BAB II
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.2.1 Konsep Okupansi Hotel

Okupansi hotel merupakan indikator kinerja utama yang menunjukkan tingkat pemanfaatan kapasitas kamar dan tempat tidur yang tersedia. Menurut Ivanov (2014), tingkat okupansi (*occupancy rate*) mengukur pemanfaatan kapasitas fisik hotel dan dapat dihitung menggunakan dua pendekatan, yaitu berdasarkan overnights dan *roomnights*, untuk memberikan gambaran aktivitas operasional hotel yang lebih akurat. Okupansi berdasarkan *overnights*:

$$\frac{Number\ of\ overnights}{Number\ of\ beds\ availabe\ for\ sale} \times 100 \qquad 2\text{-}1$$

Okupansi berdasarkan *roomnights*:

$$\frac{Number\ of\ roomnights}{Number\ of\ beds\ availabe\ for\ sale} \times 100. \qquad 2\text{-}2$$

Persamaan 2-1 menghitung tingkat okupansi dengan membandingkan jumlah malam menginap (*overnights*) terhadap total tempat tidur yang tersedia, sehingga sesuai untuk hotel dengan variasi kapasitas kamar. Persamaan 2-2 menghitung okupansi berdasarkan jumlah kamar terjual (*roomnights*) dan lebih umum digunakan karena sederhana serta berfokus pada unit kamar sebagai produk utama. Selain metode perhitungan, Ivanov (2014), menyatakan bahwa tingkat okupansi dipengaruhi oleh berbagai faktor seperti hari dalam minggu, musim, segmen pasar, acara khusus, dan strategi pemasaran hotel. Pemahaman terhadap perhitungan dan faktor-faktor tersebut penting bagi manajemen hotel dalam perencanaan operasional, penetapan harga, dan pengelolaan sumber daya secara efektif.

2.2.2 Data dan Pengolahan Data

2.2.2.1 Data *Time Series*

Tabel 2. 1 Tipe *Data Time Series*

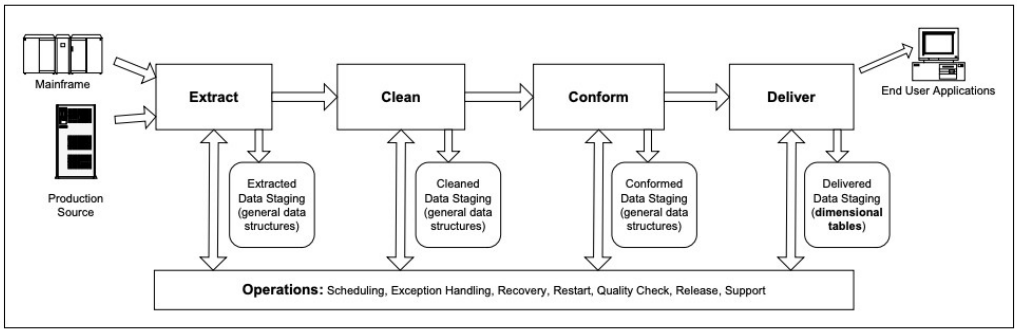
Komponen	Definisi	Contoh
Trend	Arah pergerakan data dalam jangka panjang, dapat meningkat, menurun, atau stabil	Peningkatan penggunaan mobil listrik dari tahun ke tahun
Seasonality (Musiman)	Pola yang berulang secara teratur dalam periode tertentu, seperti harian, bulanan, atau tahunan	Kenaikan permintaan listrik saat musim panas
Cyclic Variations (Siklus)	Pola berulang jangka panjang yang tidak terikat musim tertentu	Siklus naik turun pasar properti dalam beberapa tahun
Irregular Variations (Acak)	Perubahan mendadak yang tidak dapat diprediksi dan tidak dijelaskan oleh komponen lain	Kejatuhan pasar saham akibat kejadian tak terduga

Sumber: (MathWorks, n.d.)

Time series merupakan data yang dikumpulkan secara berurutan berdasarkan waktu, seperti penjualan harian, harga saham per jam, atau tingkat okupansi hotel bulanan. Ciri utamanya adalah adanya ketergantungan antar nilai yang berdekatan dalam waktu, di mana nilai saat ini dipengaruhi oleh nilai sebelumnya Box et al (2016) sehingga memerlukan metode analisis khusus yang mempertimbangkan aspek temporal.

Data *time series* umumnya tersusun atas empat komponen sebagaimana ditunjukkan pada Tabel 2.1, yaitu *trend* sebagai arah pergerakan jangka panjang, *seasonality* sebagai pola musiman yang berulang, *cyclic variations* sebagai pola siklus jangka panjang, dan *irregular variations* sebagai fluktuasi acak yang tidak terduga. Menurut Box et al (2016), analisis *time series* bertujuan memodelkan ketergantungan temporal tersebut agar pola historis dapat dimanfaatkan untuk menghasilkan prediksi nilai di masa depan secara lebih akurat.

2.2.2.2 ETL (*Extract, Transform, Load*)



Gambar 2.1 Four Staging Steps dalam Proses ETL Data Warehouse
Sumber : Kimball & Caserta (2004)

Extract-Transform-Load (ETL) merupakan fondasi utama dalam sistem *data warehouse* yang berfungsi untuk mengekstrak data dari sistem sumber, meningkatkan kualitas dan konsistensi data, serta menyajikannya dalam format yang siap dianalisis. Kimball & Caserta (2004) menjelaskan bahwa proses ETL terdiri dari empat tahapan utama, yaitu *extract* untuk pengambilan data dari sumber, *clean* untuk pembersihan dan peningkatan kualitas data, *conform* untuk menyelaraskan data dari berbagai sumber, dan *deliver* untuk menyajikan data yang siap digunakan. Tahapan utama dalam proses ETL digambarkan pada Gambar 2.1. Meskipun ETL merupakan proses *backroom* yang tidak terlihat langsung oleh pengguna akhir, Kimball & Caserta (2004) menekankan bahwa aktivitas ini memiliki peran kritis dalam memastikan keandalan dan konsistensi data yang digunakan dalam analisis.

2.2.2.3 Normalisasi Data

Normalisasi data merupakan tahap penting sebelum data digunakan dalam model *machine learning*, karena perbedaan skala antar atribut numerik dapat memengaruhi kinerja algoritma. Géron (2019) menyebutkan dua metode normalisasi yang umum digunakan, yaitu *min-max scaling* dan *standardization*.

Min-max scaling mengubah nilai data ke dalam rentang 0 hingga 1 dengan mengurangkan nilai minimum dan membaginya dengan selisih nilai maksimum dan minimum. Sementara itu, *standardization* memusatkan data pada nilai nol dengan mengurangi *mean* dan membaginya dengan *standard deviation*, sehingga

menghasilkan data dengan *unit variance*. Géron (2019) menekankan bahwa *neural networks* umumnya bekerja lebih optimal dengan nilai input pada rentang 0 hingga 1, sehingga pemilihan metode normalisasi perlu disesuaikan dengan karakteristik data dan algoritma yang digunakan.

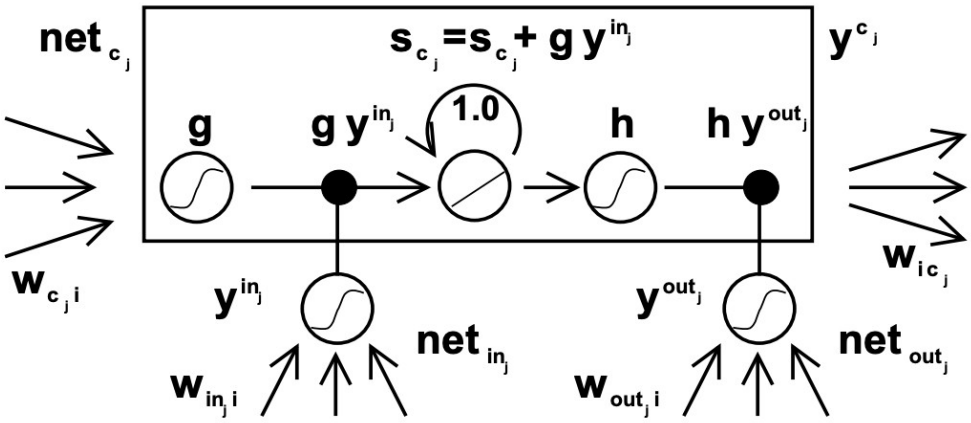
2.2.3 Machine Learning

Machine Learning dapat dikategorikan menjadi empat jenis utama berdasarkan cara model belajar dari data Géron (2019), yaitu *supervised learning*, *unsupervised learning*, *semi-supervised learning*, dan *reinforcement learning*. *Supervised learning* melatih model menggunakan data berlabel untuk mempelajari hubungan *input–output* dan melakukan prediksi pada data baru, seperti prediksi harga rumah, klasifikasi *spam*, dan prediksi okupansi hotel. *Unsupervised learning* bekerja pada data tanpa label untuk menemukan pola atau struktur tersembunyi, misalnya pengelompokan pelanggan atau deteksi anomali. *Semi-supervised learning* mengombinasikan data berlabel dalam jumlah terbatas dengan data tanpa label, sedangkan *reinforcement learning* memungkinkan model belajar melalui interaksi dengan lingkungan menggunakan mekanisme *reward* dan *penalty* Géron (2019).

Penelitian ini menggunakan pendekatan *supervised learning* karena tersedia data historis okupansi hotel yang dapat digunakan sebagai label untuk melatih model *Long Short-Term Memory (LSTM)* dalam memprediksi okupansi masa depan. Menurut Géron (2019), *Machine Learning* merupakan ilmu dan seni dalam membuat komputer belajar dari data melalui *training set* yang terdiri dari *samples*, sehingga sistem mampu meningkatkan kinerjanya pada suatu tugas tanpa harus diprogram secara eksplisit.

Keunggulan utama *Machine Learning* dibandingkan pemrograman tradisional adalah kemampuannya mendeteksi pola secara otomatis tanpa aturan manual, sehingga solusi yang dihasilkan lebih ringkas, mudah dipelihara, dan adaptif terhadap perubahan data Géron (2019).Selain itu, pendekatan ini efektif untuk menangani permasalahan kompleks dan mendukung proses *data mining* dalam menemukan pola, korelasi, atau tren baru yang sebelumnya tidak terlihat.

2.2.4 Algoritma *Long Short-Term Memory (*LSTM)



Gambar 2.2 Arsitektur *Memory Cell* LSTM dengan *Gate Units*
Sumber : Hochreiter & Schmidhuber (1997)

Neural network konvensional seperti *feedforward networks* dan *convolutional networks* tidak memiliki memori, sehingga setiap input diproses secara terpisah tanpa menyimpan *state* dari input sebelumnya Chollet (2021). Keterbatasan ini diatasi oleh *Recurrent Neural Network* (*RNN*), yang mampu memproses data berurutan dengan mempertahankan *state* untuk menangkap dependensi temporal. Namun, *RNN* mengalami kesulitan dalam mempelajari dependensi jangka panjang akibat melemahnya aliran *error* saat *backpropagation*, yang dikenal sebagai *vanishing gradient problem* Hochreiter & Schmidhuber (1997).

Untuk mengatasi masalah tersebut, *Long Short-Term Memory* (*LSTM*) diperkenalkan oleh Hochreiter & Schmidhuber (1997). *LSTM* dirancang agar mampu mempelajari *time lags* panjang dengan menjaga aliran *error* tetap stabil melalui mekanisme *constant error carousels* dan penggunaan *multiplicative gate units* yang mengatur aliran informasi. Arsitektur *memory cell* LSTM beserta komponen *gate units*-nya ditunjukkan pada Gambar 2.2.

Mekanisme *gate* pada *LSTM* terdiri dari *forget gate* untuk menentukan informasi yang dihapus, *input gate* untuk mengatur informasi baru yang disimpan, dan *output gate* untuk menghasilkan keluaran berdasarkan *cell state*. Struktur ini membuat *LSTM* lebih efektif dibandingkan *RNN* konvensional dalam menangkap dependensi jangka

panjang, sehingga sangat sesuai untuk aplikasi *time series forecasting*, termasuk prediksi okupansi hotel dengan pola temporal dan musiman yang kompleks.

Secara matematis, keunggulan LSTM terletak pada mekanisme *forget gate* yang memungkinkan jaringan menentukan informasi mana yang dipertahankan atau dibuang dari cell state. Mekanisme ini diperkenalkan oleh Gers et al. (2000) sebagai perluasan dari arsitektur LSTM awal, sehingga model mampu melakukan prediksi berkelanjutan pada data berurutan. Pembaruan cell state dan keluaran hidden state pada LSTM dapat dinyatakan sebagai berikut:

$$C_t = f_t \odot C_{t-1} + i_t \odot g(x_t \quad \text{2-3}$$

$$h_t = o_t \odot \tanh(C_t) \quad \text{2-4}$$

dimana:

C_t : adalah cell state pada waktu t yang menyimpan informasi jangka panjang

f_t : adalah aktivasi forget gate yang mengontrol informasi lama yang dipertahankan

C_{t-1} : adalah cell state dari timestep sebelumnya

i_t : adalah aktivasi input gate yang mengatur informasi baru yang masuk

$g(x_t)$: adalah fungsi aktivasi kandidat nilai baru dari input

o_t : adalah aktivasi output gate yang menentukan keluaran sel

h_t : adalah hidden state yang menjadi keluaran model pada setiap timestep

$\odot$: adalah operasi perkalian elementwise

Persamaan 2-3 menunjukkan bahwa *cell state* diperbarui dengan menggabungkan informasi lama yang diloloskan *forget gate* dan informasi baru yang diatur *input gate*, sedangkan Persamaan 2-4 menunjukkan bahwa *hidden state* sebagai keluaran model diperoleh dari *cell state* yang difilter oleh *output gate* melalui fungsi aktivasi tanh. Struktur ini memungkinkan LSTM mengatasi masalah *vanishing gradient* pada RNN konvensional dengan menjaga aliran informasi tetap stabil sepanjang sekuens data (Hochreiter & Schmidhuber (1997); Gers et al., (2000).

2.2.5 Bahasa Pemrograman Python

Python merupakan bahasa pemrograman *interpreted* yang pertama kali diperkenalkan pada tahun 1991 dan kini menjadi salah satu bahasa *interpreted* paling populer di dunia McKinney (2022). Python *interpreter* mengeksekusi program secara bertahap, satu *statement* dalam satu waktu, sehingga memberikan fleksibilitas dalam pengembangan dan *debugging*.

Menurut McKinney (2022), perkembangan komunitas *scientific computing* dan *data analysis* menjadikan Python bertransformasi dalam dua dekade terakhir menjadi bahasa utama untuk *data science*, *machine learning*, dan pengembangan perangkat lunak di lingkungan akademik maupun industri. Python didukung oleh ekosistem *library* yang lengkap, seperti *NumPy* untuk komputasi numerik, *Pandas* untuk manipulasi data, *Matplotlib* dan *Seaborn* untuk visualisasi, serta *TensorFlow* dan *Keras* untuk pengembangan model *deep learning*. Selain itu, ketersediaan *IPython* dan *Jupyter notebooks* memungkinkan eksplorasi data dan pengembangan model secara interaktif, sehingga menjadikan Python sangat sesuai untuk penelitian *machine learning* dan *time series forecasting* McKinney (2022).

2.2.6 Evaluasi Model

Evaluasi model *forecasting* merupakan tahap penting untuk mengukur kemampuan model dalam memprediksi data baru. *Forecast error* didefinisikan sebagai selisih antara nilai aktual dan nilai prediksi yang dapat diringkas menggunakan berbagai metrik evaluasi. Berbeda dengan model klasifikasi yang menggunakan metrik akurasi, model prediksi numerik menggunakan metrik yang mengukur besarnya kesalahan prediksi. Penelitian ini menggunakan empat metrik utama yaitu MAPE, R^2, MAE, dan RMSE, dengan MAPE dan R^2 sebagai metrik primer karena kombinasi keduanya memberikan gambaran komprehensif tentang performa model (Chicco et al., 2021).

Mean Absolute Percentage Error (MAPE) mengukur rata-rata kesalahan prediksi dalam bentuk persentase terhadap nilai aktual, sehingga tidak tergantung pada satuan

pengukuran. MAPE lebih sensitif terhadap variasi relatif dibandingkan variasi absolut, namun memiliki keterbatasan yaitu hanya dapat digunakan pada data positif dan cenderung bias terhadap prediksi yang rendah (Chicco et al., 2021). Interpretasi MAPE mengikuti kategori yang dikemukakan oleh Lewis (1982, dalam Montaño et al., 2013) seperti yang ditunjukkan pada Tabel 2.2, dimana MAPE <10% dianggap sangat akurat, 10-20% dikategorikan baik, 20-50% termasuk cukup, dan >50% mengindikasikan prediksi tidak akurat. Kategori Interpretasi MAPE

Tabel 2. 2 Kategori Interpretasi MAPE

Nilai MAPE (%)	Interpretasi
< 10	Sangat Akurat (Highly Accurate Forecasting)
10 - 20	Baik (Good Forecasting)
20 - 50	Cukup (Reasonable Forecasting)
> 50	Tidak Akurat (Inaccurate Forecasting)

Sumber: Lewis (1982, dalam Montaño et al., 2013)

Koefisien determinasi (R^2) mengukur proporsi variasi pada variabel dependen yang dapat dijelaskan oleh variabel independen, dengan nilai positif yang berkisar antara 0 hingga 1 dimana nilai 1 menunjukkan prediksi sempurna (Chicco et al., 2021). Berbeda dengan MAPE yang memiliki kategori interpretasi terstandar, interpretasi R^2 bersifat kontekstual dan bergantung pada domain penelitian. R^2 fokus pada kemampuan model menangkap pola dan tren data secara keseluruhan, sementara MAPE fokus pada besarnya kesalahan prediksi, sehingga kombinasi keduanya memberikan pemahaman yang lebih lengkap tentang performa model.

Mean Absolute Error (MAE) dan Root Mean Square Error (RMSE) mengukur kesalahan dalam satuan asli data. MAE menggunakan norma L1 yang tidak memberikan penalti berlebihan pada outlier, sementara RMSE menggunakan norma L2 yang mengkuadratkan kesalahan sehingga lebih sensitif terhadap outlier (Chicco et al., 2021). Interpretasi MAE dan RMSE bersifat relatif terhadap skala data yang digunakan, dimana nilai tunggal dari metrik ini sulit diinterpretasikan tanpa mengetahui distribusi nilai aktual atau nilai maksimum kesalahan pada dataset. Perbandingan antara RMSE dan MAE memberikan informasi tentang distribusi

kesalahan, dimana jika RMSE jauh lebih besar daripada MAE maka terdapat outlier dalam kesalahan prediksi.

Penelitian ini memilih MAPE dan R^2 sebagai metrik utama karena sifatnya yang saling melengkapi. MAPE memberikan interpretasi dalam bentuk persentase kesalahan dengan kategori terstandar, sementara R^2 memberikan informasi tentang kemampuan model menjelaskan variasi data. MAE dan RMSE digunakan sebagai metrik pendukung yang memberikan informasi dalam satuan asli data (Chicco et al., 2021). Penting dipahami bahwa dalam prediksi numerik tidak ada metrik "akurasi" seperti pada klasifikasi, dimana keandalan prediksi diukur melalui kombinasi MAPE, R^2, MAE, dan RMSE yang memberikan gambaran komprehensif dari berbagai aspek.

2.2.7 Hyperparameter Tuning

Hyperparameter merupakan parameter konfigurasi yang ditentukan sebelum proses pelatihan model *deep learning* berlangsung dan secara langsung memengaruhi perilaku algoritma pelatihan serta performa model yang dihasilkan. Berbeda dengan parameter model yang dipelajari secara otomatis selama pelatihan seperti bobot jaringan, *hyperparameter* harus ditentukan secara manual atau melalui proses pencarian terstruktur. Beberapa contoh *hyperparameter* pada model LSTM meliputi *learning rate*, *batch size*, jumlah unit pada setiap *layer*, dan *dropout rate* (Vidyabharathi & Mohanraj, 2023).

Hyperparameter tuning adalah proses sistematis untuk menemukan kombinasi nilai *hyperparameter* yang menghasilkan performa model terbaik. Proses ini penting karena pemilihan *hyperparameter* yang tepat akan menghasilkan konvergensi yang lebih cepat menuju titik minimum global pada permukaan *error*, sekaligus menghindari masalah seperti *overfitting* (Vidyabharathi & Mohanraj, 2023). Sebaliknya, konfigurasi *hyperparameter* yang tidak optimal dapat menurunkan akurasi model secara signifikan meskipun arsitektur jaringan yang digunakan sudah baik.

Salah satu metode *hyperparameter tuning* yang umum digunakan adalah *grid search*, yaitu pendekatan pencarian menyeluruh yang menguji semua kombinasi nilai *hyperparameter* dalam ruang pencarian yang telah ditentukan. Setiap kombinasi dilatih

dan dievaluasi secara independen, kemudian kombinasi dengan performa terbaik dipilih sebagai konfigurasi final. Metode ini sederhana dan dapat direproduksi, namun membutuhkan sumber daya komputasi yang lebih besar seiring bertambahnya jumlah *hyperparameter* yang diuji (Vidyabharathi & Mohanraj, 2023).

(El Ghazi & Aknin, 2024) menegaskan bahwa proses *hyperparameter tuning* yang dilakukan secara sistematis terbukti meningkatkan akurasi dan robustisitas model LSTM dibandingkan dengan konfigurasi *default*. Dalam penelitiannya, penyesuaian pada jumlah unit LSTM, *dropout rate*, *learning rate*, dan *batch size* secara kolektif berkontribusi terhadap peningkatan performa model yang signifikan. Hal ini menunjukkan bahwa tidak ada satu pun *hyperparameter* yang berdiri sendiri dalam menentukan kualitas model, melainkan kombinasi seluruh parameter yang saling berinteraksi satu sama lain.

2.2.8 Implementasi Sistem Prediksi

2.2.8.1 *Framework* Laravel

Laravel adalah *framework* PHP berbasis arsitektur *Model-View-Controller* (MVC) yang dirancang untuk pengembangan aplikasi web modern Stauffer (2019). Dalam penelitian ini, Laravel versi 11 digunakan sebagai *backend* utama sistem *dashboard* dengan memanfaatkan beberapa komponen utama. *Eloquent ORM* digunakan untuk manajemen komunikasi antara aplikasi dan basis data MySQL. *Inertia.js* digunakan untuk menghubungkan *backend* Laravel dengan *frontend* Vue.js tanpa memerlukan pembuatan REST API terpisah, sehingga pengembangan antarmuka interaktif dapat dilakukan lebih efisien. Untuk keperluan ekspor laporan, *library* DomPDF digunakan untuk menghasilkan file PDF, sedangkan *Maatwebsite Excel* digunakan untuk ekspor dan impor data dalam format Excel. Selain itu, Laravel Horizon digunakan untuk memantau dan mengelola antrian proses *background job*, termasuk proses *retraining* model yang berjalan di latar belakang.

2.2.8.2 *Dashboard*

Few (2006) mendefinisikan *dashboard* sebagai tampilan visual yang menyajikan informasi paling penting untuk mencapai satu atau lebih tujuan, yang dikonsolidasikan dalam satu layar sehingga dapat dipantau secara sekilas. *Dashboard* berfungsi menyajikan informasi kritis agar pengguna dapat memahami kondisi atau status dengan cepat tanpa perlu analisis data yang mendalam. Few (2006) menekankan bahwa tampilan grafis yang dirancang dengan baik mampu menyampaikan informasi secara lebih efisien dibandingkan teks, sejalan dengan kemampuan persepsi visual manusia dalam memproses informasi secara cepat. Dalam penelitian ini, visualisasi data pada *dashboard* diimplementasikan menggunakan *library* Chart.js yang memungkinkan penyajian grafik interaktif berbasis web secara ringan dan responsif.

2.2.8.3 *Yield Management* dan Sistem Rekomendasi

Yield Management adalah metode manajemen kapasitas yang bertujuan memaksimalkan pendapatan dengan cara mengalokasikan kapasitas yang tersedia kepada permintaan yang tepat, pada waktu yang tepat, dan dengan harga yang tepat (Kimes, 1989). Konsep ini awalnya dikembangkan di industri penerbangan dan kemudian diadopsi secara luas di industri perhotelan karena kesamaan karakteristik utama, yaitu kapasitas yang bersifat tetap dan inventaris yang tidak dapat disimpan. Kamar hotel yang tidak terisi pada malam tertentu tidak dapat dijual kembali di kemudian hari, sehingga manajemen perlu mengambil keputusan yang tepat berdasarkan prediksi permintaan.

(Kimes, 1989) menjelaskan bahwa penerapan *Yield Management* yang efektif membutuhkan informasi mengenai dua hal utama, yaitu tingkat permintaan saat ini dan arah pergerakan permintaan dari waktu ke waktu. Tingkat permintaan menggambarkan seberapa besar kapasitas yang terisi pada periode tertentu, sementara arah pergerakan menunjukkan apakah permintaan sedang meningkat, stabil, atau menurun. Kombinasi kedua informasi ini memungkinkan manajemen untuk menentukan strategi yang paling sesuai, baik berupa penyesuaian harga, alokasi sumber daya, maupun peluncuran program promosi.

Dalam penelitian ini, prinsip *Yield Management* diterapkan sebagai dasar perancangan sistem rekomendasi operasional yang terintegrasi dengan hasil prediksi model LSTM. Tingkat hunian bulanan hasil prediksi dikategorikan menjadi tiga level permintaan, yaitu *High Demand* (≥70%), *Moderate Demand* (50–69%), dan *Low Demand* (<50%). Arah tren ditentukan berdasarkan selisih prediksi bulan target terhadap bulan sebelumnya, dikategorikan sebagai *Increasing* (naik >5%), *Stable* (perubahan ±5%), atau *Decreasing* (turun >5%). Kombinasi kedua faktor ini menghasilkan matriks rekomendasi yang memberikan panduan operasional spesifik bagi manajemen hotel untuk setiap skenario yang mungkin terjadi.

2.2.8.4 *User Acceptance Testing* (UAT)

User Acceptance Testing (UAT) adalah pengujian yang dilakukan untuk mengevaluasi apakah sistem yang dibangun dapat diterima dan digunakan secara efektif oleh pengguna akhir sesuai dengan tujuan pengembangannya. Dalam penelitian ini, UAT dilakukan menggunakan metode *Task-Based Usability Testing* yang berlandaskan konsep pengukuran pengalaman pengguna yang dikembangkan oleh (Tullis, 2013). Metode ini mengevaluasi kemudahan penggunaan sistem dengan cara meminta pengguna menyelesaikan serangkaian skenario tugas yang representatif secara mandiri, sementara pengamat mencatat hasil pengujian berdasarkan dua metrik utama.

Metrik pertama adalah *task success*, yaitu metrik paling umum digunakan karena dapat diterapkan pada hampir semua jenis produk dan teknologi (Tullis, 2013). *Task success* mengukur apakah pengguna berhasil menyelesaikan setiap tugas yang diberikan secara biner, yaitu berhasil atau tidak berhasil. Metrik kedua adalah *time on task*, yaitu waktu yang dibutuhkan pengguna untuk menyelesaikan setiap tugas sebagai indikator efisiensi sistem. (Tullis, 2013). menjelaskan bahwa semakin cepat pengguna menyelesaikan suatu tugas, semakin baik pengalaman penggunaan yang dirasakan. Selain kedua metrik tersebut, wawancara singkat dilakukan setelah sesi pengujian untuk menggali pendapat pengguna secara kualitatif. (Tullis, 2013) menyatakan bahwa data observasi perlu dilengkapi dengan data yang dilaporkan langsung oleh pengguna

agar pemahaman terhadap pengalaman penggunaan menjadi lebih lengkap dan menyeluruh.

2.2 Penelitian Terdahulu

Penelitian terdahulu menjadi acuan penting dalam memahami perkembangan metode prediksi tingkat hunian hotel. Tinjauan terhadap dua belas penelitian terkait dirangkum pada Tabel 2.3.

Salamanis et al (2022) membuktikan bahwa LSTM efektif mengenali pola musiman pada prediksi permintaan wisata jangka panjang, namun masih pada level destinasi dan belum spesifik pada hotel tertentu. Chang et al (2021) mengombinasikan LSTM dengan *sentiment analysis* dari ulasan pelanggan dan berhasil meningkatkan akurasi prediksi, namun implementasinya kompleks dan belum dikembangkan menjadi sistem *dashboard*. Singgalen (2025) memberikan gambaran pola okupansi perhotelan Indonesia melalui pendekatan *big data analytics*, namun bersifat deskriptif tanpa prediksi kuantitatif. Kozlovskis et al (2023) membandingkan beberapa algoritma *machine learning* dan menemukan XGBoost menghasilkan *error* terendah, namun belum mengeksplorasi model *deep learning* untuk data temporal.

Lee et al (2020) menunjukkan bahwa *Artificial Neural Network* (ANN) lebih baik dari ARIMA pada data non-linear, namun arsitekturnya belum memanfaatkan kemampuan memori jangka panjang seperti LSTM. Yi et al (2025) memperkenalkan model *Transformer* dengan *attention mechanism* yang lebih akurat dari LSTM pada dataset besar, namun kebutuhan komputasinya tinggi sehingga kurang efisien untuk hotel skala kecil hingga menengah. Elsworth & Güttel (2020) mengembangkan *Symbolic* LSTM yang lebih mudah diinterpretasikan, namun masih bersifat teoritis dan belum diaplikasikan pada kasus nyata. Manullang et al (2020) menggunakan LSTM untuk prediksi kunjungan wisata di Indonesia dengan hasil yang relevan secara konteks, namun fokusnya pada jumlah wisatawan, bukan tingkat hunian hotel.

Alfarisi et al (2025), mengusulkan model hibrida ARIMA-LSTM yang menunjukkan penurunan nilai *error* dibandingkan model tunggal, namun membutuhkan sumber daya komputasi yang lebih besar. Ampountolas (2021)

membuktikan bahwa *Neural Network* mengungguli model statistik klasik dalam prediksi permintaan harian hotel, namun dilakukan pada konteks geografis yang berbeda dari Indonesia. Dowlut & Gobin-Rahimbux (2023) menerapkan *deep learning* untuk prediksi permintaan *resort* hotel mewah dan terbukti efektif mendukung manajemen pendapatan, namun belum mengembangkan sistem prediksi berbasis web. Helya et al (2024) menggunakan *Genetic Algorithm* untuk mengoptimalkan parameter *Neural Network* dan berhasil meningkatkan akurasi, namun kompleksitasnya tinggi dan belum memanfaatkan pendekatan LSTM.

Tabel 2. 3 Perbandingan Penelitian

Peneliti	Tujuan	Model	Hasil	GAP
Salamanis et al. (2022)	Prediksi permintaan wisata jangka panjang berbasis musiman	LSTM	LSTM efektif mengenali pola musiman dan menghasilkan prediksi jangka panjang	Data masih pada level destinasi, belum spesifik hotel
Chang et al. (2021)	Prediksi okupansi hotel dengan data okupansi dan ulasan daring	LSTM + *Sentiment Analysis*	Akurasi meningkat dengan model gabungan	Kompleks dan belum diimplementasikan dalam dashboard
Singgalen (2025)	Analisis tren perhotelan Indonesia berbasis big data	*Big Data Analytics*	Memberikan gambaran pola okupansi hotel di Indonesia.	Bersifat deskriptif tanpa prediksi kuantitatif
Kozlovskis et al. (2023)	Perbandingan algoritma ML untuk prediksi okupansi hotel	RF, XGBoost, SVM	XGBoost menghasilkan error terendah	Belum mengkaji deep learning untuk data temporal
Lee et al. (2020)	Perbandingan ANN, ARIMA, dan regresi linear	ANN, ARIMA, Linear Regression	ANN lebih baik pada data non-linear	Tidak memiliki memori jangka panjang seperti LSTM
Yi et al.	Peningkatan	Transforme	Lebih akurat	Kebutuhan

(2025)	akurasi prediksi wisata berbasis attention	r	dari LSTM pada data besar	komputasi tinggi
Elsworth & Güttel (2020)	Meningkatkan interpretabilitas model deret waktu	Symbolic LSTM	Model lebih transparan secara interpretasi	Masih bersifat teoritis
Manullang et al. (2020)	Prediksi kunjungan wisata di Indonesia	LSTM	Mampu mengenali pola kunjungan wisata	Fokus wisatawan, bukan okupansi hotel
Alfarisi et al. (2025	Meningkatkan akurasi peramalan	ARIMA-LSTM	RMSE dan MAE menurun	Model kompleks dan pelatihan lebih lama
Ampountolas (2021)	Prediksi permintaan harian hotel	SARIMAX, GARCH, NN	NN unggul dari model klasik	Data luar Indonesia
Dowlut & Gobin-Rahimbux (2023)	Prediksi permintaan resort hotel	Deep Learning	Mendukung manajemen pendapatan hotel	Fokus hotel mewah
Helya et al. (2024)	Optimasi parameter NN untuk prediksi wisata	GA + NN	Akurasi meningkat melalui optimasi	Belum memanfaatkan LSTM

Berdasarkan tinjauan pada Tabel 2.3, sebagian besar penelitian berfokus pada penerapan *deep learning* untuk prediksi okupansi hotel, namun umumnya diterapkan pada hotel besar di negara maju atau bersifat teoritis tanpa implementasi sistem yang dapat digunakan langsung. Berbeda dengan penelitian sebelumnya, penelitian ini menggunakan algoritma LSTM untuk memprediksi tingkat hunian bulanan Hotel Dharma Utama dengan data historis 2021–2025, yang diimplementasikan ke dalam *dashboard* berbasis web menggunakan Laravel. Selain itu, hasil prediksi diintegrasikan dengan sistem rekomendasi berbasis *Yield Management* (Kimes, 1989) sehingga menghasilkan panduan operasional yang dapat langsung digunakan oleh manajemen hotel skala kecil hingga menengah di Indonesia.