



Rapport du Projet DATA WAREHOUSE

Filière : Business intelligence and analytics

Benchmark evaluation data warehouse

Réalisé par :

JENNANE SALMA
MICHAAL YASSINE
BOUTANFIT SALMA

Encadré par

Madame BENHIBA

Année Universitaire 2024-2025

RÉSUMÉ

L'émergence des grands modèles de langage (LLMs) comme GPT, LLaMA ou Mistral révolutionne de nombreux secteurs, mais la multiplication des modèles et des méthodes d'évaluation rend difficile leur comparaison objective. Ce projet répond à cette problématique en développant un entrepôt de données complet pour le consortium EvalLLM, permettant l'analyse multidimensionnelle des performances de 4576 évaluations de modèles sur divers benchmarks.

Notre solution s'articule autour d'une architecture décisionnelle robuste utilisant SQL Server comme moteur de stockage, SSIS pour l'automatisation des processus ETL, et Power BI pour les visualisations interactives. Le cœur du système repose sur un modèle dimensionnel en étoile comprenant deux tables de faits et trois dimensions principales, avec une dimension implémentant une gestion historique des changements (SCD Type 2) pour suivre l'évolution des caractéristiques des modèles au fil du temps.

Les tableaux de bord développés ont permis d'identifier plusieurs tendances significatives . Cette infrastructure décisionnelle offre désormais aux chercheurs et aux décideurs un outil uniifié pour guider le développement des futures générations de modèles de langage. Les perspectives d'évolution incluent l'intégration de nouvelles sources de données, l'implémentation d'analyses prédictives, et la transformation en plateforme collaborative pour la communauté scientifique.

ABSTRACT

The rapid emergence of large language models (LLMs) such as GPT, LLaMA, or Mistral is transforming numerous sectors, but the proliferation of models and evaluation methodologies makes objective comparison challenging. This project addresses this issue by developing a comprehensive data warehouse for the EvalLLM consortium, enabling multidimensional analysis of performance across 4576 model evaluations on various benchmarks.

Our solution is built around a robust business intelligence architecture using SQL Server as the storage engine, SSIS for ETL process automation, and Power BI for interactive visualizations. The system's core relies on a star schema dimensional model comprising two fact tables and three main dimensions .One dimension implements historical change tracking (SCD Type 2) to monitor the evolution of model characteristics over time.

The developed dashboards have identified several significant trends. This decision-making infrastructure now provides researchers and decision-makers with a unified tool to guide the development of future generations of language models. Future development prospects include integrating new data sources, implementing predictive analytics, and transforming the system into a collaborative platform for the scientific community.

LISTE DE FIGURES

1.1	Structure des données source	8
1.2	Pipeline de traitement décisionnel EvalLLM	9
2.1	La matrice de bus	11
2.2	Modélisation en étoile (Star Schema)	11
2.3	attributs pour SCD	13
3.1	Control flow de la staging area	16
3.2	Staging area	16
3.3	Flux de données pour la dimension Modèle	17
3.4	Flux de données pour la dimension Benchmark	17
3.5	Flux de données pour la première table de faits	18
3.6	Flux de données pour la seconde table de faits	18
3.7	Validation de l'insertion des données temporelles	19
3.8	Vérification des données de benchmark dans SQL	19
3.9	Model view	20
3.10	Dashboard analytique	20

TABLE DES MATIÈRES

Résumé	1
Abstract	2
Liste de figures	3
Introduction générale	6
1 Contexte et étude exploratoire	7
1.1 Contexte général	7
1.2 Compréhension du besoin métier	7
1.3 Exploration des données source	8
1.4 Flux de traitement décisionnel	9
1.5 Conclusion	9
2 Conception	10
2.1 Objectif de la phase de conception	10
2.2 Axes d'analyse et indicateurs clés	10
2.3 Matrice de bus	10
2.4 Modélisation en étoile (Star Schema)	11
2.4.1 Tables de faits	11
2.4.2 Tables de dimensions	12
2.5 Slowly Changing Dimension (SCD)	13
2.6 Granularité des faits	14
2.7 Processus ETL et flux de données	14
2.8 Conclusion	14
3 Implémentation et Résultats	15
3.1 Technologies utilisées	15
3.2 Implémentation technique	15
3.2.1 Création de la base de données	15
3.2.2 Processus ETL avec SSIS	16
3.3 Résultats et visualisations	19
3.3.1 Tableaux de bord Power BI	19

TABLE DES MATIÈRES

3.3.2 Analyses et interprétations	20
Conclusion et perspectives	22
Bibliographie	24

INTRODUCTION GÉNÉRALE

L'émergence rapide des grands modèles de langage (LLMs) tels que GPT, LLaMA ou Mistral transforme de nombreux secteurs, de l'éducation à la recherche, en passant par l'industrie. Cependant, avec l'augmentation du nombre de modèles et de variantes, il devient de plus en plus complexe de comparer équitablement leurs performances. Les résultats d'évaluation sont souvent épars dans des fichiers CSV, des publications scientifiques, ou des pages web, rendant difficile l'analyse comparative, la traçabilité, et la reproductibilité des résultats.

Dans ce contexte, le consortium de recherche EvalLLM a pour mission de surveiller et d'analyser les performances des LLMs open source sur des benchmarks variés : raisonnement, compétences mathématiques, culture générale, etc. Le projet vise à centraliser ces résultats dans un entrepôt de données (Data Warehouse) conçu pour permettre l'analyse multidimensionnelle des performances selon plusieurs axes : génération du modèle, type de benchmark, taille du modèle, coût environnemental, etc.

Ce rapport présente la conception et la mise en œuvre d'une solution de Business Intelligence (BI) complète : modélisation en étoile, création de datamarts thématiques, intégration des données avec SSIS et visualisation via des dashboards. Cette approche permet à EvalLLM de suivre efficacement les évolutions du paysage des LLMs, d'identifier les modèles les plus performants, et de communiquer des insights clairs aux partenaires académiques et industriels.

CHAPITRE 1

CONTEXTE ET ÉTUDE EXPLORATOIRE

1.1 Contexte général

L'essor des modèles de langage de grande taille (LLM) a conduit à une explosion du nombre de versions, de benchmarks, et de méthodes d'évaluation. Dans ce contexte, EvalLLM, un consortium de recherche, s'est donné pour mission de suivre et d'analyser les performances de ces modèles sur une large variété de benchmarks. Ces évaluations couvrent des capacités telles que le raisonnement, la compréhension linguistique, la mémoire factuelle et les compétences mathématiques.

Les données sont actuellement centralisées dans un fichier CSV unique . Bien que les données soient déjà regroupées, ce format ne permet pas une analyse efficace et flexible des performances. L'exploitation sous forme de fichier plat limite considérablement les possibilités d'analyse croisée, de filtrage dynamique et de visualisation interactive.

L'objectif est de construire une architecture décisionnelle complète, incluant un *Data Warehouse*, afin de faciliter les comparaisons de modèles, le suivi dans le temps, l'analyse des compromis (ex. : performance vs coût), et la génération de rapports pour des conférences ou partenaires industriels.

1.2 Compréhension du besoin métier

Les besoins exprimés par EvalLLM se résument comme suit :

- Centraliser et structurer les résultats de performance des LLM.
- Permettre des analyses comparatives selon différents axes (benchmark, date de sortie, architecture, etc.).
- Suivre l'évolution des modèles dans le temps, y compris par génération.
- Analyser des indicateurs clés comme la performance, le nombre de paramètres, l'impact environnemental.
- Automatiser la production de rapports et de visualisations interactives pour les chercheurs.

Pour répondre à ces besoins, il est nécessaire de mettre en place une architecture décisionnelle adaptée, allant du stockage brut des données à leur visualisation exploitable.

1.3 Exploration des données source

La source principale d'information est un fichier CSV contenant 4576 lignes, avec de nombreuses colonnes représentant les caractéristiques des modèles LLM, leurs performances sur différents benchmarks, leur impact environnemental, ainsi que diverses métadonnées.

Le tableau ci-dessous résume la structure des colonnes du fichier CSV source, incluant leur type de données et le nombre de valeurs non nulles. Cette analyse confirme la présence de champs critiques pour la modélisation du Data Warehouse, tels que les scores normalisés par benchmark ou les métadonnées techniques des modèles :

0	<code>id</code>	4576	non-null	object
1	<code>model_name</code>	4576	non-null	object
2	<code>model_sha</code>	4560	non-null	object
3	<code>model_precision</code>	4576	non-null	object
4	<code>model_type</code>	4576	non-null	object
5	<code>model_weight_type</code>	4576	non-null	object
6	<code>model_architecture</code>	4570	non-null	object
7	<code>model_average_score</code>	4576	non-null	float64
8	<code>model_has_chat_template</code>	4576	non-null	bool
9	<code>evaluations_ifeval_name</code>	4576	non-null	object
10	<code>evaluations_ifeval_value</code>	4576	non-null	float64
11	<code>evaluations_ifeval_normalized_score</code>	4576	non-null	float64
12	<code>evaluations_bbh_name</code>	4576	non-null	object
13	<code>evaluations_bbh_value</code>	4576	non-null	float64
14	<code>evaluations_bbh_normalized_score</code>	4576	non-null	float64
15	<code>evaluations_math_name</code>	4576	non-null	object
16	<code>evaluations_math_value</code>	4576	non-null	float64
17	<code>evaluations_math_normalized_score</code>	4576	non-null	float64
18	<code>evaluations_gpqa_name</code>	4576	non-null	object
19	<code>evaluations_gpqa_value</code>	4576	non-null	float64
20	<code>evaluations_gpqa_normalized_score</code>	4576	non-null	float64
21	<code>evaluations_musr_name</code>	4576	non-null	object
22	<code>evaluations_musr_value</code>	4576	non-null	float64
23	<code>evaluations_musr_normalized_score</code>	4576	non-null	float64
24	<code>evaluations_mmlu_pro_name</code>	4576	non-null	object
25	<code>evaluations_mmlu_pro_value</code>	4576	non-null	float64
26	<code>evaluations_mmlu_pro_normalized_score</code>	4576	non-null	float64
27	<code>features_is_not_available_on_hub</code>	4576	non-null	bool
28	<code>features_is_merged</code>	4576	non-null	bool
29	<code>features_is_moe</code>	4576	non-null	bool
30	<code>features_is_flagged</code>	4576	non-null	bool
31	<code>features_is_official_provider</code>	4576	non-null	bool
32	<code>metadata_upload_date</code>	4046	non-null	object
33	<code>metadata_submission_date</code>	4564	non-null	object
34	<code>metadata_generation</code>	4576	non-null	int64
35	<code>metadata_base_model</code>	4576	non-null	object
36	<code>metadata_hub_license</code>	2823	non-null	object
37	<code>metadata_hub_hearts</code>	4576	non-null	int64
38	<code>metadata_params_billions</code>	4576	non-null	float64

FIGURE 1.1 – Structure des données source

L'exploration préliminaire des données a permis d'identifier plusieurs catégories d'informations importantes :

- **Identifiants et noms de modèles** : `id`, `model_name`, `model_sha`
- **Caractéristiques techniques** : `model_precision`, `model_type`, `model_weight_type`, `model_architecture`, `model_has_chat_template`
- **Performances globales** : `model_average_score`
- **Performances par benchmark** : plusieurs groupes de colonnes comme `evaluations_ifeval_name`, `evaluations_ifeval_value`, `evaluations_ifeval_normalized_score`
- **Caractéristiques spécifiques** : `features_is_not_available_on_hub`, `features_is_merged`, `features_is_moe`, etc.
- **Métadonnées** : `metadata_upload_date`, `metadata_submission_date`, `metadata_generation`, etc.
- **Métriques dérivées** : `metadata_params_billions`, `metadata_co2_cost`

Ces différentes catégories d'informations forment une base solide pour identifier les dimensions et mesures qui seront utilisées dans la modélisation du data warehouse. On distingue déjà des axes d'analyse potentiels (modèle, benchmark, temps, caractéristiques techniques) et des métriques quantifiables (scores, nombre de paramètres, coût CO_2).

1.4 Flux de traitement décisionnel

Le flux envisagé pour ce projet suivra les étapes classiques d'un projet décisionnel, comme illustré à la Figure 1.2 :



FIGURE 1.2 – Pipeline de traitement décisionnel EvalLLM

Les principales étapes du flux sont les suivantes :

- **Import du CSV vers une base de staging** : transformation des données brutes en structure relationnelle temporaire
- **Processus ETL via SSIS** : extraction, transformation et chargement des données vers le Data Warehouse
- **Organisation en Data Warehouse** : structuration selon un modèle dimensionnel adapté aux besoins analytiques
- **Développement de tableaux de bord et rapports** : pour la visualisation et l'exploitation des résultats

Une analyse plus approfondie de chacune de ces étapes, ainsi que la conception détaillée du modèle dimensionnel, seront présentées dans le chapitre suivant.

1.5 Conclusion

Ce premier chapitre a permis de comprendre le contexte du projet EvalLLM, d'analyser les besoins métier et d'explorer la structure des données source.

L'analyse des besoins a révélé la nécessité d'une plateforme décisionnelle permettant d'effectuer des analyses multidimensionnelles sur les performances des modèles de langage. L'exploration des données source a mis en évidence la richesse des informations disponibles dans le fichier CSV fourni, et leur potentiel pour l'analyse comparative des modèles LLM.

Le flux de traitement décisionnel a été esquissé dans ses grandes lignes, fournissant une vision d'ensemble du projet. Le chapitre suivant se concentrera sur la conception détaillée du data warehouse, notamment la modélisation dimensionnelle, la définition précise des tables de dimensions et de faits, ainsi que les transformations nécessaires pour passer des données brutes au modèle analytique.

CHAPITRE 2

CONCEPTION

2.1 Objectif de la phase de conception

La phase de conception vise à transformer les besoins exprimés en une architecture décisionnelle claire, cohérente et modélisable. Elle repose principalement sur deux piliers fondamentaux :

- L'identification des axes d'analyse pertinents pour le suivi et la comparaison des performances des modèles de langage.
- La modélisation de la base décisionnelle sous forme de schéma en étoile, facilitant le stockage, l'agrégation et la restitution des données.

2.2 Axes d'analyse et indicateurs clés

À partir de l'étude des besoins métier, les principaux **axes d'analyse (dimensions)** sont les suivants :

- **Modèle** : nom, famille, génération, architecture, nombre de paramètres, licence.
- **Benchmark** : nom, catégorie (raisonnement, mathématique, linguistique...), métrique utilisée.
- **Temps** : jour, mois, année de publication ou d'évaluation.

Les **indicateurs (faits)** principaux incluent :

- Score brut obtenu par benchmark.
- Score normalisé par benchmark.
- Score moyen global du modèle.
- Nombre de paramètres du modèle.
- Coût environnemental (en kg CO₂).
- Popularité du modèle (HubHearts).

2.3 Matrice de bus

La matrice de bus suivante synthétise les faits et dimensions du Data Warehouse :

Mesures / Dimensions	Model	Benchmark	Date
Benchmark Raw Score	X	X	X
Benchmark Normalized Score	X	X	X
AverageScore	X		X
Hub Hearts	X		X
CO2 Cost	X		X

FIGURE 2.1 – La matrice de bus

2.4 Modélisation en étoile (Star Schema)

Le schéma en étoile ci-dessous représente la structure du Data Warehouse. Il se compose de **deux tables de faits** et de **trois tables de dimensions** connectées par des clés primaires/étrangères :

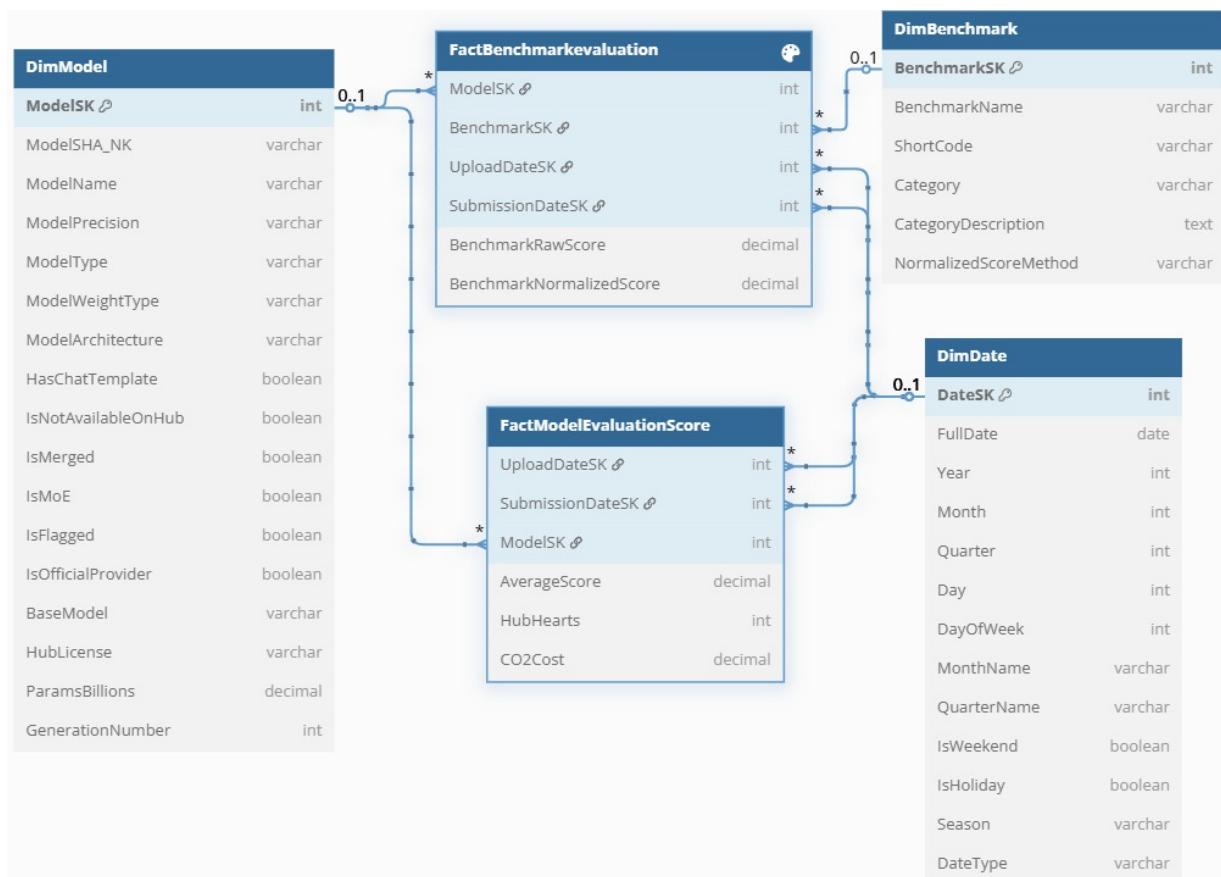


FIGURE 2.2 – Modélisation en étoile (Star Schema)

2.4.1 Tables de faits

Le modèle comprend deux tables de faits distinctes, chacune répondant à des besoins analytiques spécifiques :

- **FactBenchmarkEvaluation** : Cette table contient les mesures relatives aux performances des modèles sur des benchmarks spécifiques. Elle permet d'analyser en détail comment chaque modèle performe sur chaque benchmark individuel.
 - ModelSK (clé étrangère vers DimModel)

- `BenchmarkSK` (clé étrangère vers `DimBenchmark`)
 - `UploadDateSK` (clé étrangère vers `DimDate`)
 - `SubmissionDateSK` (clé étrangère vers `DimDate`)
 - `BenchmarkRawScore` (score brut obtenu)
 - `BenchmarkNormalizedScore` (score normalisé)
- **FactModelEvaluationScore** : Cette table centralise les mesures agrégées relatives à chaque modèle, indépendamment des benchmarks spécifiques. Elle permet d'analyser les performances globales et l'impact environnemental des modèles.
- `ModelSK` (clé étrangère vers `DimModel`)
 - `UploadDateSK` (clé étrangère vers `DimDate`)
 - `SubmissionDateSK` (clé étrangère vers `DimDate`)
 - `AverageScore` (score moyen global)
 - `HubHearts` (popularité du modèle)
 - `CO2Cost` (impact environnemental)

Cette structure à deux tables de faits permet d'optimiser les requêtes selon les cas d'utilisation. Par exemple, les chercheurs intéressés par la performance globale peuvent interroger directement `FactModelEvaluationScore`, tandis que ceux qui souhaitent une analyse détaillée par benchmark utiliseront `FactBenchmarkEvaluation`.

2.4.2 Tables de dimensions

Le modèle comporte trois tables de dimensions principales :

- **DimModel** : Cette dimension contient toutes les caractéristiques des modèles de langage.
 - `ModelSK` (clé primaire)
 - `ModelSHA_NK` (clé naturelle, identifiant unique du modèle)
 - `modelName` (nom du modèle)
 - `ModelPrecision` (précision du modèle : bfloat16, float16, etc.)
 - `ModelType` (type de modèle : chatmodel, pretrained)
 - `ModelWeightType` (type de poids : Original, LoRA, etc.)
 - `ModelArchitecture` (architecture : LlamaForCausalLM, MistralForCausalLM, etc.)
 - `HasChatTemplate` (présence d'un template de chat)
 - `IsNotAvailableOnHub` (disponibilité sur Hugging Face Hub)
 - `IsMerged` (statut de fusion)
 - `IsMoE` (Mixture of Experts)
 - `IsFlagged` (avertissement sur le modèle)
 - `IsOfficialProvider` (statut officiel du fournisseur)
 - `BaseModel` (modèle de base)
 - `HubLicense` (licence : apache-2.0, MIT, etc.)
 - `ParamsBillions` (nombre de paramètres en milliards)
 - `GenerationNumber` (numéro de génération)
- **DimBenchmark** : Cette dimension contient les informations sur les différents benchmarks utilisés pour évaluer les modèles.

- **BenchmarkSK** (clé primaire)
- **BenchmarkName** (nom du benchmark : IFEval, BBH, MATH, etc.)
- **ShortCode** (code court pour le benchmark)
- **Category** (catégorie : raisonnement, mathématique, linguistique, etc.)
- **CategoryDescription** (description détaillée de la catégorie)
- **NormalizedScoreMethod** (méthode de normalisation des scores)
- **DimDate** : Cette dimension temporelle permet d'analyser les performances et les évolutions selon différentes granularités de temps.
 - **DateSK** (clé primaire)
 - **FullDate** (date complète)
 - **Year** (année)
 - **Month** (mois numérique)
 - **Quarter** (trimestre)
 - **Day** (jour)
 - **DayOfWeek** (jour de la semaine)
 - **MonthName** (nom du mois)
 - **QuarterName** (nom du trimestre)
 - **IsWeekend** (indicateur de weekend)
 - **IsHoliday** (indicateur de jour férié)
 - **Season** (saison)
 - **DateType** (type de jour)

2.5 Slowly Changing Dimension (SCD)

Pour suivre l'évolution des caractéristiques des modèles au fil du temps, la dimension **DimModel** est implémentée comme une dimension à évolution lente (SCD) de type 2. Cette méthode permet de conserver l'historique des modifications en créant de nouvelles versions de l'enregistrement lorsque certains attributs changent.

Les attributs suivants sont particulièrement concernés par cette gestion historique :

Dimension Columns	Change Type
IsFlagged	Fixed attribute
IsMerged	Fixed attribute
IsMoE	Fixed attribute
 IsNotAvailableOnHub	Fixed attribute
IsOfficialProvider	Fixed attribute
ModelArchitecture	Historical attribute
ModelPrecision	Historical attribute
ModelSHA_Val	Fixed attribute
ModelType	Fixed attribute
ModelWeightType	Fixed attribute

FIGURE 2.3 – attributs pour SCD

Pour chaque changement sur ces attributs, une nouvelle ligne est créée dans la dimension avec :

- Une nouvelle clé de substitution (**ModelSK**)

- Des dates de validité (début et fin)
- Un indicateur de version courante

Cette approche SCD de type 2 permet aux analystes de réaliser des analyses précises sur l'évolution des modèles et d'effectuer des comparaisons historiques.

2.6 Granularité des faits

La granularité des tables de faits a été déterminée comme suit :

- **FactBenchmarkEvaluation** : Un enregistrement par combinaison unique de modèle, benchmark et date. Cette granularité fine permet d'analyser les performances spécifiques d'un modèle sur un benchmark particulier à un moment donné.
- **FactModelEvaluationScore** : Un enregistrement par modèle et date. Cette granularité plus agrégée facilite l'analyse des performances globales et des caractéristiques générales des modèles.

2.7 Processus ETL et flux de données

Le processus ETL (Extract, Transform, Load) développé sous SSIS suivra les étapes suivantes :

1. **Extraction** des données depuis le fichier CSV source vers une base de staging.
2. **Transformation** des données :
 - Parsing et structuration des données
 - Nettoyage et standardisation des valeurs
 - Génération des clés de substitution
 - Application des règles métier
 - Gestion des dimensions à évolution lente
3. **Chargement** dans les tables de dimensions et de faits du Data Warehouse.

Pour la dimension temporelle (**DimDate**), un processus SQL spécifique a été mis en place pour générer automatiquement tous les attributs temporels nécessaires à partir d'une date de début spécifiée.

2.8 Conclusion

Ce chapitre a présenté la conception détaillée du Data Warehouse pour le projet EvalLLM. La modélisation en étoile avec deux tables de faits et trois dimensions principales offre une structure flexible et performante pour l'analyse des modèles de langage.

L'architecture proposée permet de répondre efficacement aux besoins d'analyse exprimés, notamment la comparaison des performances des modèles sur différents benchmarks, le suivi de leur évolution dans le temps, et l'évaluation des compromis entre performance et coût.

La mise en œuvre de techniques avancées comme les dimensions à évolution lente (SCD) garantit la traçabilité historique des données et la fiabilité des analyses temporelles. Le chapitre suivant détaillera l'implémentation technique de cette architecture, notamment les processus ETL, le développement du cube OLAP et la création des tableaux de bord.

CHAPITRE 3

IMPLÉMENTATION ET RÉSULTATS

3.1 Technologies utilisées

L'implémentation du data warehouse pour l'évaluation des modèles de langage a nécessité l'utilisation d'un ensemble de technologies de l'écosystème Microsoft, chacune répondant à un besoin spécifique dans notre architecture décisionnelle.

- **SQL Server** : Système de gestion de base de données relationnelle utilisé comme moteur de stockage principal pour notre data warehouse, choisi pour sa robustesse et son intégration parfaite avec les autres composants Microsoft.
- **SQL Server Integration Services (SSIS)** : Plateforme ETL de Microsoft utilisée pour l'extraction, la transformation et le chargement des données depuis le fichier CSV source vers notre schéma en étoile.
- **SQL Server Reporting Services (SSRS)** : Solution de reporting utilisée pour la création de rapports statiques et programmés sur les performances des modèles de langage.
- **Power BI** : Outil de visualisation et d'intelligence d'affaires utilisé pour la création de tableaux de bord interactifs et dynamiques.
- **Visual Studio Code** : Environnement de développement utilisé pour la rédaction de scripts SQL et la personnalisation des solutions.

3.2 Implémentation technique

3.2.1 Crédit de la base de données

La première étape a consisté à créer la structure physique du data warehouse dans SQL Server, conformément au schéma en étoile défini dans la phase de conception. Voici les principales tables créées :

- **Tables de dimensions :**
 - DimModel : Stocke les caractéristiques des modèles de langage (nom, architecture, précision, etc.)
 - DimBenchmark : Contient les informations sur les différents benchmarks (IFEval, BBH, MATH, etc.)

- DimDate : Dimension temporelle standard
- Tables de faits :
 - FactBenchmarkEvaluation : Stocke les résultats d'évaluation pour chaque combinaison modèle-benchmark
 - FactModelEvaluationScore : Agrège les scores globaux des modèles

La dimension DimModel a été implémentée comme une Slowly Changing Dimension de type 2 pour suivre l'évolution des caractéristiques des modèles au fil du temps. Cette approche nous permet de conserver l'historique des changements dans les attributs des modèles, comme les améliorations d'architecture ou les modifications de paramètres.

3.2.2 Processus ETL avec SSIS

Le processus ETL a été implémenté à l'aide de SSIS en suivant les étapes ci-dessous :

1. **Extraction** : Lecture du fichier CSV source contenant les 4576 enregistrements en la chargeant dans une staging area.

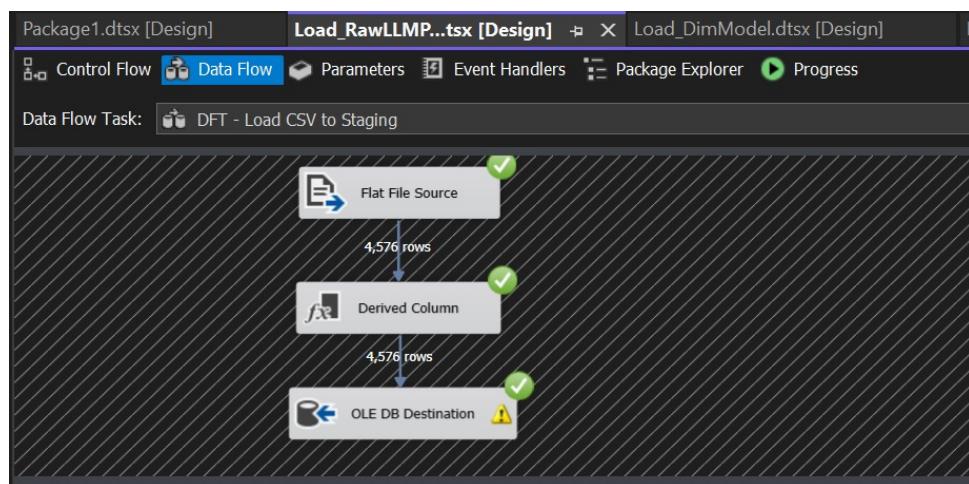


FIGURE 3.1 – Control flow de la staging area

Cette étape montre la configuration du flux de contrôle pour l'extraction des données source vers notre zone de préparation.

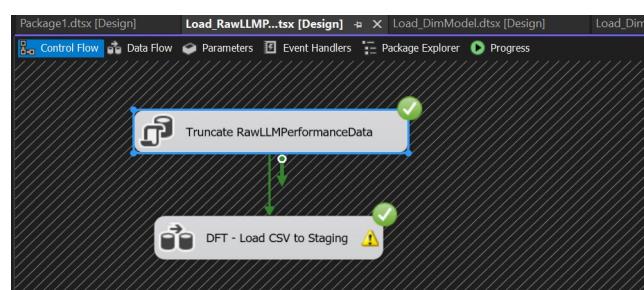


FIGURE 3.2 – Staging area

Cette figure présente le processus de transformation initiale des données brutes avant leur chargement dans le data warehouse.

2. **Transformation** : Conversion des types de données, calcul de mesures dérivées, et préparation des clés de substitution. Cette étape est cruciale pour assurer l'intégrité et la cohérence des données dans notre modèle en étoile.
3. **Chargement** : Alimentation des tables de dimensions puis des tables de faits, en respectant les dépendances entre ces structures.

Plusieurs packages SSIS ont été développés pour orchestrer ces différentes phases. Nous présentons ci-dessous les flux de données les plus significatifs qui illustrent notre approche.

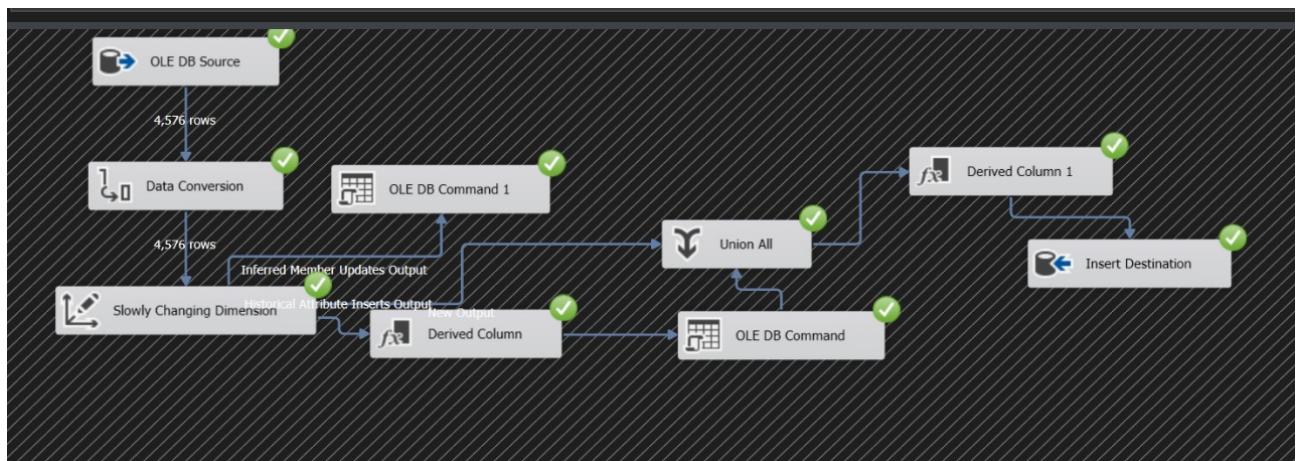


FIGURE 3.3 – Flux de données pour la dimension Modèle

Ce flux implémente la logique SCD Type 2 pour suivre l'historique des changements. On peut observer les différentes transformations nécessaires, notamment la gestion des clés de substitution et les dérivations de colonnes pour les attributs d'historisation.

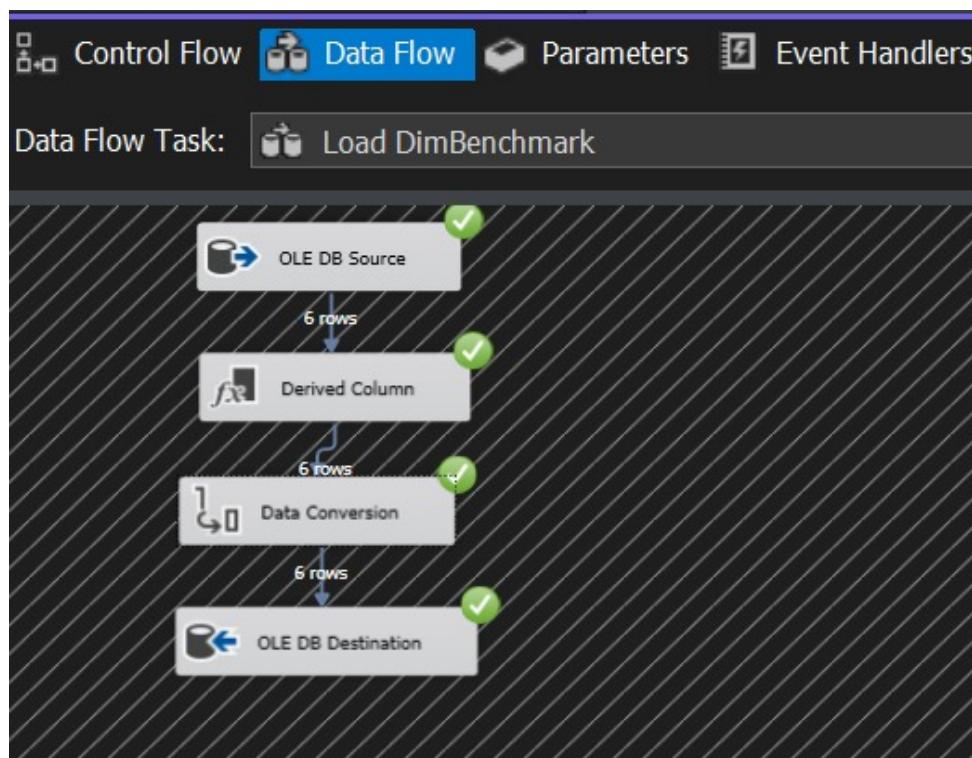


FIGURE 3.4 – Flux de données pour la dimension Benchmark

Cette vue montre les étapes de transformation et de chargement des informations relatives aux benchmarks. Le processus inclut l'extraction depuis la source, la dérivation de colonnes pour enrichir les données, et le chargement final dans la table de dimension.

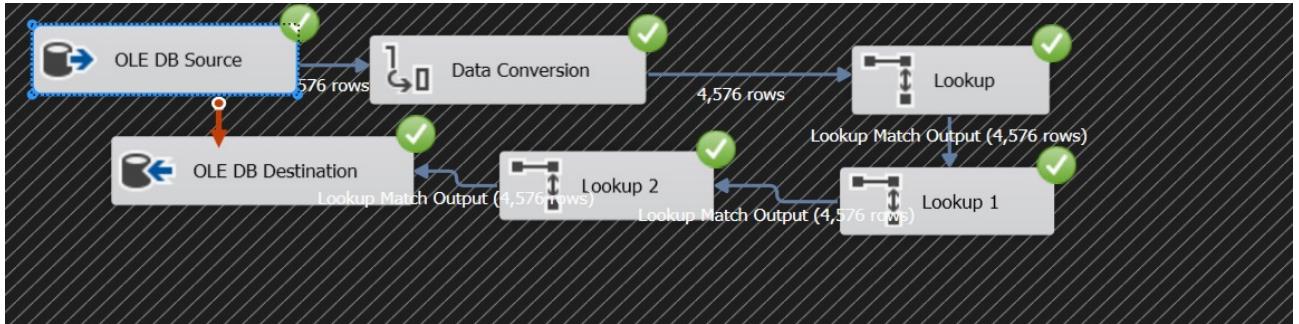


FIGURE 3.5 – Flux de données pour la première table de faits

Cette capture illustre le processus complexe de préparation des données pour la table FactBenchmarkEvaluation. On peut y voir les multiples transformations, lookups et jointures nécessaires pour associer correctement les clés de dimension aux mesures.

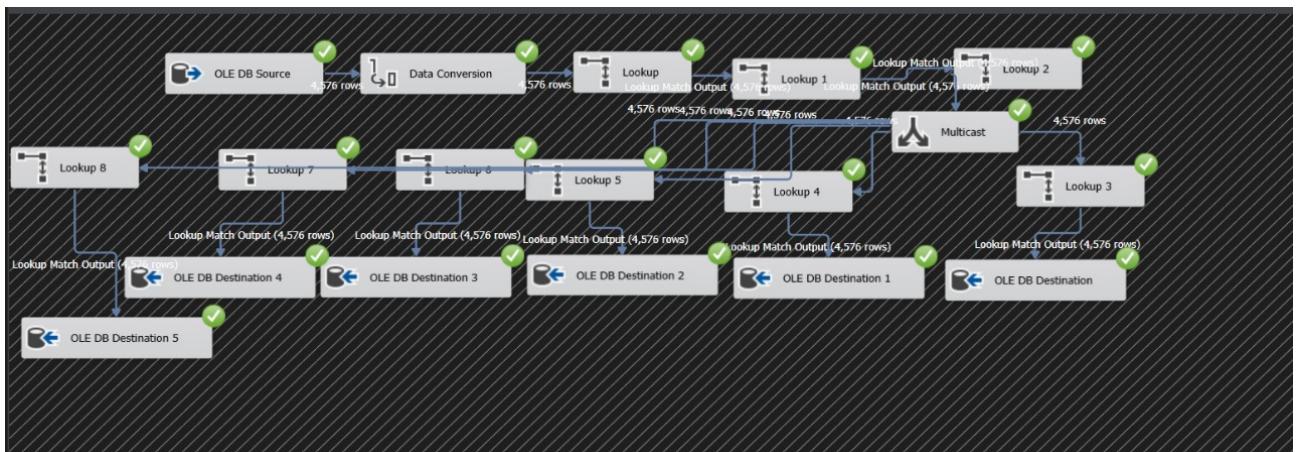


FIGURE 3.6 – Flux de données pour la seconde table de faits

Ce flux montre le traitement des données pour la table FactModelEvaluationScore, qui agrège les résultats par modèle. La complexité du flux reflète les nombreuses opérations de recherche et de transformation requises pour calculer les métriques agrégées.

	DateSK	FullDate	Year	Month	Day	DayOfWeek	MonthName	QuarterName	IsWeekend	IsHoliday	Season
1	20220302	2022-03-02	2022	3	2	4	March	Q1	0	0	Spring
2	20220303	2022-03-03	2022	3	3	5	March	Q1	0	0	Spring
3	20220304	2022-03-04	2022	3	4	6	March	Q1	0	0	Spring
4	20220305	2022-03-05	2022	3	5	7	March	Q1	1	0	Spring
5	20220306	2022-03-06	2022	3	6	1	March	Q1	1	0	Spring
6	20220307	2022-03-07	2022	3	7	2	March	Q1	0	0	Spring
7	20220308	2022-03-08	2022	3	8	3	March	Q1	0	0	Spring
8	20220309	2022-03-09	2022	3	9	4	March	Q1	0	0	Spring
9	20220310	2022-03-10	2022	3	10	5	March	Q1	0	0	Spring
10	20220311	2022-03-11	2022	3	11	6	March	Q1	0	0	Spring
11	20220312	2022-03-12	2022	3	12	7	March	Q1	1	0	Spring
12	20220313	2022-03-13	2022	3	13	1	March	Q1	1	0	Spring
13	20220314	2022-03-14	2022	3	14	2	March	Q1	0	0	Spring

FIGURE 3.7 – Validation de l'insertion des données temporelles

Cette capture d'écran confirme le bon chargement des données dans la dimension temporelle, avec les attributs de date correctement formatés et prêts pour l'analyse.

BenchmarkSK	BenchmarkName	ShortCode	BenchmarkDescription	NormalizedScoreMethod
1	Big Bench Hard (BBH)	BBH	Collection of challenging for LLM tasks across dom...	Percentage-based normalization
2	Graduate-Level Google-Proof Q&A (GPQA)	GPQA	PhD-level knowledge multiple choice questions in sc...	Percentage-based normalization
3	Instruction-Following Evaluation (IFEval)	IFEval	Tests model's ability to follow explicit formatting instr...	Percentage-based normalization
4	Mathematics Aptitude Test of Heuristics (MATH), le...	MATH Level 5	High school level competitions mathematical proble...	Percentage-based normalization
5	Massive Multitask Language Understanding - Profes...	MMLU-PRO	Expertly reviewed multichoice questions across dom...	Percentage-based normalization
6	Multistep Soft Reasoning (MuSR)	MUSR	Reasoning and understanding on/of long texts Lan...	Percentage-based normalization

FIGURE 3.8 – Vérification des données de benchmark dans SQL

L'image montre les enregistrements insérés dans la table DimBenchmark, confirmant que les informations descriptives et les métriques associées ont été correctement chargées.

3.3 Résultats et visualisations

3.3.1 Tableaux de bord Power BI

Après avoir implémenté notre data warehouse et chargé toutes les données nécessaires, nous avons créé des visualisations interactives avec Power BI pour faciliter l'analyse des performances des modèles de langage.

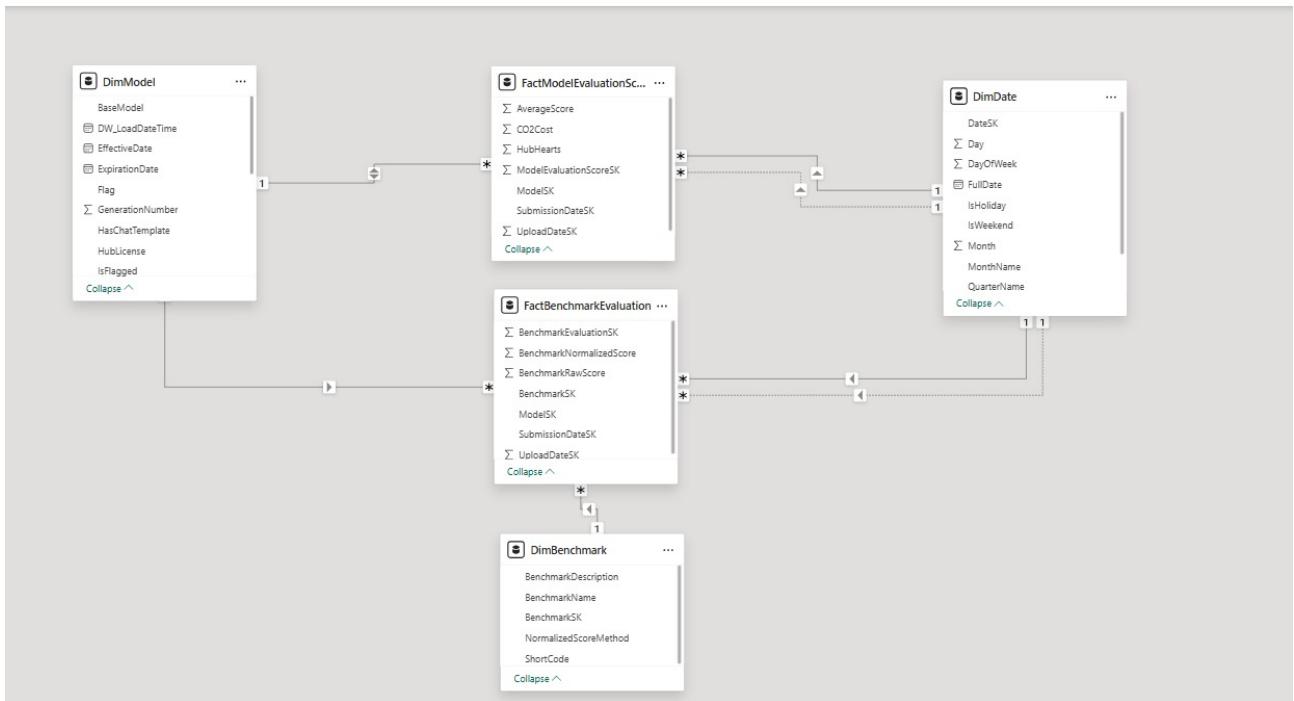


FIGURE 3.9 – Model view



FIGURE 3.10 – Dashboard analytique

3.3.2 Analyses et interprétations

L'analyse approfondie de notre data warehouse à travers les visualisations Power BI nous a permis de dégager plusieurs insights significatifs concernant les modèles de langage évalués :

- **Performance des modèles** : Nos données révèlent que les modèles les plus performants (notamment llama/13, fbigpt7b et Qwen7b) obtiennent des scores moyens entre 40 et

50 points, avec une performance globale moyenne de 21,81 pour l'ensemble des modèles. Cette métrique constitue une référence importante pour évaluer les nouveaux modèles.

- **Efficience environnementale** : De façon intéressante, nous constatons qu'il n'existe pas de corrélation directe entre le coût carbone (moyenne de 3,96) et la performance des modèles. Ce constat suggère que l'efficience environnementale peut être significativement améliorée sans compromettre les résultats, ouvrant la voie à des modèles plus écologiques.
- **Évolution de la taille des modèles** : L'analyse temporelle indique une certaine stabilisation dans la taille des modèles (mesurée en milliards de paramètres) depuis juillet 2024, malgré un pic initial atteignant 60 milliards. Cette tendance pourrait indiquer que l'industrie atteint un plateau d'efficacité en termes de nombre de paramètres, privilégiant désormais l'amélioration architecturale.
- **Popularité vs performance technique** : Nos données montrent un écart notable entre ces deux métriques : les modèles les plus utilisés (comme meta/llama avec plus de 5000 utilisateurs) ne sont pas nécessairement ceux qui obtiennent les meilleurs scores. Cela révèle un potentiel décalage entre l'adoption par les utilisateurs et l'excellence technique, probablement dû à des facteurs comme la facilité d'utilisation ou la disponibilité.
- **Difficulté des benchmarks** : La catégorisation des benchmarks par difficulté démontre que les évaluations de type IFEval (Instruction Following Evaluation) représentent le défi le plus important pour les modèles actuels, tandis que les tests de raisonnement QA (Google-PACT) s'avèrent moins exigeants. Cette hiérarchie permet de mieux cibler les efforts d'amélioration.
- **Fréquence des soumissions** : Nous observons des pics significatifs dans la soumission de nouveaux modèles, notamment en juillet 2024 et début 2025, illustrant l'intense activité de recherche et développement dans ce domaine en constante évolution.

Ces observations constituent une base solide pour orienter le développement futur des modèles de langage, en mettant l'accent sur l'optimisation de l'architecture, la réduction de l'empreinte environnementale et l'amélioration ciblée des performances sur les benchmarks les plus difficiles.

CONCLUSION ET PERSPECTIVES

Ce projet d'entrepôt de données dédié à l'évaluation des modèles de langage (EvalLLM) a permis de mettre en place une solution décisionnelle complète répondant aux besoins d'analyse comparative des performances des LLMs. À travers une modélisation dimensionnelle adaptée et l'implémentation d'un processus ETL robuste, nous avons transformé un ensemble de données brutes (fichier CSV de 4576 lignes) en un système analytique capable de révéler des tendances et des insights pertinents.

Synthèse des réalisations

L'architecture décisionnelle mise en place repose sur un modèle en étoile comportant deux tables de faits (FactBenchmarkEvaluation et FactModelEvaluationScore) et trois dimensions principales (DimModel, DimBenchmark et DimDate). Cette structure offre une grande flexibilité pour l'analyse multidimensionnelle des performances des modèles, tout en conservant l'historique des évolutions grâce à l'implémentation d'une dimension à évolution lente (SCD Type 2) pour les caractéristiques des modèles. Les processus d'extraction, de transformation et de chargement ont été automatisés pour garantir l'intégrité et la cohérence des données, depuis leur source jusqu'aux tableaux de bord d'analyse. La solution développée permet désormais aux chercheurs et aux décideurs de disposer d'une vue unifiée et fiable des performances comparatives des différents modèles de langage disponibles sur le marché.

Perspectives d'évolution

Bien que fonctionnel et répondant aux besoins initiaux, ce projet pourrait être enrichi selon plusieurs axes :

- **Intégration de nouvelles sources de données** : Au-delà du fichier CSV initial, la solution pourrait être étendue pour intégrer automatiquement les données de HuggingFace, des publications académiques et d'autres plateformes d'évaluation.
- **Analyse prédictive** : L'intégration d'algorithmes de machine learning permettrait de prévoir l'évolution des performances des modèles et d'identifier les architectures les plus prometteuses.
- **Plateforme collaborative** : Transformer l'outil en une plateforme communautaire où les chercheurs pourraient soumettre leurs propres évaluations et enrichir collectivement la base de connaissances.

- **Automatisation complète :** Mettre en place un système d'évaluation automatisé qui exécuterait les benchmarks sur les nouveaux modèles dès leur publication.

Ces perspectives d'évolution permettraient non seulement d'améliorer la solution existante, mais aussi de contribuer significativement à la démocratisation des outils d'évaluation standardisés dans le domaine des modèles de langage. Cette standardisation est essentielle pour garantir la transparence, la reproductibilité et la fiabilité des comparaisons entre modèles, dans un contexte d'innovation rapide et continue. Au final, ce projet constitue une première étape vers une compréhension plus systématique et plus approfondie de l'écosystème des modèles de langage, offrant aux chercheurs et aux décideurs un outil précieux pour guider leurs choix et leurs investissements dans ce domaine stratégique.

BIBLIOGRAPHIE

- [1] Papers with Code - LLM Benchmarks. Disponible sur : <https://paperswithcode.com/task/language-modelling>.
- [2] Hugging Face Model Hub. Disponible sur : <https://huggingface.co/models>.
- [3] Documentation officielle de SSIS. Disponible sur : <https://learn.microsoft.com/fr-fr/sql/integration-services/sql-server-integration-services>.
- [4] SQL Server Analysis Services Documentation. Disponible sur : <https://learn.microsoft.com/fr-fr/analysis-services/analysis-services-overview>.
- [5] Power BI et Machine Learning. Disponible sur : <https://learn.microsoft.com/fr-fr/power-bi/transform-model/data-sources/machine-learning>.
- [6] Advanced DAX for Power BI. Disponible sur : <https://www.sqlbi.com/guides/dax/>.
- [7] The Data Warehouse Toolkit (Kimball). Disponible sur : <https://www.kimballgroup.com/data-warehouse-business-intelligence-resources/>.