BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Adapun penelitian terdahulu yang ditampilkan untuk mendukung adanya penelitian *fine-tuning* yang dilakukan sebagai berikut :

Tabel 2. 1 Penelitian Terdahulu

Penelitian Terdahulu					
No	Informasi Jurnal	Model	Hasil	Masalah	*Future Research*
1	Nama Jurnal: ArXiv Judul Jurnal: RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers Penulis: Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, Matthew Richardson[12]	NLP Model	Menghasilkan sebuah model NLP yang dapat melakukan *task Text-to-SQL*	Proses *text-to-sql* yang dilakukan pada penelitian ini terbatas pada fungsi NLP yang masih perlu di-*define* sendiri tanpa ada kecerdasan buatan yang mampu mentranslasikan bahasa secara otomatis	Penelitian berikutnya dapat dilakukan dengan mengeksplorasi pendekatan mengenai *text-to-sql* menggunakan metode yang lebih canggih seperti LLM untuk mendapatkan hasil yang lebih fleksibel
2	Nama Jurnal: ArXiv Judul Jurnal: SQL-PaLM: Improved Large Language	Tidak dijelaskan model LLM secara spesifik yang	Menghasilkan model LLM yang mampu melakukan *task Text-to-SQL* dengan menggunakan	Model LLM yang dihasilkan pada penelitian ini masih membutuhkan jumlah	Penelitian berikutnya dapat difokuskan pada metode *fine-tuning* yang lebih efisien

	Model Adaptation for Text-to-SQL Penulis: Ruoxi Sun, Sercan Ö. Arik, Alex Muzio, Lesly Miculicich, Satya Gundabathula, Pengcheng Yin, Hanjun Dai, Hootan Nakhost, Rajarishi Sinha, Zifeng Wang, Tomas Pfister [11]	digunakan	beberapa *benchmark*	*resources* yang sangat besar sehingga tidak semua *device* dapat mengakses model tersebut	mengingat jumlah *resources* yang digunakan masih besar
3	Nama Jurnal: ArXiv Judul Jurnal: ChatDoctor: A Medical Chat Model Fine-Tuned on a Large Language Model Meta-AI (LLaMA) Using Medical Domain Knowledge Penulis: Gade, Pranav M. Lermen, Simon	Llama 2	Dengan membandingkan ChatDoctor dengan ChatGPT menggunakan BERTScore, nilai Precision yang dimiliki ChatDoctor lebih tinggi dari ChatGPT namun memiliki nilai Recall yang lebih rendah	Terdapat keterbatasan Dimana nilai *Recalling* lebih rendah daripada GPT4	Penelitian berikutnya yang dapat dilakukan adalah menggunakana metode *fine-tuning* yang lebih efisien untuk menghasilkan nilai *recalling* yang lebih tinggi tanpa mengurangi nilai lainnya

	Rogers-Smith, Charlie Ladish, Jeffrey[13]				
4	Nama Jurnal: ArXiv Judul Jurnal: Goat: Fine-tuned LLaMA Outperforms GPT-4 on Arithmetic Tasks Penulis: Liu, Tiedong Low, K. H.[14]	Llama 2	Model Llama-2 yang sudah di *fine-tune* menampilkan hasil perhitungan yang lebih baik dari model GPT 4 dan menampilkan bahwa *fine-tuning* dapat memaksimalkan performa dari LLM yang ada	Model ini terbatas dalam melakukan *task* mengenai aritmetika	Penelitian berikutnya yang dapat dilakukan adalah eksplorasi metode *fine-tuning* yang lebih efektik untuk dapat menjawab *task* aritmetika yang lebih kompleks
5	Nama Jurnal: ArXiv Judul Jurnal : Tamil-Llama: A New Tamil Language Model Based on Llama 2 Penulis : Balachandran, Abhinand[15]	Llama-2	Meningkatkan model Llama-2 untuk dapat menerima Bahasa Tamil sehingga lingkup bahasa yang dimiliki Model tersebut bertambah	Model llama-2 yang sudah di-*training* masih kesulitan untuk menjawab pertanyaan dengan bahasa non inggris	Melakukan pengembangan terhadap bahasa lain yang bisa memberikan opsi bahasa yang lebih luas
6	Nama Jurnal: ArXiv Judul Jurnal: LLaMA-Adapter: Efficient Fine-tuning	Llama	Menciptakan LlaMa-Adapter, sebuah model Llama yang sudah di *fine-tune* yang terbukti	Penelitian ini tidak menjelaskan implementasi terhadap Llama-2 dimana model tersebut	Melakukan metode *fine-tuning* pada Llama-2 dan kemudian melakukan komparasi dengan

	of Language Models with Zero-init Attention Penulis: Zhang, Renrui Han, Jiaming Zhou, Aojun Hu, Xiangfei Yan, Shilin Lu, Pan Li, Hongsheng Gao, Peng Qiao, Y.[17]		memiliki performa yang jauh lebih baik dari model Alpaca	adalah model yang lebih baru daripada Llama	model Llama untuk menemukan perkembangan performa
7	Nama Jurnal: ArXiv Judul Jurnal: LoRA Fine-tuning Efficiently Undoes Safety Training in Llama 2-Chat 70B Penulis: Lermen, Simon Rogers-Smith, Charlie Ladish, Jeffrey[20]	Llama-2	Penelitian ini menerapkan *fine-tuning* untuk menghilangkan *safety training* dari model Llama-2 sehingga model tersebut mampu menjawab pertanyaan yang biasanya dimoderasi	Metode LoRA yang digunakan masih memiliki kekurangan dalam penggunaan *resources* yang masih cukup besar	Melakukan metode *fine-tuning* yang lebih efisien dan efektif dari LoRA sehingga dapat mengurangi penggunaan *resources*
8	Nama Jurnal: ArXiv Judul Jurnal: Llama Guard: LLM-based Input-Output Safeguard	Llama-2	Sebuah implementasi model *fine-tuned* llama2 yang mampu melakukan filterisasi terhadap penggunaan	*Fine-tuning* model Llama-2 masih memiliki kekurangan terhadap *task* tertentu yang diinginkan	Melakukan pengembangan terhadap efisiensi filterisasi sehingga dapat membaca skenario

	for Human-AI Conversations Penulis: Inan, Hakan Upasani, Kartikeya Chi, Jianfeng Rungta, Rashi Iyer, Krithika Mao, Yuning Tontchev, Michael Hu, Qing Fuller, Brian Testuggine, Davide Khabsa, Madian[21]		kata sehingga dapat mencegah penggunaan kata-kata yang tidak pantas. Model ini mampu menjadi moderasi terhadap input dan memberikan output yang lebih baik.		filterisasi yang lebih besar dan bahasa yang lebih banyak
9	Nama Jurnal: KC UMN Judul: Fine Tuning Model Bahasa MBART50 untuk Question Answering Task dalam Bahasa Indonesia Penulis: Darren Vincentio, Alfonso[22]	MBART 50	*Fine-tuning* model MBART50 menggunakan *dataset* TyDiQA menghasilkan sebuah model dengan nilai ROUGE-1 sebesar 85,89	Model MBART50 menunjukkan performa yang kurang optimal dalam menjawab pertanyaan dengan fungsi agregat tertentu.	Melakukan penelitian berlanjut dengan menggunakan model yang lebih besar untuk melihat perbandingan hasil dengan model sebelumnya
10	Nama Jurnal: KC UMN Judul: Implementasi Teks	BLOOM	*Fine-tuning* model BLOOM yang dilakukan menghasilkan sebuah model	Model BLOOM yang digunakan masih memiliki	Melakukan eksplorasi terhadap metode *fine-tuning* yang lebih efektif

	Klasifikasi Hate Speech Bahasa Indonesia dengan Fine-Tuning Model Bahasa BLOOM Penulis: Sebastian, Richard[23]		yang mampu memberikan klasifikasi terhadap *hate speech*	keterbatasan dalam memproses berbagai variasi teks hate speech dalam Bahasa Indonesia.	dan *dataset* yang lebih luas untuk meningkatkan akurasi model

Jurnal 1 dan 2 dijadikan sebagai acuan untuk melakukan *Text-to-sql* pada model LLM. Kedua jurnal ini menjelaskan sebuah teknik dalam *text-to-sql* yaitu *sequence based* dan *structure-based* dimana kedua teknik tersebut menjadi dasar dari implementasi *text-to-sql*. Dalam penelitian tersebut, dihasilkan sebuah model dan sistem yang dapat melakukan translasi bahasa manusia menjadi sebuah *sql query*, namun implementasi *text-to-sql* tersebut masih terbilang cukup terbatas dengan hasil pada [12] masih menggunakan metode NLP yang perlu di-*define* sendiri sehingga membutuhkan waktu yang lama. Begitu juga dengan [11] dimana model LLM yang dihasilkan masih memerlukan *resource* yang besar. Hal ini mendorong penelitian ini untuk menghasilkan sebuah model *text-to-sql* yang dapat memberikan hasil *sql query* yang sesuai dengan *resources* yang lebih kecil sehingga dapat digunakan dengan mudah. Penelitian ini memanfaatkan metode QloRA untuk menghasilkan model LLM yang efektif dengan penggunaan perangkat keras yang lebih sedikit. Penelitian ini juga memanfaatkan model Llama-2 sebagai dasar dari LLM yang digunakan karena model Llama-2 merupakan model *open-source* yang dapat diakses dengan mudah oleh siapapun.

Penelitian yang dilakukan mengenai implementasi *fine-tuning* untuk membuat *text-to-sql* model mengacu kepada beberapa penelitian yang telah dilakukan. Penelitian nomor 3,4,5, dan 8 menjadi acuan mengenai penggunaan model Llama-2 yang akan di *fine-tune*. 4 Penelitian yang diambil ini menampilkan bagaimana model Llama-2 mampu untuk menghasilkan jawaban yang sesuai

dengan *dataset* yang digunakan. 4 Penelitian ini memberikan gambaran mengenai kemampuan model Llama-2 untuk menerima *dataset* dan memberikan hasil yang signifikan.

Kemudian pada penelitian nomor 6 dan 7 menampilkan metode *fine-tuning* yang terbaik untuk digunakan terutama pada keterbatasan *resources* yang dimiliki. Penelitian nomor 5 memperkenalkan metode LoRA yang digunakan untuk melakukan *fine-tuning* terhadap model Llama-2 untuk menghilangkan *safety training* dari model Llama-2. Metode LoRA yang diperkenalkan ini mampu melatih model Llama-2 dengan *resources* yang lebih sedikit namun memberikan hasil yang maksimal jika dibandingkan dengan metode lain yang menggunakan *resources* lebih banyak.

Terakhir pada penelitian no 9 dan 10 menjelaskan metode evaluasi yang digunakan untuk memberikan gambaran bagaimana performa model yang di *fine-tune* memberikan hasil terhadap pertanyaan yang ditanyakan. Pada Penelitian tersebut dijelaskan mengenai beberapa metode evaluasi seperti *Precision,Recalling, F1 Score, ROUGE Score* dan beberapa metode lain. Pada penelitian ini, digunakan metode *ROUGE Score* sebagai metrik evaluasi yang digunakan untuk menemukan tingkat kemiripan dengan *test dataset* yang digunakan.

Penelitian terdahulu telah menunjukkan potensi besar penggunaan model bahasa besar (LLM) dalam penerapan text-to-SQL, meskipun dengan beberapa keterbatasan seperti kebutuhan resource yang besar dan kurangnya fleksibilitas dalam proses translasi bahasa. Untuk mengatasi keterbatasan tersebut, penelitian selanjutnya dapat dilakukan dengan mengeksplorasi pendekatan menggunakan metode QLoRA (Quantized Low-Rank Adaptation). Metode qLoRA menawarkan pendekatan yang lebih canggih dan efisien dalam fine-tuning model bahasa besar, memungkinkan pengurangan kebutuhan resource sekaligus meningkatkan fleksibilitas dan akurasi dalam mentranslasikan teks natural menjadi query SQL. Dengan pendekatan ini, diharapkan model text-to-SQL dapat mencapai performa yang lebih tinggi dalam berbagai task dan konteks penggunaan, serta mampu

mengakomodasi variasi bahasa yang lebih luas tanpa kehilangan efisiensi operasional.

2.2 Tinjauan Teori

2.2.1 *SQL*

Sturctured Query Language atau SQL adalah sebuah bahasa pemrograman yang digunakan untuk mengelola serta memanipulasi data dalam sebuah sistem RDBMS atau *Relational Database Management System*[24]. SQL memungkinkan dilakukannya beberapa manipulasi data, seperti:

- *Querying* : Mengambil data dari database secara spesifik berdasarkan kondisi yang ditentukan. *Querying* menggunakan *syntax* SELECT untuk memilih data yang ingin ditampilkan atau digunakan
- *Inserting* : Menambahkan data baru ke dalam database yang diakses. Menggunakan *syntax* INSERT untuk meng-*input* data.
- *Updating* : Mengubah data yang sudah ada pada database. Data yang diubah dapat ditentukan kolomnya. Menggunakan *syntax* UPDATE untuk mengubah data.
- *Deleting* : Menghapus baris data yang ada pada database. Menggunakan *syntax* DELETE untuk menghapus baris data yang ditentukan.
- *Creating* : Membuat objek baru dari *database* yang ada, bisa berupa tabel dan indeks. Menggunakan *syntax* CREATE untuk membuat tabel baru.
- *Altering* : Melakukan modifikasi terhadap database yang sudah ada seperti mengubah tipe data pada kolom. Menggunakan *syntax* ALTER untuk modifikasi *database*
- *Dropping* : Menghapus objek yang ada pada database seperti table. Menggunakan *syntax* DROP untuk menghapus objek tersebut.

SQL bekerja berdasarkan relasi dari *database* yang ada dimana data disimpan dalam tabel yang memiliki relasi menggunakan *main key* dan *foreign key*[24]. Setiap tabel dari database tersebut terdiri atas baris(*record*) dan kolom (*field*) dengan setiap baris memiliki angka atau *value* yang berbeda dan kolom sendiri mewakili atribut dari *value* tersebut.

Dengan banyaknya kelebihan pada proses pengolahan data, penggunaan SQL juga memiliki tantangan tersendiri, terutama bagi pengguna awam dan non-teknis. Tingkat kompleksitas SQL menuntut ketepatan dan pemahaman mengenai struktur dari *relational database* itu sendiri sehingga menjadi salah satu hambatan utama[25]. SQL memiliki penulisan yang *case-sensitive* sehingga kesalah kecil seperti kesalahan ketika dan penggunaan tanda baca yang tidak sesuai dapat menyebabkan kegagalan dalam melakukan *execute query* tersebut. Konsep seperti relasi antar tabel, *Primary Key*,*Foreign Key*, dan konsep lainnya juga cukup sulit untuk dipahami bagi pengguna yang tidak memiliki latar belakang teknis[26]. Hal ini menyebabkan adanya kesulitan dalam merancang *query* yang kompleks, serta meningkatkan resiko kesalahan dalam proses pengolahan data.

Dalam implementasi *text-to-sql* pada penelitian ini, diperlukan adanya standarisasi untuk bentuk *sql* yang dapat diberikan oleh model. Salah satu RDBMS yang memiliki jumlah penggunaan terbanyak di dunia adalah mySQL[27]. MySQL merupakan salah satu RDBMS yang dikenal karena kemampuan dan kemudahan dalam penggunaannya. MySQL juga merupakan sebuah sistem yang bersifat *open-source* dan memiliki *support* untuk semua sistem operasi yang ada seperti *windows, Linux,* dan *MacOS.*[28]

MySQL memiliki beberapa *keypoint* atau *key features* yang memberikan perbedaan MySQL terhadap sistem lainnya[29]. Beberapa *key features* tersebut adalah sebagai berikut:

- *Performance*

  MySQL memiliki kinerja yang sangat baik untuk menangani pengolahan data dalam jumlah data yang sangat besar, menjadikan sistem ini sebagai pilihan yang cocok untuk data dalam skala kecil maupun besar
- *Scalability*

  MySQL mendukung beberapa hal seperti *Partitioning*, *Replication*, dan *Clustering* yang mampu membantu tingkat skalabilitas dalam aplikasi
- *Security*

  MySQL memberikan opsi keamanan yang banyak seperti *user authentication*, kontrol akses, dan SSL
- *Flexibility*

  MySQL memiliki kemampuan untuk proses pengolahan data dari berbagai jenis data dan *storage engine*, sehingga memudahkan pengguna MySQL untuk melakukan optimalisasi dalam proses kinerja *database* berdasarkan kebutuhan.
- *Community and Support*

  MySQL merupakan sebuah sistem yang bersifat *open-source* yang Dimana memiliki komunitas yang besar dengan berbagai dokumentasi mengenai penggunaan MySQL yang luas.

Implementasi *text-to-sql* adalah untuk melakukan konversi Bahasa manusia ke dalam bentuk *SQL Query*. Sistem dengan dokumentasi yang lengkap dan terstruktur dengan baik seperti MySQL menjadi pilihan yang baik untuk menjadi standarisasi jawaban yang disampaikan oleh model[30]. Beberapa poin yang dapat dijadikan sebagai justifikasi mengapa MySQL dijadikan sebagai standarisasi bagi model *text-to-sql* tersebut adalah sebagai berikut[28]:

- Popularitas

  MySQL menjadi salah satu pilihan paling sering digunakan dalam bidang industri maupun akademisi, menjadikan sistem ini sebagai pilihan yang relevan
- Dokumentasi yang komprehensif

  Dokumentasi untuk penggunaan MySQL sangat mendalam dan juga dengan adanya *community support* membantu proses pemecahan masalah yang ada
- Kompabilitas

  MySQL mampu digunakan dalam berbagai bahasa pemrograman dan sistem yang ada sehingga impelementasi *text-to-sql* dapat lebih efektif
- Kemudahan Penggunaan

  MySQL memiliki sistem yang mudah digunakan untuk kebanyakan sistem
- Kinerja

  MySQL memiliki kinerja dan performa yang sangat baik untuk mengolah data dalam bentuk besar maupun kecil.

Dalam penelitian mengenai *text-to-sql* ini, MySQL akan digunakan sebagai target RDBMS untuk model *Text-to-sql*. Model nantinya akan menjawab pertanyaan ke dalam bentuk *sql query* yang dapat dijalankan pada database MySQL.

2.2.2 Large Language Model (LLM)

Large Language Model atau LLM merupakan salah satu model dari *Natural Language Process* yang telah dikembangkan. LLM merupakan NLP yang dapat melakukan *task-task* dalam NLP seperti *summarizing* dengan hasil yang sangat baik[31]. LLM dapat menjadi salah satu bentuk NLP yang sangat baik dengan melalui berbagai bentuk *training data* yang dilakukan dengan menggunakan jumlah dataset yang sangat banyak bahkan bisa mencapai miliaran data sehingga dapat menunjukkan performa yang sangat baik. Model scaling terbukti menjadi alasan dari peningkatan

performa hingga dapat menunjukkan hasil yang jauh lebih baik lagi daripada *language* model yang lebih kecil[32]. Salah satu keunggulan dari LLM juga adalah model tersebut terbukti memiliki potensi untuk menjadi sebuah model kognitif dimana pada awalnya ketika melihat hasil dari LLM, jawaban yang diberikan masih memiliki karakteristik seperti robot atau *unhuman-like.* Dengan adanya proses yang dapat disebut *fine-tuning*, LLM dapat dilatih lagi untuk menjadi sebuah model yang dapat mereplika sifat manusia sehingga dapat memberikan jawaban yang mirip dengan cara manusia menjawab sebuah pertanyaan.[33]

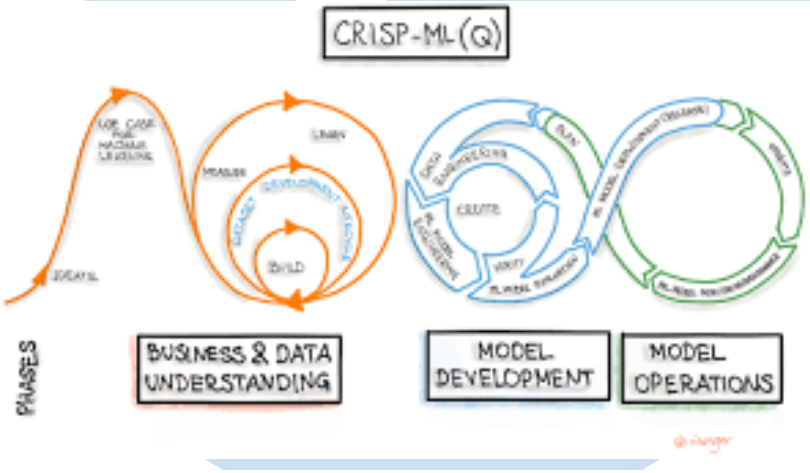
Untuk memahami bagaimana kemampuan LLM dan potensinya, penting untuk memahami cara kerja LLM tersebut. Berikut adalah beberapa prinsip dan teknik yang digunakan pada LLM:

1. Transformer: *Transformer* merupakan sebuah arsitektur *Deep-Learning* yang serbaguna dengan kemampuannya yang mampu memproses penanganan konteks dan fleksibilitas untuk proses *natural language process* atau NLP[34].
2. *Pre-Trained*: LLM dilatih menggunakan data dengan jumlah yang sangat besar dari berbagai sumber yang ada. Selama proses *Pre-Training*, Model LLM tersebut akan belajar bagaimana memprediksi kata berikutnya dalam sebuah kalimat. Proses ini membantu model untuk mempelajari pola bahasa, tata bahasa, dan beberapa fakta lain yang didapatkan dari data yang diberikan[35].

2.3 Framework dan Model yang digunakan

2.3.1 Framework

2.3.1.1 CRISP-ML



Gambar 2. 1 CRISP-ML Development Cycle

CRISP-ML merupakan sebuah metode pengembangan dari CRISP-DM yang dimana memanfaatkan proses dari CRISP-DM sendiri dan menyesuaikannya untuk proses *machine learning*. Metode ini dikembangkan dengan alasan untuk menemukan metode *process model* yang sesuai untuk kebutuhan *Machine Learning*, namun belum ditemukan model yang sesuai sehingga menyebabkan hasil akhir dari sebuah projek *machine learning* dan *data science* tidak dapat berjalan dengan maksimal.[36]

Metode CRISP-ML pada dasarnya memiliki tahap-tahap yang sama seperti CRISP-DM dengan ada penambahan pada proses untuk disesuaikan dengan kebutuhan *data science* dan *machine learning* itu sendiri seperti yang dapat dilihat pada gambar 2.1 diatas mengenai CRISP-ML *Development Cycle*. Dengan adanya CRISP-ML, proses standarisasi untuk mengatasi problem bisnis dapat dilakukan dengan bantuan *machine*

learning[37]. Tahapan-tahapan metode CRISP-ML adalah sebagai berikut::

1. Business and Data Understanding

   Pada tahap yang pertama ini, proses yang dilakukan adalah memahami kebutuhan bisnis atau tujuan dari melakukan metode ini. Selanjutnya setelah memahami kebutuhan bisnis yang dilakukan adalah mengumpulkan dan mempelajari data yang diperlukan untuk keperluan modeling tersebut. Pada tahap ini juga diperlukan untuk menentukan metode dan algoritma apa yang akan digunakan pada proses Modeling.
2. Data Preparation

   Pada tahap kedua ini data-data yang telah dikumpulkan tadi diolah dengan berbagai metode *preprocessing* data seperti melakukan *data cleaning* yang diperlukan untuk menghilangkan data *null*, penambahan informasi pada data dan juga melakukan pemisahan terhadap data *training*.
3. Modeling

   Setelah data telah dibersihkan, proses berikutnya adalah melakukan modeling menggunakan algoritma yang sudah ditentukan pada tahap pertama. Beberapa proses yang dilakukan adalah seperti *model training*.
4. Evaluation

   Tahap berikutnya adalah evaluasi terhadap model untuk memastikan bahwa model yang dibuat sudah sesuai dengan kebutuhan bisnis.
5. Deployment

   Setelah model dapat dipastikan sudah sesuai dengan kebutuhan, maka artinya model sudah dapat digunakan dan dapat diimplementasikan dengan mengintegrasikan model kepada *software* atau sistem yang sudah ada.
6. Monitoring and Maintenance

Tahap terakhir adalah melakukan *maintenance* atau perawatan terhadap model yang ada agar model dapat terus digunakan dan juga dapat dikembangkan lagi.

2.3.2 Model LLM

2.3.2.1 Llama-2

Llama-2 merupakan sebuah *Large Language Model* yang dikembangkan oleh perusahan Meta dan merupakan sebuah pengembangan terbaru dari model lama mereka yaitu Llama-1. Llama sebagai sebuah LLM memiliki proses *training* data yang mirip dengan GPT-3 dengan melakukan *pre-normalization* dimana Llama menggunakan *RMS Norm Normalizing Function* untuk digunakan pada setiap transformer. Llama-2 dilatih dengan menggunakan *training* data yang 40% lebih besar dari pendahulunya yaitu Llama-2 dan juga menambahkan penggunaan *token* untuk konteks dari 2048 menjadi 4096 *tokens*. Llama-2 dilatih dengan menggunakan total 1 triliun *tokens* pada model terkecilnya yaitu Llama 7B dan 1,4 triliun *tokens* pada model Llama 65B dan Llama 33B [38].

Llama-2 merupakan sebuah model *open-source* yang dapat diakses secara publik yang membuat model tersebut dapat dengan mudah digunakan dan diimplementasikan bagi kebutuhan-kebutuhan umum seperti *text processing, summarizing,* dan lain-lain. Llama-2 sendiri memiliki beberapa model yang memiliki opsi jumlah parameter yang dimiliki oleh model tersebut seperti Llama-2 7B yang memiliki 7 miliar parameter hingga Llama-2 70B yang memiliki total 70 miliar parameter sehingga cakupan jawaban yang bisa disampaikan model 70B akan semakin lebih banyak. Terdapat juga Llama-2-chat yang merupakan sebuah model hasil *fine-tuned* dari Meta yang diciptakan sebagai model yang dapat dimanfaatkan sebagai *chatbot*.[39]

2.3.3 Metode *Fine-Tuning*

2.3.3.1 QLoRA

Pada dasarnya, metode *fine-tuning* yang dilakukan untuk melatih kembali model LLM yang sudah ada memerlukan *resources* yang sangat besar untuk diterapkan. Dalam hal ini, diperlukan sebuah

metode yang mampu melakukan *fine-tuning* dengan menggunakan *resources* yang lebih sedikit namun memiliki tingkat akurasi yang sama tingginya dengan metode biasa. PEFT atau *Parameter Efficient Fine-Tuning* menjadi sebuah metode yang digunakan dalam *fine tuning* model LLM yang memungkinkan proses *fine tuning* berjalan dengan mengurangi jumlah parameter yang dilatih pada model LLM tersebut. Hal ini dilakukan karena sebuah model LLM biasanya dapat memiliki jutaan bahkan miliaran parameter sehingga proses *fine tuning* dapat berjalan dengan sangat lama dan memakan *resources* serta biaya yang besar[40].

Pada PEFT sendiri terdapat beberapa metode yang memilliki basis yang sama. Salah satu metode yang paling sering digunakan adalah LoRA atau *Low-Rank Adaption*. LoRA sendiri merupakan metode *fine tuning* PEFT yang memudahkan proses *fine tuning* tersebut dengan melakukan pengurangan terhadap jumlah parameter yang dilatih dibandingkan dengan melakukan *fine-tuning* secara keseluruhan. Hal ini memungkinkan dilakukan karena terjadi proses *freezing* pada model awal dan menciptakan sebuah metode dekomposisi pada setiap layer *transformer* dari model yang sudah ada. Metode ini dapat mengurangi beban pada *resources* yang dibutuhkan seperti RAM GPU, *Storage*, dan beban *hardware* lainnya. Dengan merujuk pada jurnal [41], metode LoRA dipercaya dapat mengurangi pelatihan parameter hingga 10000 kali lipat lebih sedikit dan beban GPU 3 kali lebih sedikit dibandingkan metode *Fine-Tuning* secara keseluruhan. Metode LoRA sendiri juga mampu memberikan hasil yang setara dengan metode *fine-tuning* pada umumnya.

Metode LoRA tersebut juga memiliki sebuah pengembangan yang disebut QloRA atau *Quantized Low-Rank Adaptation* yang memanfaatkan 4-bit *quantization* dan metode LoRA untuk mengurangi penggunaan *memory* pada GPU atau CPU. Pada pengetesan yang dilakukan dalam jurnal [42], metode QloRA mampu

mengurangi memori pada proses *fine-tuning* sebuah model dengan total 65 miliar parameter dari 780GB memori menjadi kurang dari 48GB sehingga terjadi pengurangan memori hingga 93,8% tanpa adanya pengurangan performa pada hasil *fine-tuning* dan juga tidak ada perubahan dalam waktu *fine-tuning* yang dilakukan.

QloRA mampu bersaing dengan proses 16-bit *finetuning* dan 16-bit LoRA pada berbagai proses dengan memanfaatkan 4-bit NormalFloat (NF4). 4-bit NormalFloat merupakan tipe data yang dioptimalkan sehingga bobot pada model sehingga dapat terdistribusi dengan baik. QloRA sendiri dapat mengurangi penggunaan memori tersebut dengan mengurangi jumlah bit untuk melakukan *fine-tuning* pada model dengan memanfaatkan 4-bit NormalFloat tersebut untuk merepresentasikan bobot model yang di-*fine tune*. Cara kedua yang dilakukan adalah dengan proses *double quantization*. Proses *double quantization* merupakan proses menkuantisasi dari proses *quantization* untuk melakukan penghematan memori. *Double Quantization* membantu mengurangi rerataan *memory footprint* dengan melakukan proses kuantisasi cFP32 dari kuantisasi pertama sebagai input pada proses kuantisasi kedua. Proses DQ tersebut dapat mengurangi rata-rata memori per parameter menjadi 0,127 bit dari yang awalnya 0,5 bit[42]. Proses QloRA dijelaskan pada gambar 2.2 dimana metode ini mengkuantisasi model transformer menjadi 4-bit dari yang awalnya 16-bit pada LoRA serta menggunakan *paged optimizers* untuk mengatasi adanya lonjakan *memory usage*.

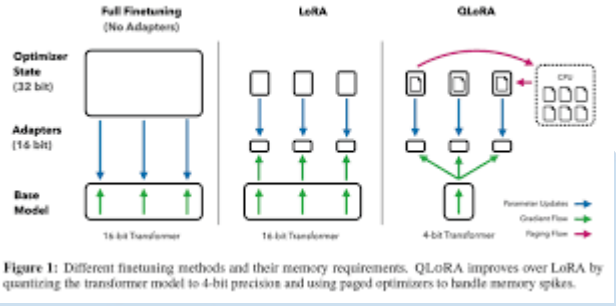


Figure 1: Different finetuning methods and their memory requirements. QLoRA improves over LoRA by quantizing the transformer model to 4-bit precision and using paged optimizers to handle memory spikes.

Gambar 2. 2 QloRA Configuration

2.4 Evaluasi

Beberapa evaluasi dilakukan untuk menentukan apakah model yang dilatih sudah sesuai dengan tujuan penelitian atau tidak. Metrik Evaluasi yang digunakan pada model ini adalah *Accuracy*,*Recall*,*Precision*, *F1-Score* dan METEOR Score[43], [44]

2.4.1 Accuracy

Evaluasi akurasi bertujuan untuk melihat proporsi dari jawaban yang benar dari keseluruhan jawaban dari model terhadap *dataset* yang digunakan. Rumus dari perhitungan *Accuracy* dapat dilihat pada rumus 2.1 berikut.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Rumus 2 1 Perhitungan Accuracy

2.4.2 Recall

Recall adalah sebuah metrik untuk evaluasi untuk menghitung jumlah hasil prediksi yang positif jika dibandingkan dengan keseluruhan data pembanding (dalam hal *training* LLM). Rumus dari perhitungan *Recall* dapat dilihat pada rumus 2.2 berikut

$$Recall = \frac{(TP)}{(TP + FN)}$$

Rumus 2 2 Perhitungan *Recall*

2.4.3 Precision

Precision merupakan sebuah metrik untuk menghitung keakuratan dan seberapa presisi hasil dari model dibandingkan pada *dataset testing* yang ada. Dalam hal *text-to-sql*, *precision* menghitung seberapa mirip hasil dari model terhadap data *testing* atau *ground truth* yang diberikan. Rumus perhitungan dari *precision* dapat dilihat pada rumus 2.3 berikut

$$Precision = \frac{(TP)}{(TP + FP)}$$

Rumus 2 3 Perhitungan *Precision*

2.4.4 F1-Score

F1-Score merupakan metrik yang menghitung rata-rata dari *precision* dan *recall*. Evaluasi ini memberikan gambaran yang lebih seimbang mengenai kemampuan model, terutama ketika terdapat ketidakseimbangan antara hasil *query* model yang benar dan salah. Rumus dari perhitungan *F1-Score* dapat dilihat pada rumus 2.4 berikut.

$$\text{F1-Score} = 2 * \frac{Precision * Recall}{Recall + Precision}$$

Rumus 2 4 Perhitungan *F1-Score*

2.4.5 ROUGE Score

Recall-Oriented Understudy for Gisting Evaluation atau *ROUGE Score* merupakan sekumpulan metrik yang digunakan untuk melakukan evaluasi *generated output* dari model terhadap referensi atau *dataset testing* yang digunakan menggunakan *n-grams*. Evaluasi *Recall* yang biasa dilakukan menghitung kesamaan kata yang ada pada *generated output* dan referensi yang diberikan. Hal ini kurang baik untuk *generated output* yang panjang karena tetap dapat membaca kesamaan kata yang ada pada referensi, sehingga walaupun banyak kata dalam *output* yang sebenarnya tidak sesuai namun tetap terhitung sebagai nilai *recall* yang baik. *ROUGE Score* menghitung *n-gram* atau urutan kata yang berdampingan dari rangkaian teks

atau *output* yang dihasilkan model. *N-gram* ini akan digunakan untuk menghitung urutan kata yang muncul dalam *output* dan referensi *dataset* untuk melihat kesamaan dari urutan kata tersebut[45].

Dalam hal *text-to-sql*, *ROUGE Score* dapat digunakan untuk melihat struktur *query* yang dihasilkan dari *output* model. Dengan nilai ROUGE yang tinggi, struktur *query* yang dihasilkan berarti memiliki kesamaan dengan *struktur query* yang ada pada *dataset testing*. Metode evaluasi *ROUGE Score* ini dapat digunakan sebagai salah satu metrik evaluasi untuk *text-to-sql* meskipun dengan kekurangan seperti hanya menghitung kesamaan leksikal atau kata per kata sehingga tidak melihat secara keseluruhan konteks dari *sql query* yang diberikan[46].

2.5 Teori tentang Tools / Software yang digunakan

2.5.1 Python

Python adalah salah satu Bahasa pemrograman yang sifatnya adalah *interrpreted language. Interpreted Language* adalah sebuah Bahasa pemrograman yang menjalankan program secara langsung dari *source code* yang dibuat. *Interpreter* akan melakukan penerjemahan *source code* secara baris ke baris menjadi *machine codes* yang akan langsung dieksekusi. Hal ini membantu proses analisis data semakin mudah karena beberapa hal yaitu *syntax* yang lebih mudah dan bersih, integrasi yang baik antara simulasi dengan visualisasi, *feedback* yang langsung diberikan ketia *commands* pada *code* berjalan, operasi numerical yang berjalan dengan cepat, dan masih banyak lagi.[47]

2.5.2 Google Colaboratory

Google Colaboratory merupakan sebuah *platform* yang mengadopsi Jupyter Notebooks dan *Virtual Machine* milik Google. Platform ini dapat berfungsi sebagai sebuah *classrsoom* yang dapat melakukan pembelajaran secara terbuka untuk banyak *device* dengan melakukan *sharing* terhadap file *read-only*. Salah satu alasan Google Colab sendiri dibuat adalah untuk kepentingan penelitian terhadap AI dan *Data Science* karena Google Colab

memberikan layanan gratis untuk menggunakan *Virtual Machine* yang memiliki spesifikasi mumpuni untuk kebutuhan proses penelitian AI yang membutuhkan banyak RAM dan juga *graphic processing unit* (GPU) yang sangat besar. Dengan hadirnya Google Colab ini membantu peneliti untuk mempercepat proses pengolahan data untuk keperluan *data science* dan bisa dijalankan tanpa perlu khawatir terhadap *device* lokal yang digunakan[48].

2.5.3 HuggingFace

HuggingFace merupakan sebuah komunitas yang memiliki fokus pada pengembangan *Natural Language Processing* atau NLP dan *Machine Learning*. HuggingFace menyediakan *library* yang dapat digunakan untuk melakukan pengembangan serta penggunaan berbagai model LLM atau NLP serta beberapa *dataset* yang dapat mendukung proses pelatihan maupun implementasi model-model tersebut. HuggingFace memberikan kemudahan akses terhadap teknologi NLP dan mempermudah penggunaan NLP bagi pihak yang tidak memiliki latar belakang teknis yang mendalam. HuggingFace sendiri menyediakan beberapa *library* seperti *Transformers* yang digunakan untuk mengakses model-model seperti Llama-2,GPT,BERT serta mengakses metode *fine-tuning* yang ada seperti PEFT[49].