BAB I

PENDAHULUAN

1.1 Latar Belakang

Pemeliharaan kendaraan roda empat saat ini masih bersifat reaktif, di mana perbaikan dilakukan hanya setelah terjadi kerusakan. Pendekatan ini dapat menyebabkan kendaraan mengalami *downtime* yang tidak terduga, dilihat dari data layanan Kalla Care, yang mencatat sekitar 44 pengaduan per bulan terkait kerusakan kendaraan yang terjadi di jalan. Tindakan perbaikan baru dilakukan setelah komponen mengalami kegagalan, sehingga berpotensi meningkatkan risiko operasional (Rojek dkk., 2023). Akibatnya, biaya perbaikan meningkat dan kepuasan pelanggan berkurang, sehingga menunjukkan perlunya pendekatan pemeliharaan yang lebih proaktif.

Untuk mengatasi keterlambatan perbaikan, *predictive maintenance* menawarkan solusi berbasis data yang memungkinkan deteksi dini potensi kegagalan melalui analisis riwayat perawatan kendaraan (Chen dkk., 2021). Dengan kecerdasan buatan seperti *Machine Learning* (ML), kerusakan dapat diidentifikasi sejak tahap awal (Calabrese dkk., 2022). Salah satu penerapan ML dalam *predictive maintenance* adalah sistem rekomendasi, yang dapat memberikan saran penggantian *spare part* lebih awal berdasarkan riwayat pemeliharaan kendaraan, sehingga meningkatkan efisiensi operasional dan mengurangi risiko kegagalan mendadak.

Sistem rekomendasi dalam *predictive maintenance* dapat menjadi alat bantu bagi teknisi dalam mendeteksi potensi kerusakan sejak dini. Rekomendasi ini meningkatkan efisiensi perbaikan dan mengurangi risiko kegagalan komponen. Dengan penerapan teknologi *Machine Learning* ini, teknisi di lapangan maupun *service advisor* mempunyai pengetahuan tambahan, sehingga lebih mudah dalam mendeteksi gejala awal dan mencegah kerusakan sebelum terjadi (Rojek dkk., 2023).

Meskipun sistem rekomendasi telah diterapkan dalam berbagai industri, metode konvensional seperti *collaborative filtering* atau *content-based filtering* masih memiliki beberapa kelemahan. Algoritma tersebut sering kali menghadapi permasalahan seperti data hubungan yang jarang (*sparse relation data*) antara pengguna dan entitas. Selain itu, banyak model rekomendasi yang belum memanfaatkan pengetahuan tentang hubungan antar entitas untuk menangkap ketergantungan intra-sesi dalam sebuah sistem rekomendasi berbasis sesi (Zhang dkk., 2024).

Satu kendaraan terdiri dari berbagai komponen yang saling berhubungan, di mana kegagalan satu komponen dapat menyebabkan kerusakan berantai pada bagian lainnya (Ojima dkk., 2024). Untuk memahami keterkaitan antar komponen, hubungan ini dapat dimodelkan dalam bentuk *knowledge graph* (KG), yang merepresentasikan informasi secara terstruktur dan menangkap hubungan semantik

antar entitas. Dokumen teknis yang berisi detail mengenai kondisi kendaraan, metode perbaikan, serta waktu dan konteks kerusakan dapat direpresentasikan sebagai KG untuk membantu analisis dan pengambilan keputusan pemeliharaan. Dengan pendekatan ini, sistem dapat mengidentifikasi pola kegagalan dan memberikan rekomendasi berbasis hubungan antar sparepart guna meningkatkan akurasi prediksi pemeliharaan (Ojima dkk., 2024).

KG mampu merepresentasikan hubungan antar sparepart, pola perbaikan, dan koneksi dalam sesi perawatan kendaraan. Pendekatan KG ini memungkinkan eksploitasi informasi antar-entitas dengan lebih baik dibandingkan pendekatan berbasis *collaborative filtering* atau *content-based filtering*, sehingga memberikan wawasan lebih mendalam mengenai pola pemeliharaan. KG dapat menghubungkan *spare part* yang sering diganti bersama dan membantu memperkirakan kemungkinan penggantian *spare part* berikutnya berdasarkan hubungan antar komponen, sehingga mampu memberikan rekomendasi yang lebih akurat (Wang dkk., 2019).

Namun, representasi hubungan dalam KG memerlukan sebuah model yang mampu mengeksplorasi konektivitas kompleks antar entitas secara efektif. Untuk menjawab tantangan ini, Knowledge Graph Attention Network (KGAT) diperkenalkan sebagai sebuah metode yang secara ekspxlisit memodelkan hubungan tingkat tinggi (*high-order relations*)—yaitu hubungan tidak langsung yang melibatkan beberapa entitas perantara—dalam struktur KG (Wang dkk., 2019). KGAT bekerja dengan cara menyebarkan informasi (*embedding propagation*) dari entitas tetangga untuk memperkaya representasi sebuah *spare part*. Lebih penting lagi, model ini mengadopsi mekanisme atensi (*attention mechanism*) yang memungkinkannya untuk secara cerdas menentukan bobot kepentingan dari setiap hubungan. Dalam konteks pemeliharaan kendaraan, mekanisme ini membantu model untuk memprioritaskan hubungan antar *spare part* yang paling krusial, sehingga menghasilkan rekomendasi yang lebih akurat.

Oleh karena itu, penelitian ini mengusulkan Judul “Implementasi *Knowledge Graph Attention Network* (KGAT) pada Sistem Rekomendasi Pemeliharaan *Spare Part* Kendaraan Roda Empat”. Sistem ini dirancang untuk mengatasi kelemahan metode konvensional dengan memanfaatkan kemampuan KGAT dalam memodelkan hubungan tingkat tinggi (*high-order relations*) dan memprioritaskan koneksi antar *spare part* yang paling relevan melalui mekanisme atensi. Dengan demikian, sistem rekomendasi yang dihasilkan diharapkan tidak hanya mampu memberikan saran penggantian *spare part* yang lebih akurat berdasarkan pola historis, tetapi juga berkontribusi secara signifikan terhadap peningkatan efisiensi perbaikan, penurunan risiko kegagalan mendadak, dan peningkatan keandalan kendaraan secara keseluruhan.

1.2 Landasan Teori

1.2.1 Sistem Rekomendasi

Sistem rekomendasi adalah teknologi yang dirancang untuk menyarankan item yang paling sesuai bagi pengguna. Saran ini didasarkan pada preferensi (kesukaan), perilaku, ataupun kebutuhan pengguna pada situasi tertentu. Fungsi utama sistem ini adalah mengatasi "*information overload*" dengan cara menyaring sejumlah besar pilihan yang ada dan hanya menampilkan yang berpotensi diminati oleh pengguna (Avazpour dkk., 2014). Dengan demikian, sistem ini berperan penting dalam mendukung pengambilan keputusan, karena membantu pengguna menemukan produk, layanan, atau informasi yang mereka butuhkan secara lebih cepat. Saat ini, sistem rekomendasi telah diterapkan secara luas di berbagai bidang, mulai dari *e-commerce* (saran produk), layanan *streaming* musik dan video (rekomendasi film), media sosial (personalisasi konten), hingga bidang khusus seperti layanan otomotif untuk menyarankan jadwal perawatan atau penggantian suku cadang (Shani & Gunawardana, 2011).

Untuk menghasilkan rekomendasi yang tepat, terdapat beberapa teknik yang umum digunakan dalam sistem rekomendasi, di antaranya:

1. *Content-based, merupakan* sistem rekomendasi berbasis konten (*content-based*) yang berfokus untuk menyarankan item yang serupa dengan item yang pernah diminati oleh pengguna sebelumnya. Menurut Lops dkk., (2011) , konsep dasarnya adalah. dengan membandingkan karakteristik suatu item terhadap profil pengguna, yang menyimpan data preferensi dan minat mereka. Jika seorang pengguna pernah menyukai item tertentu di masa lalu, sistem akan mengidentifikasi item lain dengan atribut serupa dan merekomendasikannya karena dianggap relevan.
2. *Collaborative Filtering* (CF), adalah sistem rekomendasi yang menghasilkan saran personal bagi pengguna berdasarkan pola peringkat (*rating*) atau data penggunaan. Berbeda dengan *content-based*, CF tidak memerlukan informasi eksternal mengenai pengguna ataupun item. Sebaliknya, teknik ini sepenuhnya mengandalkan perilaku kolektif dari sekumpulan pengguna. Logika dasarnya adalah jika dua pengguna menunjukkan preferensi yang serupa di masa lalu, sistem berasumsi bahwa selera mereka akan tetap sama di masa mendatang. Demikian pula, item yang dinilai atau digunakan dengan cara yang sama oleh banyak pengguna dapat direkomendasikan kepada pengguna lain yang memiliki pola serupa.

   CF dapat bekerja menerima masukan dari dua jenis *feedback* utama. Yang pertama, *explicit feedback*, terjadi ketika pengguna secara langsung menyatakan preferensinya, seperti memberikan peringkat bintang pada film atau ikon suka/tidak suka (*thumbs-up/down*). Sementara itu, *implicit feedback* bersifat tidak langsung dan disimpulkan dari perilaku pengguna, misalnya riwayat pembelian, pola penjelajahan (browsing), atau aktivitas pencarian. Karena data eksplisit tidak selalu tersedia, umpan balik implisit memainkan peran krusial dalam meningkatkan kualitas rekomendasi, terutama bagi pengguna yang jarang memberikan masukan langsung (Koren & Bell, 2011).

3. *Knowledge-based,* menghasilkan rekomendasi yang berasal dari pengetahuan spesifik suatu bidang (*domain knowledge*) tentang bagaimana fitur-fitur suatu item sesuai dengan kebutuhan dan preferensi pengguna. Berbeda dengan metode kolaboratif atau berbasis konten, sistem ini tidak bergantung pada data historis seperti peringkat atau perilaku pengguna. Sebaliknya, fokus utamanya adalah memahami hubungan antara kebutuhan pengguna dan karakteristik item untuk dapat menyarankan pilihan yang paling sesuai.
   Dua pendekatan yang umum dalam kategori ini adalah *case-based* dan *constraint-based*. Pendekatan berbasis kasus bekerja dengan membandingkan kebutuhan pengguna (yang dianalogikan sebagai masalah) dengan item-item yang menjadi solusinya. Sebuah fungsi kemiripan (*similarity function*) digunakan untuk mengukur tingkat kecocokan antara item dan kebutuhan pengguna. Sebaliknya, pendekatan berbasis batasan mengandalkan aturan-aturan eksplisit yang tersimpan dalam sebuah basis pengetahuan (*knowledge base*) (Ricci dkk., 2011).
4. *Hybrid model,* menggabungkan dua atau lebih teknik rekomendasi untuk meningkatkan performa. Metode ini sering kali digunakan untuk mengatasi keterbatasan pada metode tunggal, seperti masalah *cold-start*, di mana pengguna atau item baru tidak memiliki data yang cukup untuk prediksi akurat. Dengan memadukan berbagai metode, *hybrid model* dapat memanfaatkan keunggulan dari setiap pendekatan, sehingga menghasilkan rekomendasi yang lebih akurat. Penelitian telah menunjukkan bahwa sistem semacam ini sering kali lebih efektif daripada pendekatan metode tunggal (Burke, 2007).
   Knowledge Graph Attention Network (KGAT) adalah salah satu contoh *hybrid recommender system*. Meskipun KGAT sering diklasifikasikan sebagai *knowledge-based recommender*, model ini juga mengintegrasikan sinyal dari *collaborative filtering*. Pendekatan CF memanfaatkan pola interaksi pengguna-item, sementara *knowledge graph* menyajikan hubungan semantik antar entitas seperti item dan atributnya. KGAT belajar dari kedua sumber data ini secara bersamaan, menggabungkan data perilaku dengan pengetahuan domain yang terstruktur. Desain hibrida ini terbukti dapat mengurangi masalah seperti kelangkaan data (*data sparsity*) dan *cold-start*, sekaligus meningkatkan akurasi rekomendasi (Wang dkk., 2019).

Evaluasi merupakan langkah krusial dalam pengembangan sistem rekomendasi, karena menjadi tolak ukur kualitas rekomendasi yang dihasilkan dan memastikan relevansinya dengan kebutuhan pengguna. Tanpa evaluasi yang tepat, model yang canggih sekalipun berisiko menghasilkan rekomendasi yang gagal memenuhi harapan pengguna atau tujuan bisnis. Menurut Jadon & Patil (2024), performa sistem rekomendasi tidak dapat digambarkan hanya dengan satu metrik karena bersifat multidimensi. Untuk menilai sistem ini secara menyeluruh, diperlukan penggunaan berbagai metrik, di mana setiap metrik memberikan *insight* yang

berbeda mengenai aspek-aspek kinerja sistem. Untuk mengukur performa tersebut, beberapa metrik yang relevan adalah:

1. *Mean Reciprocal Rank* (MRR), adalah metrik statistik yang digunakan untuk mengevaluasi posisi kemunculan item relevan pertama dalam daftar rekomendasi yang ber-peringkat. Konsep dasarnya sederhana: semakin di atas posisi item relevan pertama, semakin baik performa sistem tersebut. Untuk setiap kueri, jika item relevan ditemukan pada peringkat ke-k, maka nilainya adalah $1/k$. Selanjutnya, nilai MRR dihitung dengan mengambil nilai rata-rata dari seluruh *reciprocal rank* pada semua kueri yang dievaluasi. Untuk satu set kueri Q, MRR dihitung dengan:

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i} \tag{1.1}$$

   dimana $|Q|$ adalah jumlah total kueri dan $rank_i$ adalah posisi (peringkat) item relevan pertama untuk kueri ke-i.

2. *Normalized Discounted Cumulative Gain* (nDCG), adalah metrik yang mengukur kualitas *ranking* dengan mempertimbangkan dua faktor utama: urutan posisi dan tingkat relevansi dari setiap item. Berbeda dengan MRR yang hanya berfokus pada item relevan pertama, nDCG mengevaluasi kualitas seluruh daftar rekomendasi. Perhitungannya diawali dengan *Discounted Cumulative Gain* (DCG), yaitu nilai yang mengakumulasi skor relevansi dari atas ke bawah, namun dengan memberikan 'diskon' (penalti) logaritmik pada item yang berada di posisi lebih rendah. Artinya, item relevan yang berada di posisi bawah akan memberikan kontribusi nilai yang lebih kecil.

$$DCG_p = \sum_{i=1}^{p}\frac{2^{rel_i}-1}{log_2(i+1)} \tag{1.2}$$

   dengan rel_i adalah skor relevansi item pada posisi ke-i dan p adalah panjang daftar rekomendasi. Agar skor DCG dapat dibandingkan antar kueri, nilai tersebut dinormalisasi dengan cara membaginya dengan nilai *Ideal DCG* (iDCG), yaitu skor DCG dari sebuah daftar peringkat yang dianggap sempurna. Hasil normalisasi inilah yang disebut nDCG

$$nDCG_p = \frac{DCG_p}{iDCG_p} \tag{1.3}$$

   Nilai nDCG berkisar dari 0 hingga 1, di mana skor 1 menandakan urutan rekomendasi yang sempurna.

3. *Precision@k* adalah metrik yang mengukur proporsi item relevan yang terdapat dalam sejumlah k rekomendasi teratas. Metrik ini mencerminkan seberapa akurat saran-saran yang ditampilkan sistem pada posisi teratas bagi pengguna.

$$Precision@k = \frac{Jumlah\ item\ relevan\ dalam\ k\ teratas}{k} \quad (1.4)$$

Dalam konteks ini, "item relevan" merujuk pada item yang di anggap diminati oleh pengguna berdasarkan data acuan yang valid (*ground truth*), seperti riwayat interaksi pengguna atau peringkat yang diberikan secara langsung. Dengan demikian, metrik ini memberikan indikator yang lugas mengenai kualitas rekomendasi untuk daftar dengan jumlah k yang telah ditentukan.

4. *Recall@k*, berbeda dengan *Precision@k*, adalah metrik yang mengevaluasi seberapa banyak item relevan yang berhasil ditemukan oleh sistem, dan ditampilkan dalam k rekomendasi teratas, dari keseluruhan jumlah item relevan yang ada. Metrik ini mengukur kemampuan sistem untuk menemukan kembali (*retrieve*) semua item yang seharusnya diminati oleh pengguna.

$$Recall@k = \frac{Jumlah\ item\ relevan\ dalam\ k\ teratas}{jumlah\ total\ item\ relevan} \quad (1.5)$$

Nilai *recall* yang tinggi menandakan bahwa pengguna disajikan sebanyak mungkin pilihan relevan.

5. *Hit-Rasio*@K, adalah sebuah metrik yang mengukur seberapa sering setidaknya satu item relevan muncul dalam daftar K rekomendasi teratas yang diberikan kepada pengguna. Jika item preferensi aktual seorang pengguna ditemukan dalam K sugesti teratas, maka hal tersebut dihitung sebagai sebuah "hit"; sebaliknya, jika tidak ditemukan, dihitung sebagai "miss".

$$HR@K = \frac{1}{|U|}\sum_{u \in U} \partial\left(\hat{R}(u) \cap R(u) \neq \emptyset\right) \quad (1.6)$$

$\partial(\cdot)$ adalah fungsi indikator. $\partial(b) = 1$ jika kondisi b benar, dan bernilai 0 jika tidak. $\emptyset$ menyatakan himpunan kosong (Li dkk., 2020).

Evaluasi sistem tidak hanya dilakukan dengan metrik kuantitatif, namun juga dapat dilengkapi dengan penilaian dari pengguna secara langsung. Salah satu metode umum adalah melalui formulir *feedback* yang menggunakan skala Likert, di mana responden menilai tingkat kesepakatan atau kepuasan mereka pada skala

terstruktur (misalnya, skala 1-5). Pendekatan ini memungkinkan peneliti untuk mengukur faktor-faktor seperti kemudahan penggunaan, kejelasan rekomendasi, dan penerimaan sistem secara keseluruhan dengan cara yang terstandarisasi dan dapat diukur (Knijnenburg dkk., 2012).

Sektor otomotif, khususnya dalam bidang perawatan kendaraan, merupakan area yang sangat potensial untuk penerapan sistem rekomendasi. Keputusan terkait penggantian suku cadang dan jadwal perawatan sering kali menjadi rumit akibat permintaan yang tidak pasti, adanya ketergantungan antar komponen, serta tingginya biaya yang disebabkan oleh waktu henti kendaraan (*downtime*). Sistem rekomendasi dapat menjawab tantangan-tantangan ini dalam bentuk pemeliharaan prediktif (*predictive maintenance*), optimisasi inventaris, dan proses pengambilan keputusan di bengkel.

Sebagai contoh, sistem rekomendasi telah dirancang untuk mendukung pengisian kembali stok suku cadang, yang membantu para *stakeholder* menghadapi *demand* yang tidak pasti dan *supply* yang fluktuatif dengan memberikan saran berbasis data (Hinrichs dkk., 2023). Model *predictive spare parts maintenance* terbukti meningkatkan efisiensi operasional dengan memprediksi waktu penggantian dan mengurangi *downtime*, sehingga meningkatkan layanan purnajual (*after-sales*) dan kepuasan pelanggan (Usman, 2024). Serupa dengan itu, pengintegrasian *predictive maintenance* dengan manajemen inventaris suku cadang dapat mengurangi jadwal *maintenance* yang tidak terencana, menekan biaya inventaris, dan meningkatkan ketahanan (*resilience*) di sektor otomotif (Shokri dkk., 2025).

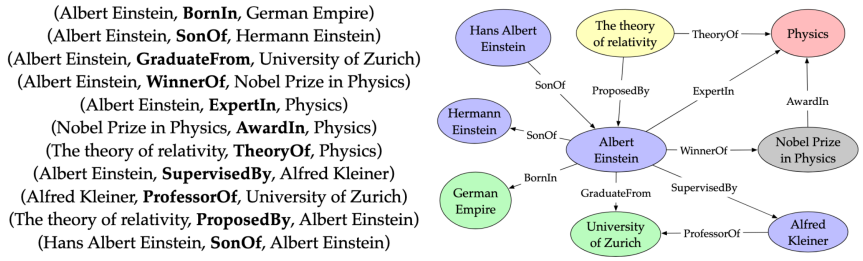
Di luar prediksi dan kontrol inventaris, sistem rekomendasi juga memiliki kontribusi untuk aspek *sustainability* dan efisiensi biaya. Contohnya, platform digital dikembangkan untuk merekomendasikan dan menjadi wadah pertukaran suku cadang bekas, yang mengedepankan keterjangkauan harga sekaligus tujuan pelestarian lingkungan (Varun Kumar B, 2025). Lebih lanjut, *framework* manajemen stok yang dirancang khusus untuk suku cadang otomotif mampu menjamin ketersediaan barang sekaligus meminimalkan biaya, membuktikan peran penting model *data-driven decision making* (Teixeira et al., 2024).

Secara ringkas, penerapan sistem rekomendasi dalam domain *maintenance* otomotif dapat meningkatkan akurasi prediksi kegagalan *spare parts*, mengoptimalkan inventaris, mengurangi biaya, serta meningkatkan aspek *sustainability*. Aplikasinya secara langsung berkontribusi pada peningkatan efisiensi layanan *after-sales* serta kepercayaan pelanggan terhadap jasa perbaikan dan perawatan kendaraan.

1.2.2 Knowledge Graph (KG)

Graf pengetahuan (*knowledge graph*) adalah sebuah representasi terstruktur dari informasi, yang memodelkan entitas dunia nyata (seperti orang, tempat, atau objek) beserta hubungan di antara mereka dalam bentuk *node* (simpul) dan *edge* (tepi) dalam sebuah struktur graf. Konsep dasarnya adalah pada upaya menangkap dan mengorganisasi pengetahuan ke dalam bentuk yang dapat ditafsirkan baik oleh

manusia maupun mesin. Sehingga KG memungkinkan sistem untuk memiliki kapabilitas penalaran (*reasoning*), pencarian, dan rekomendasi yang lebih canggih (Abu-Salih, 2021; Ji dkk., 2022; J.-C. Zhang dkk., 2024b). KG dibangun dengan mengekstrak *entity* dan *relation* dari berbagai sumber data, yang kemudian direpresentasikan sebagai *triple* yang saling terhubung, yaitu (subjek, predikat, objek). Secara kolektif, kumpulan *triple* ini membentuk sebuah jaringan semantik yang mencerminkan domain pengetahuan asalnya.

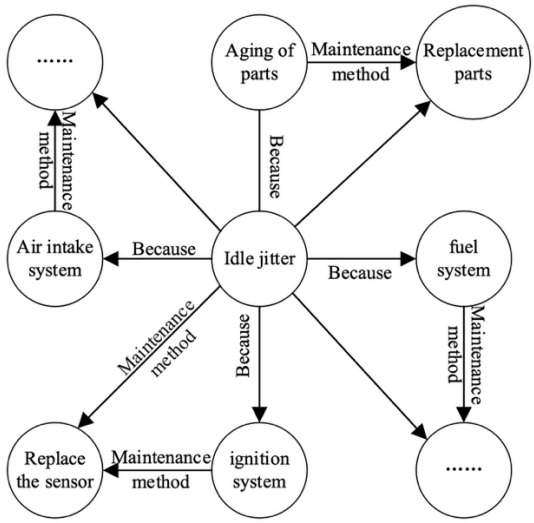


Gambar 1. Triples (kiri) dan entitas dan relasinya (kanan) dalam KG (Ji dkk., 2022).

Dasarnya, KG terbentuk dari tiga komponen utama, yaitu *node* (entitas), *edge* (relasi), dan atribut;

- *Node*, atau entitas, mewakili objek atau konsep dari dunia nyata seperti orang, tempat, produk, atau peristiwa. Setiap *node* berfungsi sebagai titik pusat dalam graf yang menandakan keberadaan sebuah entitas dalam domain yang sedang dimodelkan.
- *Edge*, atau relasi, merupakan koneksi antar-*node* untuk menjelaskan bagaimana entitas saling berhubungan (misalnya, "merupakan bagian dari", "terletak di", atau "menggantikan"). Hubungan ini sangat penting untuk menangkap struktur semantik dari graf pengetahuan.
- Sementara itu, atribut menyediakan informasi deskriptif tambahan tentang *node* atau *edge*, seperti tahun model mobil, jenis suku cadang, atau tanggal perawatan. Atribut memperkaya graf dengan menambahkan konteks dan detail, sehingga memungkinkan kueri dan rekomendasi yang lebih presisi

Secara keseluruhan, semua komponen ini memungkinkan graf pengetahuan untuk memodelkan informasi yang kompleks dan saling terhubung secara terstruktur dan mudah ditafsirkan (Jadon & Patil, 2024; Ji dkk., 2022; Sheth dkk., 2019).



Gambar 2. Knowledge Graph untuk Automobile Engine Fault dalam Tang dkk. (2023).

KG dapat merepresentasikan pengetahuan terstruktur di domain otomotif, di mana terdapat hubungan yang kompleks antara kendaraan, suku cadang, tugas perawatan, dan kebutuhan pelanggan. Hal ini sejalan dengan penelitian yang menunjukkan bagaimana KG dapat secara efektif memetakan domain pemeliharaan kerusakan mesin mobil (*automobile engine fault maintenance*) untuk meningkatkan efisiensi dan akurasi diagnosis (Tang dkk., 2023). Dengan memodelkan entitas seperti model mobil, komponen, dan catatan servis, graf pengetahuan memungkinkan integrasi data otomotif yang beragam (heterogeneous) dan mendukung aplikasi cerdas seperti perawatan prediktif (predictive maintenance) dan diagnosis kerusakan. Sifatnya yang terstruktur membantu menangkap semantik spesifik domain, menjadikannya cocok untuk penalaran dan identifikasi ketergantungan (dependensi) dalam sistem kendaraan (Yang dkk., 2022)

1.2.3 GNN

Graph Neural Networks (GNNs) adalah kelas model *deep learning* yang dirancang khusus untuk memproses dan menganalisis data berstruktur graf. Keunggulan GNN terletak pada kemampuannya untuk menangkap dependensi yang kompleks dan informasi relasional dalam data, menjadikannya sangat kuat untuk tugas yang melibatkan data bersifat *non-Euclidean* dan memiliki keterhubungan tinggi (Zhou dkk., 2022). Prinsip utama dari GNN terdiri dari:

- *Graph Representation*: Proses untuk menyederhanakan struktur graf yang rumit, yang terdiri dari *nodes* dan *edges*, menjadi format berdimensi rendah yang mudah diolah oleh komputer. Melalui representasi graf, fitur-fitur penting mengenai setiap simpul dan relasinya dapat ditangkap, sehingga tugas seperti prediksi atau klasifikasi menjadi lebih mudah. Hal ini

memungkinkan model *machine learning* untuk mengambil keputusan yang lebih akurat dari data graf.

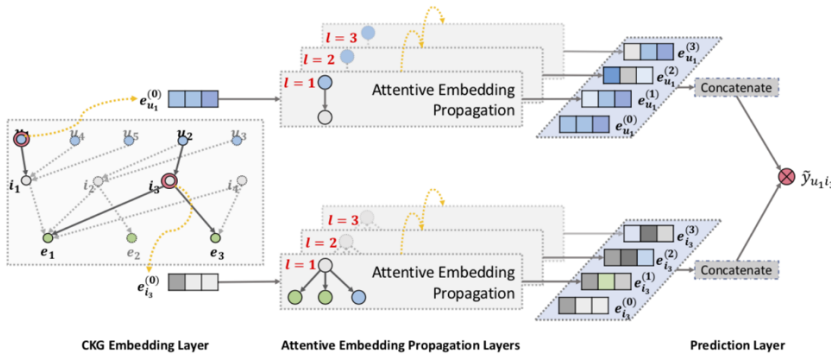
- *Message Passing*: Operasi dasar pada GNN adalah *message passing* (bisa juga disebut *neighbor aggregation*). Dalam proses ini, setiap simpul memperbarui representasinya dengan cara mengumpulkan (*aggregate*) informasi dari simpul-simpul tetangganya dan dari dirinya sendiri melalui sebuah fungsi yang dapat dilatih (*learnable function*). Proses ini diulang pada beberapa lapisan, sehingga memungkinkan sebuah simpul untuk menangkap informasi dari bagian graf yang jangkauannya lebih luas (Malla, t.t.; Zhou dkk., 2022).

GNN telah menjadi pendekatan yang kuat dalam sistem rekomendasi berkat kemampuannya untuk memodelkan interaksi *user*-item sebagai sebuah graf dan menangkap hubungan yang kompleks dan *high-order*. Berbeda dengan metode tradisional, GNN bekerja dengan menyebarkan (*propagate*) dan *aggregate* informasi antar-simpul yang terhubung, sehingga mampu menghasilkan representasi (*embeddings*) yang lebih kaya untuk *user-item*. Kemampuan ini membuat GNN sangat efektif untuk mengatasi tantangan seperti kelangkaan data (*data sparsity*) dan masalah *cold-start*, karena GNN memanfaatkan dependensi struktural dan sinyal kontekstual di dalam graf (Chu dkk., 2022; Gao dkk., 2022).

Dari konsep ini, *Knowledge Graph Attention Network* (KGAT) merupakan pengembangan dari sistem rekomendasi berbasis GNN dengan mengintegrasikan KG sebagai informasi tambahan (*side information*) dan menerapkan mekanisme atensi (*attention mechanism*). KG memperkaya rekomendasi dengan menghubungkan item dengan berbagai atribut dan relasinya, sehingga membantu model menemukan koneksi semantik yang lebih dalam, di luar data historis (Huang, 2022).

1.2.4 KGAT

Knowledge Graph Attention Network (KGAT) adalah model yang dirancang untuk memperkaya sistem rekomendasi dengan menggabungkan informasi struktural dan semantik dari *knowledge graph* (KG) dengan kemampuan pembelajaran GNN. Tujuan utama pengembangan KGAT adalah untuk mengatasi keterbatasan pada algoritma rekomendasi tradisional, seperti masalah kelangkaan data (*data sparsity*), *cold-start*, dan ketidakmampuan untuk menangkap hubungan kompleks antara pengguna, item, dan atributnya secara menyeluruh. Dengan memanfaatkan KG, KGAT mampu menyediakan informasi kontekstual yang lebih kaya. Sementara itu, mekanisme *attention* memungkinkan model untuk memberi bobot pada setiap hubungan dan simpul tetangga berdasarkan tingkat kepentingannya, sehingga menghasilkan rekomendasi yang lebih akurat (Zhu dkk., 2020).



Gambar 3. Framework KGAT dalam Wang dkk. (2019).

Dalam penelitian yang dilakukan oleh Wang dkk., (2019), KGAT terbagi dalam tiga komponen utama, yaitu:

1. *Embedding Layer*, berfungsi untuk merepresentasikan setiap *nodes* (*user*, *item*, atau *entity*) dalam *collaborative knowledge graph* (CKG) ke dalam bentuk *vector embedding* dengan menggunakan *TransR*. Metode ini mempelajari representasi vektor untuk entitas dan relasi dengan cara mempertahankan prinsip translasi pada struktur data *triple* (*head, relation, tail*). Dengan demikian, *embedding* yang dihasilkan dapat menangkap struktur graf secara detail, menanamkan informasi relasi dari knowledge graph langsung ke dalam *node representation*.
2. *Attentive Embedding Propagation Layer*, adalah mekanisme utama dari model KGAT. Pada tahap ini, model akan terus-menerus memperbarui representasi vektor setiap *nodes* dengan mengambil informasi dari *nodes* di sekitarnya (bisa berupa *user*, *item*, atau *attribute*). Proses ini memanfaatkan mekanisme *attention* untuk memberi bobot setiap tetangga (*neighbor)* secara berbeda-beda, berdasarkan relevansinya. Melalui cara ini, representasi embedding secara progresif dapat menangkap hubungan yang rumit dan bertingkat dalam struktur graf, sekaligus memungkinkan model untuk fokus pada informasi dari *neighbor* yang paling berguna untuk menghasilkan rekomendasi.
3. *Prediction Layer*, bekerja dengan menggabungkan representasi vektor *user* dan *item* yang diperoleh dari seluruh lapisan *embedding propagation*. Setiap lapisan propagasi menangkap tingkat hubungan yang berbeda antar-simpul, sehingga model mengumpulkan semua representasi ini untuk menciptakan profil yang kaya dan mendetail bagi *user* maupun *item*. Selanjutnya, model menghitung 'skor kecocokan' (*matching score*) antara keduanya dengan mengukur tingkat kemiripan *embedding* gabungan mereka, umumnya melalui operasi *dot product*. Skor ini merepresentasikan prediksi probabilitas bahwa *user* akan menyukai atau berinteraksi dengan item tersebut.

KGAT telah diimplementasikan dalam berbagai aplikasi, yang membuktikan fleksibilitas dan efektivitasnya. Di bidang kesehatan, KGAT digunakan untuk memprediksi interaksi antar-obat dengan menangkap korelasi kompleks di antara obat dan gen, yang berkontribusi pada peningkatan keamanan dan efikasi penemuan obat (Kundi dkk., 2024). Di sektor warisan budaya, KGAT memungkinkan *embedding* data ontologi untuk merekomendasikan item warisan yang serupa dan mengatasi masalah *cold-start* secara efisien (Kim dkk., 2024). Selain itu, KGAT juga diterapkan dalam konteks business-to-business (B2B) untuk merekomendasikan calon pelanggan kepada penjual, dengan memberikan rekomendasi yang dapat dijelaskan (*explainable*) guna mendukung strategi pemasaran dan penjualan (Cho dkk., 2023). Berbagai implementasi ini membuktikan kemampuan KGAT dalam mengintegrasikan informasi dari banyak sumber, mengatasi kelangkaan data, dan menyediakan rekomendasi berkualitas tinggi yang dapat diinterpretasikan di berbagai bidang.

1.2.5 Mekanisme Atensi pada KGAT

Mekanisme atensi adalah konsep dalam arsitektur saraf yang memungkinkan model untuk fokus secara selektif pada bagian data input yang paling relevan saat membuat prediksi. Mekanisme ini memberikan "bobot atensi" yang berbeda-beda pada setiap komponen input, sehingga meningkatkan kemampuan interpretasi dan kinerja model, terutama pada tugas yang melibatkan data kompleks. Dalam konteks graf seperti Knowledge Graph (KG), atensi membantu mengidentifikasi node (simpul) tetangga dan hubungan mana yang paling berpengaruh terhadap representasi target, yang pada akhirnya meningkatkan akurasi prediksi. (S. Zhang & Xie, 2020)

Dalam *Knowledge Graph Attention Network* (KGAT), mekanisme atensi digunakan untuk memodelkan tingkat kepentingan hubungan dan entitas saat informasi menyebar di seluruh graf. KGAT menghitung skor atensi antara sebuah entitas dan tetangganya, yang memungkinkan model untuk memberikan bobot lebih besar pada *node* yang berpengaruh sekaligus mengurangi dampak dari *node* yang kurang relevan (*noise*) (Z. Zhang dkk., 2020). Pembobotan selektif ini meningkatkan *representation learning* dengan menangkap fitur semantik (makna) dan struktural entitas dalam graf. Selain itu, mekanisme atensi yang sadar akan relasi dapat membedakan tingkat signifikansi dari setiap jenis hubungan yang berbeda.

Dalam penelitian Wang dkk. (2019), proses atensi di KGAT dapat dipecah menjadi tiga langkah:

1. Menghitung Skor Kepentingan Relasional (Skor Mentah)
   Langkah pertama adalah menghitung skor afinitas (kedekatan) mentah untuk setiap koneksi (h, r, t), di mana h adalah *head* (entitas target), t adalah *tail* (entitas tetangga), dan r adalah relasi di antara keduanya. Skor ini dihitung menggunakan formula:

$$\pi(h, r, t) = (W_r e_t)^\top \tanh(W_r e_h + e_r) \tag{1.7}$$

Secara sederhana, formula ini mengukur seberapa cocok entitas h dan t jika dilihat dari "kacamata" relasi r. W_r adalah matriks yang mempelajari cara memandang entitas dalam konteks relasi r tersebut, sementara e_h, e_r, dan e_t adalah representasi vektor (*embedding*) dari masing-masing komponen.

2. Normalisasi Skor (Menjadi Persentase Kepentingan)
   Skor mentah dari langkah 1 sulit untuk dibandingkan. Oleh karena itu, skor-skor tersebut dinormalisasi menggunakan fungsi *softmax*. Proses ini mengubah skor mentah menjadi bobot atensi final yang nilainya berkisar antara 0 dan 1. Jika dijumlahkan, total bobot atensi dari semua tetangga sebuah node akan menjadi 1 (atau 100%).
3. Agregasi Berbobot (Pengumpulan Informasi)
   Langkah terakhir adalah memperbarui representasi entitas h dengan mengumpulkan informasi dari para tetangganya. Di sinilah bobot atensi tadi berperan. Representasi baru dari tetangga (e_{N_h}) dihitung dengan menjumlahkan representasi semua tetangga (e_t), namun masing-masing dikalikan terlebih dahulu dengan bobot atensinya $\pi(h,r,t)$ yang sudah dinormalisasi:

$$e_{N_h} = \sum_{(h,r,t)\in N_h} \pi(h,r,t)_{\text{normalized}} \cdot e_t \tag{1.8}$$

   Tetangga dengan skor atensi tinggi (misalnya 0,7 atau 70%) akan memberikan kontribusi fitur yang jauh lebih besar pada representasi baru h, dibandingkan tetangga dengan skor atensi rendah (misalnya 0,05 atau 5%). Dengan cara ini, KGAT secara cerdas memprioritaskan informasi yang paling relevan untuk menghasilkan rekomendasi yang lebih akurat.

Setelah proses propagasi informasi melalui L lapis selesai, KGAT tidak hanya menggunakan embedding dari lapis terakhir. Sebaliknya, model ini menerapkan mekanisme agregasi lapis (layer-aggregation) untuk menggabungkan representasi dari semua lapis, mulai dari embedding awal (lapis 0) hingga lapis terakhir (lapis L).

Menurut Wang dkk. (2019), representasi dari setiap lapis digabungkan (*concatenation*) menjadi satu vektor final yang komprehensif untuk pengguna ($\boldsymbol{e_u^*}$) dan *item* ($\boldsymbol{e_i^*}$), seperti ditunjukkan pada Persamaan (1.9):

$$e_u^* = e_u^{(0)} \,\|\, \cdots \,\|\, e_u^{(L)} \quad \text{dan} \quad e_i^* = e_i^{(0)} \,\|\, \cdots \,\|\, e_i^{(L)} \tag{1.9}$$

Di mana $e^{(0)}$ adalah embedding awal dan $\|$ adalah operasi concatenation. Skor prediksi akhir ($\widehat{y(u,i)}$) kemudian dihitung dengan mengambil *inner product* (produk skalar) dari kedua vektor representasi final tersebut (Persamaan 1.10)

$$\widehat{y(u,i)} = {e_u^*}^{\top} e_i^* \tag{1.10}$$

Nilai $\widehat{y(u,i)}$ ini merepresentasikan prediksi kecocokan antara pengguna u dan item i.

1.2.6 Recbole

RecBole merupakan pustaka (*library*) sumber terbuka (*open-source*) yang populer, dibuat dengan tujuan untuk melakukan standardisasi, pengujian (*benchmarking*), dan percepatan riset di bidang sistem rekomendasi. *Library* ini dikembangkan menggunakan PyTorch dan menyediakan kumpulan model, *dataset*, serta perangkat (tools) yang lengkap untuk kebutuhan sistem rekomendasi (Zhao dkk., 2021).

Dalam dokumentasi resminya (RecBole Team, 2025) dan penelitian oleh Zhao dkk., (2021), *library* ini menawarkan beberapa fitur utama, di antaranya:

1. Unified Framework
   RecBole menyediakan *interface* yang seragam untuk mengimplementasikan, melatih, dan mengevaluasi berbagai algoritma rekomendasi, mulai dari filtrasi kolaboratif tradisional hingga model pembelajaran mendalam yang canggih.
2. Extensive Model and Dataset Support
   Menyediakan beragam koleksi model dan *dataset*, yang mencakup algoritma untuk rekomendasi umum, sekuensial (*sequential*), sadar-konteks (*context-aware*), dan berbasis pengetahuan (*knowledge-aware*). Secara khusus, *library* ini menyertakan implementasi KGAT, yang memudahkan dalam *training, evaluating,* hingga menyajikan hasil prediksi. RecBole juga menyediakan versi *processed* dari banyak *benchmark dataset* yang mempermudah proses perbandingan antara suatu metode dengan metode *baseline* lainnya.
3. Grid Search & Hyperparameter Tuning
   RecBole memiliki fitur bawaan untuk melakukan pencarian *hyperparameter* dengan metode *grid search* (pencarian menyeluruh) melalui modul *HyperTuning*. Pengguna cukup menentukan rentang nilai yang ingin diuji pada *file* konfigurasi. Selanjutnya, RecBole akan secara otomatis menguji semua kemungkinan kombinasi untuk menemukan pengaturan terbaik.

1.2.7 Streamlit

Streamlit adalah *framework open-source* Python yang dirancang untuk menyederhanakan pembuatan aplikasi visualisasi data yang interaktif. *Library* ini memungkinkan *developer* untuk mengubah skrip Python menjadi dasbor dan aplikasi berbasis data dengan mudah. Kemudahan penggunaannya dan kemampuannya untuk berintegrasi dengan alur kerja *machine learning* membuatnya diadopsi di berbagai bidang, terutama yang memerlukan *interface* ramah pengguna (user-friendly) dan eksplorasi data secara *real-time*.

Beberapa kelebihan menggunakan Streamlit:

1. *User-Friendly Interface* untuk Model *Machine Learning*

Streamlit mempermudah tahapan deployment atau penerapan model *machine learning* dan *deep learning*. Dengan Streamlit, para *developer* bisa mengubah skrip Python mereka menjadi sebuah aplikasi interaktif hanya dengan sedikit *coding*. Hal ini membuat proses untuk menguji, mendemonstrasikan, dan membagikan hasil dari sistem prediktif menjadi jauh lebih mudah (Bandal, 2024).

2. Visualisasi Data yang Interaktif
   Salah satu fitur unggulan Streamlit adalah kemampuannya untuk membuat visualisasi yang dinamis dan interaktif dengan memanfaatkan *library* seperti Matplotlib, Seaborn, dan Plotly.
3. Waktu Pengembangan yang Cepat
   Framework ini memungkinkan pembuatan prototipe aplikasi secara cepat, sehingga secara signifikan mengurangi waktu yang dibutuhkan dari tahap pengembangan model hingga *deployment*. Developer dapat mengubah kode dari Jupyter Notebook menjadi sebuah demo berbasis web yang fungsional hanya dalam hitungan jam. Hal ini menjadikan Streamlit dapat diimplementasikan di bidang-bidang yang bersifat eksperimental seperti pemrosesan citra (*image processing*) maupun sistem rekomendasi (Chaurase dkk., 2024).

1.3 Tujuan dan Manfaat

1.3.1 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Mengimplementasikan *Knowledge Graph Attention Network* pada sistem rekomendasi pemeliharaan spare part kendaraan roda empat.
2. Memanfaatkan sistem informasi untuk menyajikan rekomendasi pemeliharaan spare part.

1.3.2 Manfaat Penelitian

Manfaat dari penelitian ini adalah memberikan dukungan bagi tim *aftersales* atau *service advisor* bengkel dalam meningkatkan kualitas layanan perawatan kendaraan. Sistem rekomendasi berbasis *Knowledge Graph Attention Network* (KGAT) dapat menjadi sumber informasi tambahan untuk mengidentifikasi *spare part* yang memiliki kemungkinan tinggi mengalami kerusakan atau membutuhkan penggantian. Dengan adanya rekomendasi ini, mekanik maupun staf *aftersales* dapat melakukan pengecekan lebih dini terhadap *spare part* yang disarankan, sehingga potensi kerusakan dapat diminimalisir dan kepuasan pelanggan meningkat. Selain itu, rekomendasi ini juga dapat dimanfaatkan sebagai peluang *upselling* secara lebih terarah, karena tim *aftersales* memiliki dasar data yang kuat dalam menawarkan penggantian *spare part* kepada pelanggan, sehingga strategi penjualan menjadi lebih efektif sekaligus tetap berorientasi pada kebutuhan nyata kendaraan.

BAB II

METODE PENELITIAN

2.1 Tempat dan Waktu

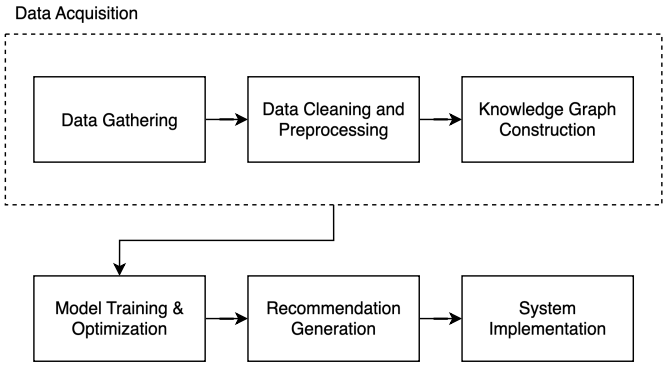
Penelitian ini dilaksanakan dalam kurun waktu enam bulan, terhitung sejak proposal penelitian disetujui pada bulan Maret 2025 hingga selesai pada bulan Agustus 2025. Kegiatan penelitian, mulai dari studi literatur, implementasi model, hingga analisis hasil, utamanya bertempat di Laboratorium CCIS (*Cloud Computing and Information System*), Departemen Teknik Informatika, Fakultas Teknik, Universitas Hasanuddin.

2.2 Instrumen Penelitian

Instrumen yang digunakan dalam penelitian ini dirincikan sebagai berikut:

1. Perangkat lunak:
    a. Windows 10
    b. Visual Studio Code
    c. Navicat
    d. Arc Browser
    e. Microsoft Word
    f. Microsoft Excel
    g. Microsoft PowerPoint
    h. Zotero
2. Perangkat keras:
    a. Prosesor (CPU): 11th Gen Intel® Core™ i9-11900F
    b. Memori (RAM): 32 GB
    c. Kartu Grafis (GPU): NVIDIA® GeForce® RTX 3070
3. Bahasa pemrograman:
    a. Python 3.10
4. Algoritma:
    a. Knowledge Graph Attention Network (KGAT)
5. Library:
    a. Pandas v2.0.3, digunakan untuk menampilkan dan memanipulasi data.
    b. Recbole v1.2.1, digunakan sebagai framework *training, evaluating,* dan *predicting* untuk model KGAT.
    c. Streamlit V1.40.1, digunakan untuk menyajikan hasil prediksi ke dalam bentuk aplikasi web.

2.3 Tahapan Penelitian



Gambar 4. Tahapan penelitian.

Fase pertama dari penelitian, Akuisisi Data, diawali dengan pengumpulan data riwayat servis mentah dari *Dealer Management System* (DMS) Toyota. Data tersebut kemudian melalui proses pembersihan dan *preprocessing* untuk mengatasi data anomali, duplikasi, dan inkonsistensi, sekaligus diperkaya dengan informasi tambahan seperti klasifikasi subsistem *spare part* dan kategori usia kendaraan. Akhir dari fase ini adalah konstruksi *Knowledge Graph* (KG) yang memetakan relasi kompleks antar entitas (kendaraan, sparepart, work order, dll.) dan diformat menjadi *atomic dataset* yang siap digunakan oleh RecBole. Fase kedua adalah Pelatihan & Optimisasi Model, di mana model *Knowledge Graph Attention Network* (KGAT) dilatih menggunakan *dataset* KG yang telah dibuat. Proses ini dilanjutkan dengan optimisasi melalui *grid search* untuk menemukan konfigurasi parameter terbaik yang menghasilkan performa prediksi paling akurat. Fase ketiga, Generasi Rekomendasi, memanfaatkan model yang terbaik untuk menghasilkan daftar rekomendasi *spare part* beserta skor prediksinya untuk setiap kendaraan. Terakhir, pada fase keempat, Implementasi Sistem, hasil rekomendasi tersebut diintegrasikan ke dalam sebuah aplikasi web interaktif. Sistem ini memungkinkan visualisasi riwayat servis dan menampilkan 10 rekomendasi *spare part* teratas setelah pengguna memasukkan nomor rangka, di mana rekomendasi tersebut diklasifikasikan berdasarkan tingkat urgensi ("Segera Cek", "Perlu Perhatian", "Opsional") dan dapat difilter untuk mengecualikan komponen servis berkala.

2.4 Perancangan dan Implementasi Sistem

2.4.1 Data Collection

Tahap awal dalam perancangan sistem rekomendasi ini adalah pengumpulan data. Data yang digunakan dalam penelitian ini merupakan data sekunder yang diperoleh dari d *Dealer Management System* (DMS) milik Kalla Toyota. DMS sendiri merupakan sistem manajemen terintegrasi yang mencatat seluruh aktivitas transaksi di bengkel,

termasuk riwayat servis, detail *spare part* yang digunakan, serta identitas kendaraan pelanggan.

Proses pengumpulan data dilakukan dengan langkah-langkah sebagai berikut:

a. Ekstraksi Data: Data historis transaksi servis dari rentang waktu 2017 hingga 2024 diekstraksi dari DMS menggunakan perangkat lunak Navicat. Data mentah yang relevan untuk penelitian ini tersusun dalam tiga tabel utama, yaitu:
   - *Workorder*: berisi informasi umum perkerjaan servis seperti metadata servis, informasi kendaraan, dan informasi pelanggan.
   - *Partorder*: berisi detail *spare part* yang dipesan/diganti untuk setiap *workorder*.
   - *Joborder*: berisi detail jenis pekerjaan atau jasa yang dilakukan.

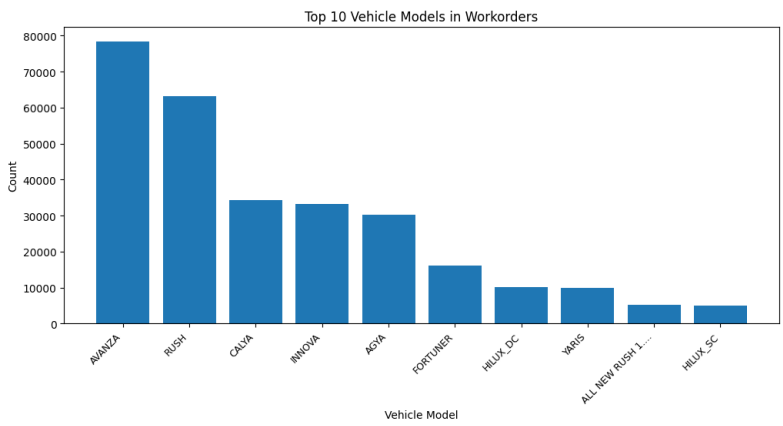
b. Ekspor Data: Seluruh data dari ketiga tabel tersebut diekspor ke dalam format file *Comma-Separated Values* (CSV). Pemilihan format ini bertujuan untuk memastikan kompatibilitas dan kemudahan dalam pemrosesan data pada tahap selanjutnya.

c. Pemuatan Data: File CSV yang telah diekspor kemudian dimuat (*import*) ke dalam lingkungan pengembangan Python (*notebook*) untuk persiapan tahap pra-pemrosesan data.

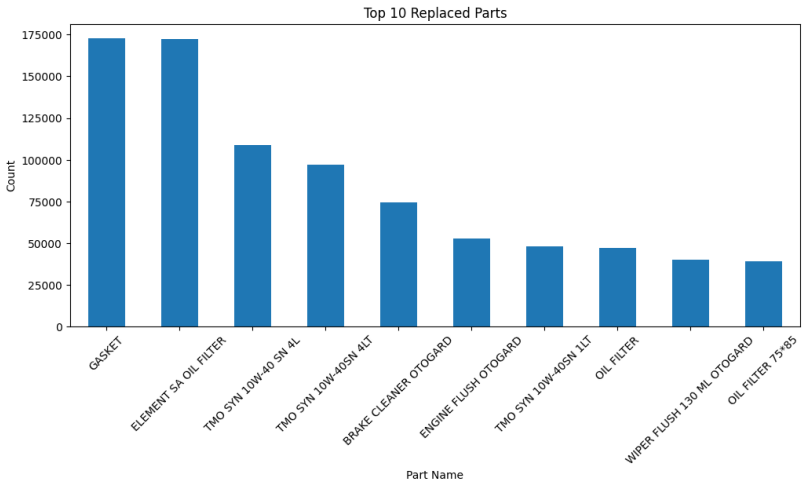
2.4.2 Data Preprocessing and EDA

Tahap ini merupakan langkah fundamental untuk mempersiapkan data mentah dari sistem bengkel agar berkualitas, relevan, dan siap digunakan untuk konstruksi *knowledge graph* serta pelatihan model. Proses ini terbagi menjadi dua kegiatan utama:

1. *Exploratory Data Analysis* (EDA)



Gambar 5. Distribusi 10 model kendaraan dengan frekuensi servis tertinggi.



Gambar 6. Distribusi 10 spare part dengan frekuensi penggantian tertinggi.

Tujuan dari tahap ini adalah untuk mengidentifikasi pola, distribusi, anomali, serta karakteristik umum dari data riwayat servis. Beberapa hal yang dilakukan pada tahap ini meliputi:

a. Identifikasi Statistik Deskriptif: Menghitung metrik dasar untuk memahami skala *dataset*, seperti jumlah total *work order*, jumlah kendaraan unik, jumlah *spare part* unik, dan rentang waktu data yang digunakan.
b. Analisis Distribusi Data: Menganalisis distribusi data kunci, seperti:
    - Model kendaraan yang paling sering melakukan servis seperti pada Gambar 5.
    - *Spare part* yang paling sering diganti di seluruh riwayat servis seperti pada Gambar 6.
    - Kendaraan (berdasarkan nomor rangka) dengan frekuensi kunjungan atau jumlah penggantian *spare part* tertinggi.

2. Pra-pemrosesan Data (*Data Preprocessing*)

Berdasarkan temuan dari tahap EDA, proses pra-pemrosesan dilakukan untuk memastikan integritas dan konsistensi data. Langkah-langkah yang diimplementasikan adalah sebagai berikut:

a. Pembersihan dan Standarisasi Data:
    - Normalisasi Nama Model Kendaraan: Membersihkan kolom model kendaraan dari informasi yang tidak relevan seperti kode warna atau varian promosi (misalnya, " FORTUNER 4x2 G M/T DIESEL SPORTIVO TRD VNT [NEW] DARK STEEL MICA METALLIC" menjadi “FORTUNER 4x2 G M/T DIESEL SPORTIVO TRD VNT”). Tujuannya adalah untuk mengelompokkan kendaraan dengan model dasar yang sama.

- Ekstraksi Model Dasar Kendaraan: Membuat kolom baru base_vehicle_model yang berisi nama model utama (misalnya, "FORTUNER") dari nama model yang panjang dan detail.
- Pembersihan nomor *work order*: Menghapus karakter yang tidak perlu (seperti spasi atau koma di awal) dan memastikan format yang konsisten.

b. Filtrasi Data Berdasarkan Relevansi:
- Filter Berdasarkan Panjang ID: Menyaring data *work order* dan nomor rangka (*frame serial number*) yang tidak memenuhi standar panjang karakter minimum (10 karakter) untuk menghindari data yang tidak valid atau salah input.
- Filter Jenis Pekerjaan (Job Order): Mengecualikan *work order* yang tidak terkait langsung dengan servis pelanggan, seperti pekerjaan internal bengkel ('PRT') dan servis pra-pengiriman ('PDS' atau 'SERVICE PDS').
- Filter Status Transaksi: Menghapus data transaksi (*work order* dan *part order*) dengan status "Cancelled" atau "Error".

c. Penanganan Data Duplikat:
- Melakukan identifikasi dan penghapusan data duplikat pada level *work order*, *part order*, dan *job order* untuk memastikan setiap transaksi servis bersifat unik dan mencegah bias dalam pemodelan.

2.4.3 Data Enrichment

Tahap *Data Enrichment* bertujuan untuk memperkaya dan meningkatkan kualitas *dataset* dengan menambahkan informasi kontekstual baru yang tidak tersedia secara eksplisit pada data mentah. Proses ini menghasilkan *knowledge graph* yang lebih informatif sehingga dapat meningkatkan akurasi dan relevansi dari sistem rekomendasi yang dikembangkan. Proses pengayaan data ini mencakup beberapa langkah utama sebagai berikut:

1. Standardisasi dan Kategorisasi Spare Part

Tabel 1. Standardisasi dan kategorisasi spare part.

Nama Mentah	Nama Standarisasi	Subsistem
DISC CLUTCH A/S 1.5	DISC CLUTCH	Brake System
SHOCKABSORBER FR LH	SHOCK ABSORBER	Chassis
SHOE KIT FR DRUM BRK	DRUM BRAKE SHOE	Brake System

BEARING C/SHAFT STD MARK2	CRANKSHAFT BEARING	Engine

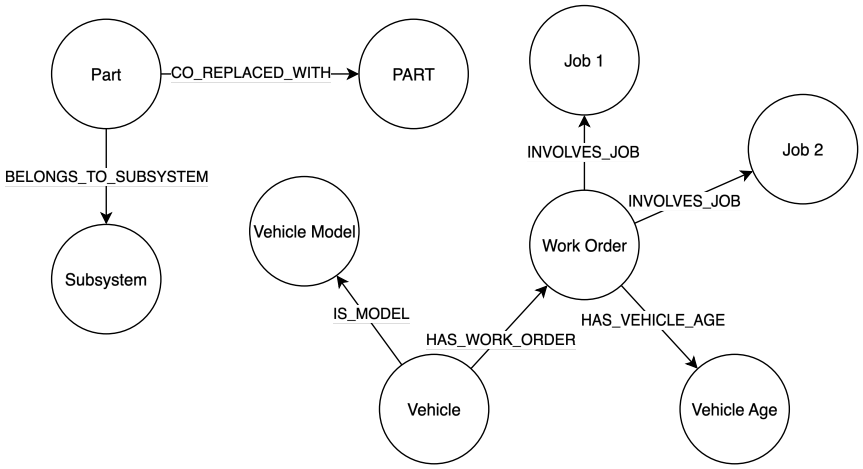
Langkah pertama adalah membuat daftar entitas *spare part* yang unik. Proses ini melibatkan beberapa proses:

- Standardisasi Nama: Nama-nama *spare part* yang beragam dan seringkali mengandung kode teknis disederhanakan menjadi istilah yang lebih umum dan konsisten seperti pada tabel 1. Sebagai contoh, “BRAKE PAD KIT FR” distandarisasi menjadi “Brake Pad”.
- Penyaringan Entitas: *Spare part* yang tidak relevan dengan komponen teknis atau mekanis kendaraan, seperti aksesoris, stiker, atau item non-mesin lainnya, dihilangkan dari daftar.
- Kategorisasi Sub-sistem: Setiap *spare part* yang telah distandardisasi kemudian diklasifikasikan berdasarkan sub-sistem fungsionalnya pada kendaraan (misalnya: Sistem Pengereman, Sistem Suspensi, Mesin, Transmisi).

2. Penambahan Atribut Usia Kendaraan
   Untuk menambahkan dimensi waktu yang kontekstual pada setiap transaksi servis, sebuah atribut baru yaitu kategori usia kendaraan dibuat. Usia kendaraan dihitung berdasarkan selisih antara tanggal *work order* dengan tanggal pembelian kendaraan. Hasil perhitungan tersebut kemudian dikelompokkan ke dalam lima kategori rentang usia: 0-2 tahun, 2-4 tahun, 4-6 tahun, 6-8 tahun, dan >8 tahun.
3. Identifikasi Pola Penggantian Bersamaan (*Co-Replaced Parts*)
   Untuk memodelkan relasi implisit antar *spare part*, dilakukan analisis untuk mengidentifikasi pasangan komponen yang sering diganti secara bersamaan dalam satu sesi servis. Relasi ini diukur menggunakan metrik frekuensi dan lift. Pasangan *spare part* yang dianggap memiliki hubungan kuat dan dimasukkan ke dalam data relasi adalah yang memenuhi salah satu dari dua kondisi berikut:
   - Kondisi 1: Lift ≥ 500 dan Frekuensi > 50
   - Kondisi 2: Lift ≥ 1000 dan Frekuensi > 10
     Hasil dari analisis ini menghasilkan sebuah tabel baru yang secara eksplisit mendefinisikan relasi "sering diganti bersama" antar entitas *spare part* dalam *knowledge graph*.
4. Seleksi Fitur Relevan
   Tahap terakhir adalah melakukan seleksi kolom (fitur) dari tabel-tabel utama seperti *Work Order* (WO), *Part Order* (PO), dan *Job Order* (JO). Hanya kolom-kolom yang esensial untuk konstruksi knowledge graph (seperti ID kendaraan, ID *spare part*, ID pekerjaan, dan tanggal transaksi) yang dipertahankan. Langkah ini bertujuan untuk mereduksi kompleksitas data dan memastikan model fokus pada informasi yang paling signifikan.

2.4.4 Knowledge Graph Construction

Tahapan ini bertujuan untuk menstrukturkan data dari riwayat servis kendaraan ke dalam format KG. KG berfungsi untuk merepresentasikan entitas-entitas data (seperti *spare part* dan kendaraan) sebagai *node* dan hubungan antar entitas tersebut sebagai *edge*.



Gambar 7. Entitas dan relasi dalam knowledge graph.

Proses konstruksi KG juga merujuk pada model yang telah dibuat oleh Tang dkk., (2023), yang dibagi menjadi dua langkah utama: perancangan skema (definisi entitas dan relasi) dan implementasi pembuatan *triplet*.

1. Perancangan Skema *Knowledge Graph*
   Skema KG pada Gambar 7, dirancang untuk memetakan elemen-elemen kunci dari data servis. Skema ini terdiri dari entitas (*node*) dan relasi (*edge*) yang didefinisikan sebagai berikut:
   a. Entitas (*Nodes*)
      Entitas merepresentasikan objek-objek utama dalam domain penelitian. Entitas yang didefinisikan dalam sistem ini adalah:
      - Part: Merepresentasikan komponen atau *spare part* yang digunakan dalam servis, contohnya SHOCKBREAKER dan GASKET, DISC BRAKE.
      - Subsystem: Kategori mengelompokkan sparepart berdasarkan sistem kendaraan, seperti Transmission, Brake System, dan Engine.
      - Vehicle Model: Model dasar dari sebuah kendaraan, contohnya AVANZA dan INNOVA.
      - Job Type: Jenis pekerjaan spesifik yang dilakukan selama servis, seperti ENGINE TUNE UP dan 40.000 KM SERVICE.
      - Vehicle: Kendaraan unik yang diidentifikasi melalui nomor rangka (Frame Serial No), contohnya MHFLA8EM7M1316841.

- Vehicle Age Category: Kategori usia kendaraan pada saat servis dilakukan, seperti >8 years, 1 - 2 years, dan 2 - 4 years.
- Work Order: ID unik yang merepresentasikan satu perintah kerja servis, contohnya 20302/SWO/21/06/00506.

b. Relasi (Edges)
Relasi mendefinisikan hubungan semantik antar entitas. Berikut adalah relasi yang dirancang untuk menghubungkan node-node tersebut dalam format (Head Entity) -[RELATION]-> (Tail Entity):

- (Part) -[CO_REPLACED_WITH]-> (Part):
  Menghubungkan dua sparepart yang memiliki frekuensi penggantian bersamaan yang tinggi.
- (Part) -[BELONGS_TO_SUBSYSTEM]-> (Subsystem):
  Menghubungan sebuah sparepart dengan kategori sistem fungsionalnya.
- (Vehicle) -[IS_MODEL]-> (Vehicle Model):
  Menghubungkan kendaraan unik dengan model dasarnya.
- (Work Order) -[INVOLVES_JOB]-> (Job Type):
  Menghubungkan perintah kerja dengan jenis pekerjaan yang dilakukan.
- (Vehicle) -[HAS_WORK_ORDER]-> (Work Order):
  Menghubungkan kendaraan dengan riwayat perintah kerja yang pernah dijalani.
- (Work Order) -[HAS_VEHICLE_AGE]-> (Vehicle Age Category):
  Menghubungkan perintah kerja dengan kategori usia kendaraan pada saat itu.

2. Implementasi Konstruksi Triplet

Tabel 2. Format head, relation, dan tail pada KG

Head	Relation	Tail
MHFGW8ABCD123	IS_MODEL	INNOVA
11223/SWO/44/55/00987	HAS_VEHICLE_AGE	2-4 years
11223/SWO/44/55/00987	INVOLVES_JOB	[X] MOTOR FAN RADIATOR
CRANKSHAFT PULLEY	BELONGS_TO_SUB SYSTEM	Engine

Setelah skema didefinisikan, KG dibangun dengan cara mengekstrak *triplet* (*head, relation, tail*) dari *dataset* yang telah melalui tahap pra-pemrosesan dan pengayaan data. Proses ini diimplementasikan menggunakan skrip

Python untuk setiap jenis relasi, yang hasilnya akan membentuk format seperti pada Tabel 2.

2.4.5 Dataset Preparation for KGAT

Pada tahap ini, dataset yang telah melalui proses pra-pemrosesan dan pengayaan (*enrichment*) disiapkan ke dalam format spesifik yang dibutuhkan oleh library RecBole untuk dapat melatih model KGAT. RecBole menggunakan struktur yang disebut *atomic files*, yang terbagi menjadi:

Tabel 3. Daftar file atomic dataset.

Nama File	Kolom	Deskripsi	Contoh Data
dataset.inter	user_id	ID unik kendaraan (No. Rangka)	MHFX12345ABC
	item_id	Nama standar *spare part*	DISC BRAKE
	timestamp	Waktu penggantian (Unix timestamp)	1672531200
dataset.kg	head_id	Entitas awal	DISC BRAKE
	relation_id	Jenis relasi	BELONGS_TO_SUBSYSTEM
	tail_id	Entitas tujuan	Brake System
dataset.link	Item_id	ID dari file .inter	CLUTCH BEARING
	entity_id	ID dari file .kg	CLUTCH BEARING

1. Pembuatan File Interaksi (.inter)
   File ini berfungsi untuk merekam interaksi historis antara pengguna dan item. Dalam konteks penelitian ini, "pengguna" direpresentasikan oleh kendaraan dan "item" oleh spare part yang diganti. Setiap baris dalam file ini mencatat satu peristiwa penggantian spare part dan disusun dalam format [user_id] [item_id] [timestamp].
2. Pembuatan File Knowledge Graph (.kg)
   File ini berisi seluruh struktur KG yang telah dirancang pada tahap sebelumnya. Setiap baris data merepresentasikan satu triplet pengetahuan dengan format [head_id] [relation_id] [tail_id].
3. Pembuatan File Penghubung (.link)

File ini memiliki peran krusial sebagai "jembatan" yang menghubungkan ID item pada file interaksi (.inter) dengan ID entitas yang relevan di dalam *Knowledge Graph* (.kg). Hal ini memastikan model memahami bahwa, misalnya, spare part "Filter Oli" yang tercatat dalam interaksi adalah entitas yang sama dengan "Filter Oli" di dalam KG. File ini disusun dalam format [item_id] [entity_id].

2.4.6 Model Training (KGAT Implementation)

Implementasi model KGAT dalam penelitian ini difokuskan pada proses pelatihan yang terintegrasi dengan optimasi hiperparameter untuk menemukan konfigurasi arsitektur terbaik. Untuk mencapai tujuan ini, metode *Grid Search* diterapkan guna menguji secara sistematis berbagai kombinasi hiperparameter dan mengevaluasinya berdasarkan metrik yang telah ditentukan.

Untuk memastikan setiap model kandidat dievaluasi secara adil dan konsisten, sebuah skenario evaluasi yang terstandardisasi dirancang. Konfigurasi evaluasi ini digunakan untuk setiap iterasi dalam proses Grid Search. Rincian lengkap mengenai parameter evaluasi disajikan pada Tabel 4.

Tabel 4. Parameter konfigurasi evaluasi model.

Parameter	Nilai / Format	Keterangan
split	{'RS': [0.8, 0.1, 0.1]}	Menentukan proporsi pembagian dataset secara acak (*Random Split*), yaitu 80% untuk data pelatihan (*training*), 10% untuk validasi (*validation*), dan 10% untuk pengujian (*testing*).
group_by	'user'	Mengatur bahwa proses evaluasi dilakukan berdasarkan setiap pengguna (dalam konteks ini: kendaraan).
metrics	['Recall', 'MRR', 'NDCG', 'Hit', 'Precision']	Daftar metrik yang digunakan untuk menilai performa sistem rekomendasi.
topk	[10]	Menentukan jumlah rekomendasi teratas yang dievaluasi, yaitu 10 item terbaik (*Top-10 Recommendation*).

valid_metric	MRR@10	Metrik utama untuk memilih model terbaik adalah *Mean Reciprocal Rank* pada 10 item teratas.

Proses *Grid Search* melibatkan pengujian beberapa set nilai untuk hiperparameter kunci yang paling berpengaruh pada performa model KGAT, seperti laju pembelajaran (*learning rate*), arsitektur lapisan (*layers*), dan bobot regularisasi (*regularization weight*). Kombinasi *hyperparameter* yang diuji dalam penelitian ini dirangkum pada Tabel 5.

Tabel 5. Kombinasi hiperparameter untuk grid search.

Parameter	Nilai yang Diuji	Keterangan
learning_rate	[0.001, 0.0005]	Ukuran vektor representasi entitas seperti kendaraan dan sparepart.
layers	['[64,32,16]', '[128,64,32]']	Konfigurasi lapisan model.
reg_weight	[1e-5, 1e-6]	Bobot regularisasi untuk menjaga kestabilan dan mencegah parameter terlalu besar.

Berdasarkan parameter yang diuji pada Tabel 5, terdapat total 8 kombinasi unik (2 nilai learning_rate × 2 konfigurasi layers × 2 nilai reg_weight). Setiap kombinasi akan dilatih menggunakan 80% data *training* dan dievaluasi pada 10% data *validation*. Model dengan performa terbaik berdasarkan metrik MRR@10 akan dipilih sebagai model final dan diuji kembali performanya menggunakan 10% sisa data testing untuk mendapatkan hasil evaluasi akhir.

2.4.7 Recommendation Generation

Pada tahap ini, skor mentah dari model diubah menjadi rekomendasi yang siap pakai melalui langkah-langkah berikut:

1. Pembuatan Skor Prediksi: Model KGAT yang diimplementasikan menggunakan *framework* RecBole menerima input nomor rangka untuk menghasilkan skor prediksi bagi setiap *spare part*. Skor ini secara fundamental dihitung dengan mengukur tingkat "kecocokan" atau "kedekatan" (*similarity*) antara representasi vektor kendaraan dan vektor *spare part*.
2. Penyaringan Hasil: Skor tersebut kemudian disaring untuk menyajikan dua mode rekomendasi top-10 (sepuluh teratas):

- Rekomendasi Penuh: Daftar 10 *spare part* dengan skor tertinggi.
- Rekomendasi Non-Servis Berkala: Daftar 10 *spare part* teratas di luar komponen servis rutin (seperti oli, filter, dll).

3. Normalisasi Skor: 10 skor teratas dari mode yang dipilih diubah skalanya ke rentang 0-1 menggunakan metode *Min-Max Normalization* untuk mempermudah klasifikasi.
4. Penentuan Prioritas: Skor yang telah dinormalisasi lalu dikategorikan ke dalam tiga tingkat urgensi:
   - Segera Cek (Prioritas Tinggi): Skor > 0.8
   - Perlu Perhatian (Prioritas Sedang): Skor 0.6 - 0.8
   - Opsional (Prioritas Rendah): Skor < 0.6

2.4.8 System Implementation

Untuk menyajikan hasil rekomendasi dari model KGAT, dibangun sebuah aplikasi web prototipe menggunakan *framework* Streamlit. Selain implementasi teknis, dirancang pula sebuah metode evaluasi untuk mengukur persepsi dan penerimaan pengguna terhadap sistem:

1. Implementasi Antarmuka Pengguna (*Web Application*)
   Aplikasi dirancang dengan alur kerja yang sederhana. Pengguna cukup memasukkan nomor rangka kendaraan untuk menampilkan dua informasi utama:
   - Riwayat Servis Kendaraan: Ringkasan dari seluruh pekerjaan perbaikan yang pernah tercatat untuk kendaraan tersebut.
   - Rekomendasi Penggantian *Spare Part*: Daftar 10 *spare part* teratas yang direkomendasikan untuk diganti. Hasil ini disajikan dalam dua mode tampilan (semua *spare part* dan tanpa *part* servis berkala) dan dikelompokkan ke dalam tiga kategori prioritas: Segera Cek, Perlu Perhatian, dan Opsional.
2. Perancangan Evaluasi Pengguna (*User Evaluation Design*)
   Sebagai pelengkap evaluasi kuantitatif, dirancang pula evaluasi kualitatif menggunakan formulir umpan balik berbasis skala Likert (1-5). Formulir ini bertujuan mengukur persepsi pengguna terhadap:
   - Aspek Antarmuka dan Kemudahan Penggunaan (*Usability*).
   - Aspek Kualitas dan Relevansi Rekomendasi.
   - Aspek Kegunaan dan Penerimaan Sistem (*Usefulness* & *Acceptance*).

   Instrumen evaluasi lengkap disajikan pada Lampiran 1.