



Optimiser l'IA Générative Avec Les Données Internes

**Exploiter la RAG pour
déployer des bots
performants**

L'Intelligence Artificielle Générative (GenAI) a profondément transformé notre quotidien et nos méthodes de travail. Avec l'accès facilité aux modèles d'IA via ordinateurs et smartphones, il est crucial de comprendre les technologies derrière cette transformation.

Les Large Language Models (LLM) sont parmi les innovations les plus marquantes. Conçus par des leaders technologiques majeures tels qu'OpenAI, Google, Meta, Mistral, ces modèles comprennent et génèrent du texte de manière impressionnante. Grâce à leurs milliards de paramètres, ils produisent des réponses cohérentes et pertinentes à une variété de requêtes.

Adapter l'IA aux entreprises

Adoption et défis des LLM dans les entreprises

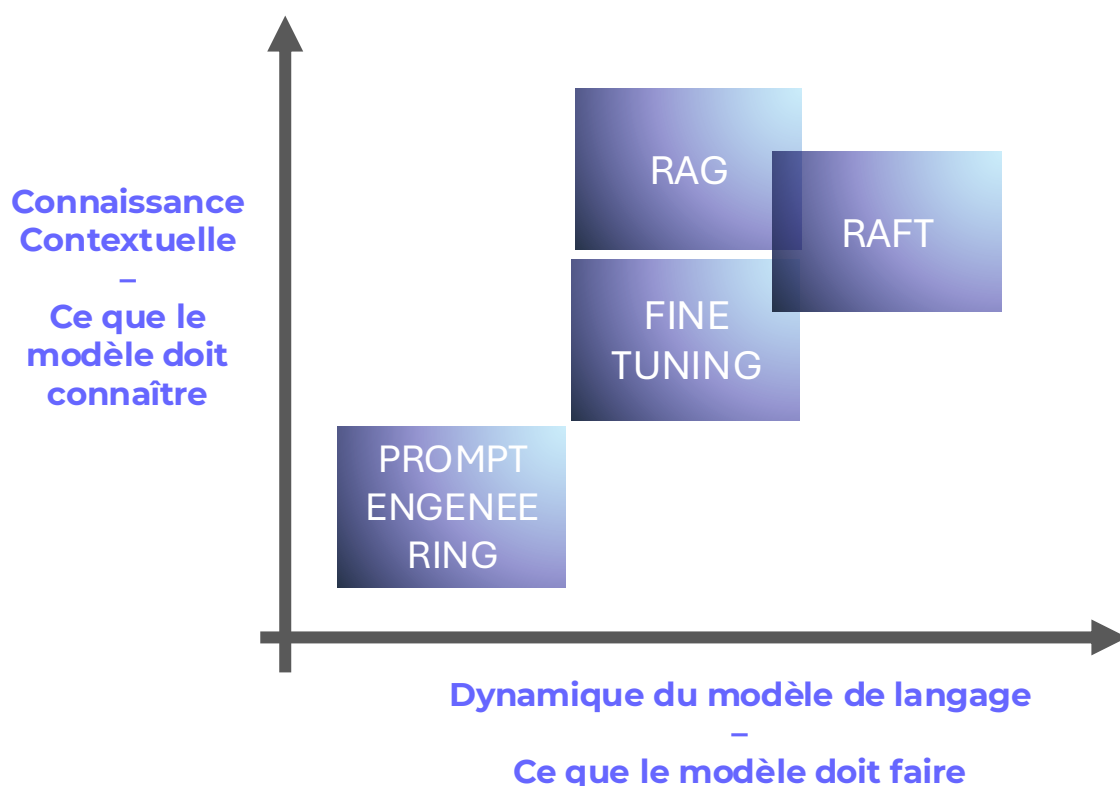
L'adoption de la GenIA en entreprise progresse rapidement. D'après une étude de McKinsey, plus de 70% des grandes entreprises prévoient d'intégrer l'IA dans leurs processus d'ici cinq ans, conscientes de l'impact significatif de ces technologies sur la productivité et l'efficacité. Bien que puissants, les LLM sont souvent trop généralistes et nécessitent une adaptation pour mieux correspondre aux spécificités et à l'identité de chaque entreprise. Le principe fondamental de la GenIA est de fournir la réponse la plus pertinente à une requête utilisateur. Pour des utilisations spécifiques en entreprise, atteindre ce niveau de pertinence nécessite de personnaliser les interactions avec des données internes tout en assurant la sécurité des données.

Un challenge d'envergure

Pour réussir cette personnalisation, il est essentiel d'identifier des cas d'utilisation précis, afin de piloter efficacement la transformation. Les modèles doivent être ajustés pour s'aligner avec la culture, la raison d'être et les objectifs de l'entreprise. Un suivi rigoureux et une gouvernance efficace sont indispensables pour garantir une amélioration continue des performances du modèle. Dirigeants, managers, experts métiers, responsables produits, il est crucial d'adopter stratégiquement ces technologies pour maintenir la compétitivité. Cette transformation s'accompagne de défis majeurs, notamment la personnalisation, la sécurité des données, les biais, les coûts et la conformité réglementaire.

Stratégies de personnalisation

La personnalisation de l'IA est indispensable pour répondre aux besoins spécifiques des entreprises. Plusieurs techniques permettent d'adapter les modèles de langage, chacune ayant ses propres avantages et applications. Les principales méthodes incluent l'ingénierie des prompts (Prompt Engineering), le fine-tuning, l'intégration des données internes spécifique (RAG), et les ajustements adaptatifs (RAFT). Chaque technique procure un niveau distinct de personnalisation et d'optimisation contextuelle pour améliorer les performances du modèle. Les questions de sécurité des données et de confidentialité sont également cruciales, car les LLMs manipulent souvent des informations sensibles. Des mesures robustes de protection des données doivent être mises en place pour éviter les fuites et les accès non autorisés.



Prompt Engineering

Maximiser les performances des modèles.

Le prompt engineering consiste à créer des instructions claires pour orienter le modèle de langage vers des réponses pertinentes, améliorant ainsi l'efficacité des interactions.

Zero-shot Learning

Le zero-shot learning permet au modèle de répondre à une requête sans contexte ni exemples, offrant la réponse la plus générique possible.

One-shot Learning

Cette technique consiste à fournir au modèle un exemple de réponse (« shot »), pour une requête donnée, permettant des réponses plus ciblées.

Few-shot Learning

Elle utilise plusieurs exemples concrets pour valider des cas d'utilisation spécifiques avec peu de ressources, facilitant des ajustements rapides.

Role Prompting

Le role prompting attribue des rôles spécifiques à l'IA (par exemple, data scientist) pour affiner le modèle et améliorer la pertinence des résultats. Par exemple, un prompt détaillé comme « Explique en détail la différence entre la météo et le climat en abordant la définition, les échelles de temps, les facteurs influençant et des exemples concrets » oriente le modèle vers une réponse précise et complète.

Techniques spécialisées

L'art de la conversation inclut aussi le style prompting, l'emotion prompting, le chain-of-thought, le step-back prompting, etc.

Limites

La qualité des résultats dépend des exemples fournis. Des exemples mal conçus peuvent entraîner des réponses incohérentes.

Le prompt engineering n'introduit pas de nouvelles informations au modèle, limitant son efficacité pour des mises à jour fréquentes. Les prompts détaillés peuvent consommer beaucoup de tokens, réduisant l'efficacité des interactions.

Fine-Tuning

Le fine-tuning consiste à ajuster un modèle de langage pré-entraîné avec des données spécifiques à une tâche ou un domaine particulier, sans en modifier l'architecture. Ce processus optimise les poids internes du modèle pour améliorer ses réponses.

L'objectif est d'optimiser les performances du modèle pour des tâches spécifiques sans repartir de zéro. Grâce à des données ciblées, le modèle ajuste ses réponses pour mieux correspondre aux attentes de l'entreprise, offrant une personnalisation efficace et moins complexe que d'autres systèmes interactifs.

Pour réussir un fine-tuning, il est essentiel de fournir un jeu de données ciblé et pertinent. La qualité et la spécificité des données influencent directement la performance du modèle ajusté. La durée du processus peut varier selon la taille du modèle et l'étendue des données. Les modèles pré-entraînés, comme ceux de GPT, comptent des milliards de paramètres. Ces paramètres doivent être ajustés pour optimiser les réponses à des tâches spécifiques, nécessitant des ressources informatiques importantes.

Par exemple, une entreprise souhaitant améliorer son service client automatisé peut recourir au fine-tuning pour adapter un modèle de langage en fonction des interactions spécifiques de ses clients. En intégrant des données historiques de conversations, le modèle peut apprendre à fournir des réponses plus pertinentes et précises, alignées avec les attentes et le langage de la clientèle.

Cependant, le fine-tuning présente certaines limites. Il nécessite une quantité importante de données spécifiques et de qualité. Si les données sont insuffisantes ou inappropriées, le modèle peut générer des réponses inexactes ou incohérentes. Le processus peut également être coûteux en ressources informatiques et en temps, surtout pour les grands modèles. Un modèle trop spécialisé peut également perdre sa capacité à généraliser sur des tâches non couvertes par les données de fine-tuning.

Malgré ces limitations, le fine-tuning reste une technique puissante pour adapter les modèles de langage aux besoins spécifiques des entreprises.

RAG : Retrieval-Augmented Generation

La RAG est une technique avancée qui combine un modèle de génération de texte avec un système de récupération d'informations pour améliorer la qualité et la pertinence des réponses. Cette technique utilise des données externes supplémentaires pour enrichir les réponses générées, offrant ainsi des interactions plus précises, tout en réduisant ainsi les erreurs et les "hallucinations" du modèle.

Mise en œuvre

La RAG nécessite une gestion efficace du modèle de génération et une intégration harmonieuse des interactions entre la récupération d'informations et la génération de texte. Elle fonctionne en trois étapes : la récupération, l'augmentation et la génération. Lorsqu'une requête est reçue, le système recherche des informations pertinentes dans une base de données externe. Ces données enrichissent l'entrée contextuelle du modèle génératif, permettant ainsi de produire une réponse détaillée et informée. Cela implique de développer des mécanismes robustes pour sélectionner et utiliser les données pertinentes afin d'optimiser le processus de génération.

Par exemple, une entreprise financière peut utiliser la RAG pour répondre à des questions complexes sur les produits financiers. Lorsqu'un utilisateur pose une question, le système récupère des informations pertinentes à partir d'un corpus de connaissances internes, puis utilise ces informations pour générer une réponse précise.

Limites et Défis

La qualité des réponses générées par la RAG dépend fortement des informations récupérées. Si ces données ne sont pas à jour ou précises, les réponses peuvent être inexactes. Le processus de récupération d'informations peut également introduire une latence, ralentissant la génération de réponses et impactant l'expérience utilisateur. La RAG nécessite également une infrastructure complexe et une expertise technique pour maintenir et optimiser le système, ce qui peut représenter des coûts opérationnels significatifs.

GraphRAG

GraphRAG est une approche novatrice basée sur la théorie des graphes pour la recherche d'informations, encore en cours d'étude et d'optimisation. Elle surmonte certaines limites de la RAG classique en réduisant le décalage sémantique entre les questions et les réponses. GraphRAG modélise les connexions entre les concepts au sein d'un graphe de connaissances, ce qui permet de combler un manque de lien direct entre la requête d'un utilisateur et les vecteurs de la base de données.

Ce processus offre une récupération d'information plus précise et flexible. Plutôt que de comparer tous les vecteurs à une seule requête, le système explore un graphe pour récupérer et étendre les nœuds les plus pertinents. Ces systèmes sont plus transparents et interprétables car les nœuds et relations récupérés à chaque étape peuvent être visualisés. Cela facilite l'identification des erreurs et augmente la confiance dans les décisions prise par le modèle.

Cette transparence permet aux entreprises de mieux comprendre et contrôler le processus de récupération d'information. Elles peuvent optimiser leurs graphes de connaissances avec des objectifs spécifiques en ajoutant des métadonnées, rendant la récupération d'information plus efficace.

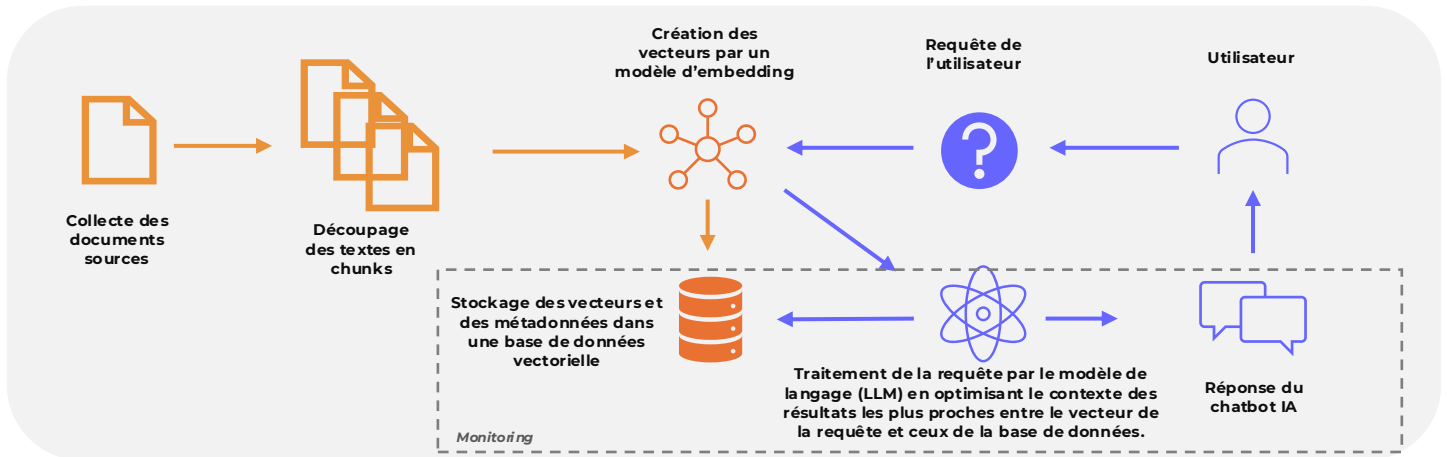
RAFT : Retrieval-Augmented Fine-Tuning

La RAFT, développé par des chercheurs de l'Université de Californie à Berkeley, combine les avantages de la RAG et du fine-tuning pour améliorer les capacités des modèles de langage dans des contextes spécifiques. Ce processus inclut plusieurs étapes : la préparation des données d'entraînement, le fine-tuning du modèle, et la génération de réponses structurées.

La RAFT utilise des documents récupérés pour guider le fine-tuning. Lorsqu'une question est posée, le modèle récupère des documents pertinents, classés en "oracles" et "distracteurs". Le modèle est entraîné à ignorer les distracteurs et à se concentrer sur les oracles pour générer des réponses.

Personnalisation d'un Chatbot IA ESG

Les diverses méthodes de personnalisation de l'IA offrent des possibilités intéressantes, mais il peut être difficile de savoir par où commencer. Cette section se concentre sur la RAG (Retrieval-Augmented Generation), choisie pour sa simplicité et son intuitivité. Elle permet la mise en place de cas d'usage et l'évaluation de leur utilité avant de s'engager éventuellement vers des solutions plus complexes et coûteuses. Le choix de la solution la plus adaptée dépend du contexte spécifique de l'entreprise, qu'il s'agisse d'optimiser un chatbot ou de fournir des données en temps réel.



1 API & LLM

La première étape consiste à charger un modèle de langage de grande taille (LLM) optimisé pour les interactions conversationnelles. Les LLM sont développés pour exceller dans diverses tâches spécifiques : certains sont particulièrement efficaces pour la synthèse de texte, d'autres pour la traduction, et d'autres encore pour les conversations.

Pour accéder à des modèles de LLM performants, une méthode efficace consiste à utiliser une API. Des plateformes comme Hugging Face ou Ollama facilitent l'accès à une variété de modèles de langage. Pour ce projet spécifique, le choix du LLM conversationnel s'est porté sur le modèle d'OpenAI, accessible via une API payante.

Hugging Face 🤖

Cette startup française, valorisée à plusieurs milliards, héberge une large collection de modèles de langage prêts à l'emploi. Elle permet aux développeurs de télécharger, entraîner et déployer facilement des modèles pour diverses applications de traitement du langage naturel. Hugging Face propose notamment des modèles comme LLaMA (Meta), Gemma (Google) ou Mistral (startup française).

Ollama 🐙

Ollama est une plateforme qui simplifie l'utilisation des modèles de langage en offrant des API et des outils pour les intégrer dans diverses applications. Elle se distingue par son approche axée sur la convivialité et l'intégration rapide des modèles en local.

2 Chargement et exploration des données

Les données nécessaires peuvent provenir de diverses sources, telles que des fichiers texte, des PDF ou des tableurs Excel. Il est essentiel de charger ces fichiers efficacement, et de nombreux packages Python, comme Pandas, PyPDF2 ou Openpyxl, facilitent ce processus.

Une fois les données chargées, une analyse exploratoire (EDA) est cruciale pour comprendre la nature et la taille des textes.

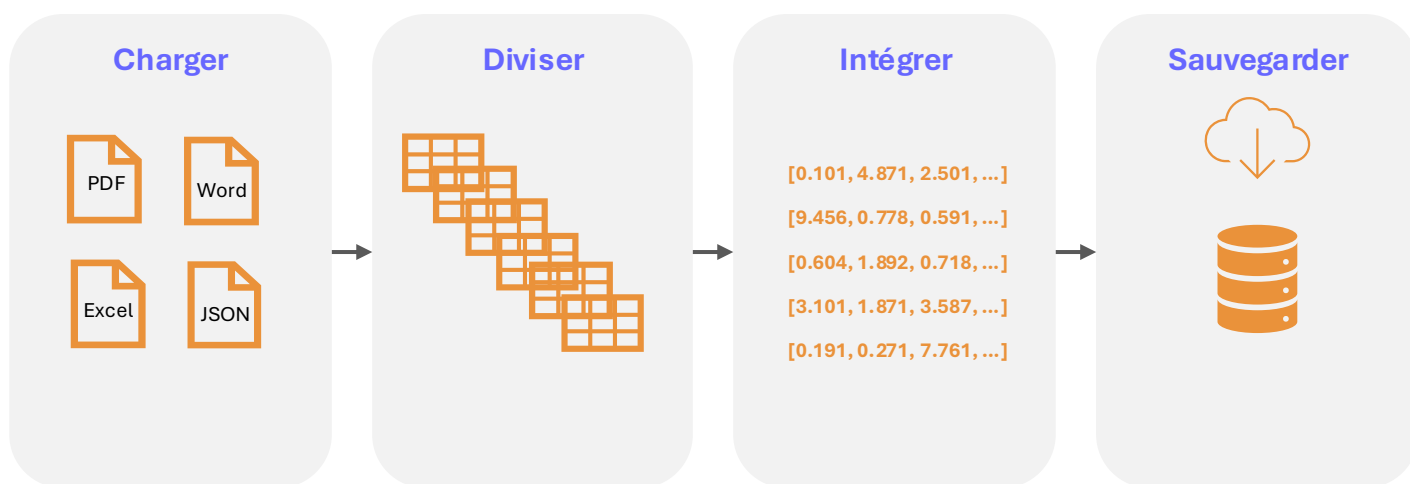
Cette analyse fournit des informations détaillées sur les caractéristiques des données, telles que le nombre de caractères, de mots, ou la distribution des termes clés. L'EDA aide également à identifier les éventuelles incohérences ou anomalies dans les données. Elle permet ainsi d'obtenir une vue d'ensemble pour mieux structurer les données pour les phases de traitement ultérieures.

3 Préparation et segmentation du texte

Les modèles de langage ne sont pas capables de comprendre directement le texte brut. Il est donc nécessaire de le convertir en une forme numérique. Cette conversion commence par le découpage du texte en segments appelés "chunks". Ce retraitement des données constitue une part significative de l'effort total et a un impact important sur les résultats finaux.

La segmentation du texte en chunks facilite la gestion des morceaux de texte de taille similaire, assurant une cohérence dans le traitement des données. La fenêtre contextuelle du modèle de langage, c'est-à-dire la quantité de tokens qu'un LLM peut traiter en une seule fois est limitée et nécessite des ressources de calcul. Plusieurs stratégies de segmentation peuvent être utilisées, telles que Recursive Character Text Splitter, Token splitter, Sentence Splitter ou encore Regex Splitter. Un texte bien nettoyé et structuré génère généralement de meilleurs résultats.

En segmentant correctement les textes, on prépare le terrain pour la création des embeddings. La manière idéale de traiter le texte avant l'intégration est encore un domaine de recherche actif.



4 Génération des embeddings et stockage

Pour représenter chaque chunk sous forme numérique, des modèles d'embeddings créent des vecteurs sous la forme de séquences de nombres organisées. Ces vecteurs permettent au modèle de langage de traiter et de comprendre les informations textuelles.

De nombreux modèles d'embeddings sont disponibles sur des plateformes comme Hugging Face, qui offre notamment un leaderboard pour suivre les modèles existants. Les méthodes d'embeddings sont des étapes cruciales dans le traitement des données, bien qu'elles ne soient pas des méthodes d'apprentissage complètes. Une fois les embeddings créés, ils doivent être stockés dans une base de données vectorielle avec des métadonnées descriptives pour chaque vecteur. Des solutions comme ChromaDB, Pinecone ou Faiss sont couramment utilisées pour gérer ces bases de données vectorielles et offrent des performances optimisées pour les requêtes de similarité en facilitant l'intégration avec les modèles de langage.

Lors de la sélection d'un modèle d'embedding, plusieurs aspects essentiels doivent être considérés pour garantir des performances optimales. Pour les applications nécessitant l'embedding de longues séquences de texte, il est crucial de choisir un modèle avec une capacité d'entrée adéquate. Un vecteur d'embedding de grande dimension représente mieux les nuances du texte, mais nécessite plus de calculs et de stockage. Les modèles de grande taille produisent des embeddings de meilleure qualité, mais demandent plus de puissance de calcul et de temps, nécessitant un choix adapté aux capacités matérielles de l'entreprise.

5 Intégration du modèle de langage avec la base de données



Il est essentiel de lier le modèle de langage et la base de données vectorielle. Des frameworks comme **LangChain** simplifient considérablement cette intégration en offrant divers composants adaptés à ces applications. Les entreprises peuvent rapidement mettre en œuvre des chatbots IA performants et adaptés à leurs besoins spécifiques, optimisant ainsi leurs interactions automatisées et améliorant l'efficacité opérationnelle.

Un chatbot IA doit récupérer l'historique des conversations et être configuré avec des éléments de contexte appropriés. Attention, l'ajout excessif de contexte peut entraîner des hallucinations, où le modèle génère des informations incorrectes ou non pertinentes.

Le processus de développement suit ces étapes : l'utilisateur fait une requête qui est intégrée dans un texte contenant l'historique de la conversation et des éléments de contexte définis. Le modèle de langage utilise son embedding pour traduire cette requête en vecteur, en utilisant le même processus que celui pour la construction des vecteurs de chunks. Le modèle recherche ensuite dans la base de données vectorielle les vecteurs les plus similaires à la requête, en utilisant des méthodes telles que le produit scalaire, la similarité cosinus, la distance de Manhattan ou Euclidienne afin d'obtenir des scores de similarité. Les scores sont triés par ordre décroissant de similarité. Le modèle identifie alors les meilleures réponses et leurs métadonnées correspondantes. Enfin, le modèle génère une réponse augmentée basée sur les données récupérées et son corpus de connaissances.

6 Déploiement avec une interface graphique

Pour déployer rapidement un chatbot avec une interface graphique et tester son efficacité, des outils comme Streamlit ou Gradio sont utiles. Ils permettent de créer des interfaces graphiques intuitives sans expertise approfondie en développement web. Pour des projets nécessitant plus de personnalisation, des frameworks web en Python comme Flask ou Django peuvent être utilisés.

Dans ce cas d'usage, des textes et des documents PDF ont été transformés en vecteurs grâce au **modèle d'embedding d'OpenAI**, puis stockés dans une base de données vectorielle de **Pinecone**. Le modèle d'OpenAI est appelé via une **chaîne** construite avec le **framework LangChain**, facilitant ainsi la gestion des interactions entre le modèle de langage et la base de données. Le déploiement a été réalisé avec **Streamlit**. Le chatbot utilise cette configuration pour fournir une réponse précise et pertinente. Cette méthode exploite pleinement les capacités des modèles de langage avancés tout en intégrant des données spécifiques et actualisées. Cela maximise l'efficacité des interactions automatisées et améliore l'expérience utilisateur.

GPT-4o ❌

RAG avec GPT-3.5 ✅

What does VSME and ESRS mean ?

Sure, here are the explanations for VSME and ESRS:

1. VSME:

- **Voltage Supervision and Monitoring Equipment:** This typically refers to devices or systems used to monitor and supervise the voltage levels in electrical systems, ensuring stability and safety.
- **Very Small Manufacturing Enterprise:** In a business context, this term is often used to categorize enterprises based on their size, specifically those with a very small number of employees or low revenue compared to larger firms.

2. ESRS:

- **European Sustainability Reporting Standards:** These standards are part of the European Union's Corporate Sustainability Reporting Directive (CSRD). They aim to ensure that companies report reliable and comparable sustainability information.
- **Emergency Services Radio System:** This refers to communication systems used by emergency services (like police, fire, and medical services) to coordinate and communicate during emergencies.

If you need information on a specific context or industry where these acronyms are used, please let me know!

🗨️ 🔄 📄 🌐 🛠️

Enter your OpenAI API key

✓ API key already provided!

Clear chat history

ESG OpenIA RAG Bot

What does VSME and ESRS mean ?

VSME stands for "Voluntary Sustainability Management Education" and refers to educational programs or initiatives that focus on teaching businesses and organizations about sustainable management practices.

ESRS stands for "Environmental, Social, and Governance Reporting System" and refers to a framework or system used by companies to report on their environmental, social, and governance performance. This can include information on their sustainability efforts, social impact, and corporate governance practices.

Type your message here...