* pada jumlah *node-node* (yang telah diranking berdasarkan akurasi) dan menyeleksi jumlah *neuron* dari minimum *error* nya. *Neuron* hasil pemangkasan tersebut digunakan untuk menghitung *output weight* yang didapat dari hasil *invers* dari matrik *hidden layer* dan target *output*. Mengetahui performa algoritma dari tiap metode maka dihitung nilai *fitness* yang menunjukkan MAPE (*Mean Absolute Percent Error*) dari tiap perhitungan fungsi objektif *training* dan *testing*, yaitu *error* rata-rata dari pemodelan OPELM dan ELM [8]. Persamaan untuk mengetahui performa algoritma dapat dilihat pada persamaan (2.9).

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{\text{Y}_{\text{prediksi}}-\text{Y}_{\text{target}}}{\text{Y}_{\text{target}}}\right|.100\% \qquad (2.9)$$

Persamaan (2.9) merupakan persamaan untuk mengetahui performa algoritma, dimana $\text{Y}_{\text{prediksi}}$ adalah nilai prediksi dan Y_{target} adalah nilai target. Nilai MAPE makin mendekati nol, maka kinerja hasil peramalan semakin baik [8].

2.15 Kajian Pustaka

Penelitian tentang Peramalan Beban Listrik Jangka Pendek Menggunakan OPELM pada Sistem Kelistrikan Cilegon, Banten sudah dilakukan sebelumnya pada penelitian-penelitian serupa. Kajian pustaka yang berkaitan pada beberapa referensi sebagai acuan atau penunjang pada peneletian yang dilakukan berikut:

1. Penelitian pertama membahas tentang perbandingan peramalan beban listrik menggunakan OPELM dengan ELM. Hasil yang didapatkan adalah metode OPELM lebih akurat dibandingkan dengan metode ELM [8].
2. Penelitian kedua membahas tentang peramalan beban listrik harian menggunakan JST. Hasil yang didapatkan dari metode tersebut adalah MAPE (*Mean Absolute Percent Error*) terkecil adalah 0%, sedangkan MAPE terbesar adalah 28,71%, sehingga rata-rata MAPE pada penelitian menggunakan JST ini adalah 7,233% [12].
3. Penelitian ketiga membahas tentang peramalan beban listrik jangka pendek menggunakan metode AR (*Autoregressive*). Model matematis yang

didapatkan dari berbagai uji yang telah dilakukan dapat dilihat pada persamaan (2.6).

$$y_t = 502{,}452 + 0{,}7268 y_{t-1} + a_t \quad (2.6)$$

Persamaan (2.6) merupakan model matematis yang dapat digunakan untuk peramalan satu minggu berikutnya di tahun tersebut [11].

4. Penelitian keempat membahas tentang pengaplikasian logika *fuzzy* untuk peramalan beban listrik jangka pendek. Hasil yang didapatkan dari penelitian ini adalah model logika *fuzzy* yang dikembangkan untuk peramalan beban listrik jangka pendek ini yaitu terdiri dari 2 *input*,yaitu beban historis dan suhu. Nilai persentase kesalahan logika *fuzzy* berkisar antara 10,78% sampai dengan 16,98% [9].
5. Penelitian kelima membahas tentang prediksi penjualan mi menggunakan ELM. Hasil yang didapatkan berdasarkan pengujian yang telah dilakukan jumlah *neuron* berpengaruh pada hasil perhitungan. Hal ini terbukti dengan menghasilkan nilai *error* yang kecil sebesar 0,0176% dengan jumlah *neuron* sebanyak 7 [22].