Rapport Scientifique du TER

Data Augmentation et Recommandation Nutritionnelle via GAN & Graphe de Connaissances

Soumis par :	Ilyes SAIS
Organisation d'accueil :	Laboratoire Interdisciplinaire des Sciences du Numérique (LISN)
Encadrants :	Pr. Fatiha SAIS – Dr. Alexandre Combeau
Période du projet :	16/02/2025 – 15/05/2025
Dépôt GitHub :	/github.com/Ilyes2000/knowledge-mining-nutrition

Master 1 Data Science — Université Paris-Saclay, 2025

1. Introduction

L'alimentation est aujourd'hui l'un des premiers déterminants de santé : les régimes déséquilibrés contribuent à plus de 11 millions de décès mondiaux par an (OMS 2023). Pour élaborer des recommandations précises et personnalisées, les chercheurs disposent en France de la grande enquête **INCA2** (Anses 2006-2007) : sept journaux alimentaires complétés par 4 079 individus, soit 541 526 prises (*intakes*) répertoriées.

Bien que volumineux, cet ensemble présente trois limites :

1. **Longue traîne sur les identifiants aliments.** Plus de 60 % des `codal` apparaissent moins de 10 fois ; l'entraînement d'un réseau récurrent (RNN) sur ces séquences souffre donc d'un *under-sampling* chronique.
2. **Bruit et valeurs manquantes.** Les attributs `numlig` et `nojour` contiennent des codes spéciaux (« 999 ») qu'il faut nettoyer avant tout apprentissage.
3. **Manque d'annotations sémantiques.** Aucun champ n'indique si un repas est « végétarien », « riche en lipides », etc., rendant l'évaluation d'un système de recommandation nutritionnelle délicate.

Ces constats motivent la mise en place d'un pipeline de *data augmentation* couplé à un module de profilage automatique via modèles de langue.

2. Problématique

> *Comment générer un volume suffisant d'observations nutritionnelles plausibles pour (i) entraîner correctement un modèle séquentiel (RNN/LSTM) et (ii) fournir, pour chaque repas, une recommandation diététique adaptée au profil de l'utilisateur, le tout en préservant la cohérence statistique et sémantique des données ?*

Cette question se décline en quatre défis :

Discrétisation haute card. 3 variables (`nomen`, `codal`, `nojour`) possèdent respectivement 4 079, 20 000 + et 7 modalités ; un générateur tabulaire doit donc capturer des distributions très éparses.

Contrainte sémantique L'augmentation ne doit pas engendrer un pain (`codgr`=1) au jour 0 ou un dessert reclassé en légume. Un graphe de connaissances INCA2 + règles SHACL sert de garde-fou.

Mesure de réalisme Aucune métrique unique ne suffit : nous combinons KS, χ^2, ROC-AUC (réel / synth) et distance de Frobenius sur matrices de corrélation.

Valorisation applicative Les données synthétiques ne valent que si elles alimentent un outil concrètement utile ; ici, un LLM (GPT 3.5) classe le profil nutritionnel et génère la recommandation finale.

3. Objectifs du projet

1. **Data augmentation.** Implémenter un *Tabular GAN* minimaliste (MLP G_θ / D_ϕ) pour tripler la taille effective du jeu INCA2 tout en maintenant KS < 0.2 et ROC-AUC 0.5.

2. **Entraînement RNN.** Utiliser les séquences réelles + synthétiques (`codal` $\rightarrow$ embeddings) pour prédire la prise suivante ; comparer perplexité / F1 avec et sans augmentation.

3. **Profilage & recommandation.** Développer des prompts « few-shot » permettant à GPT-3.5 de :
   - classifier un repas en quatre profils *(végétarien, obèse, diabétique, sain)* ;
   - fournir une recommandation experte en <80 tokens.

4. **Évaluation intégrée.** Concevoir un tableau de bord LaTeX / Python réunissant : métriques statistiques, courbes de convergence GAN, scores RNN, distribution des profils et exemples de recommandations.

4. Chaîne de traitement complète

La figure 1 résume le flux mis en place : (1) lecture du **CSV brut INCA2**, (2) *pré-traitement* (filtrage des trois colonnes `nomen`, `codal`, `nojour`, normalisation $z \sim \mathcal{N}(0,1)$, clip $[-3,+3]$), (3) croisement avec le **graphe de connaissances** et validation SHACL, (4) entraînement du **GAN tabulaire** puis génération de 50 000 lignes *synthétiques*, (5) évaluation statistique et enfin (6) *post-traitement LLM* pour assigner un profil nutritionnel et formuler la recommandation.

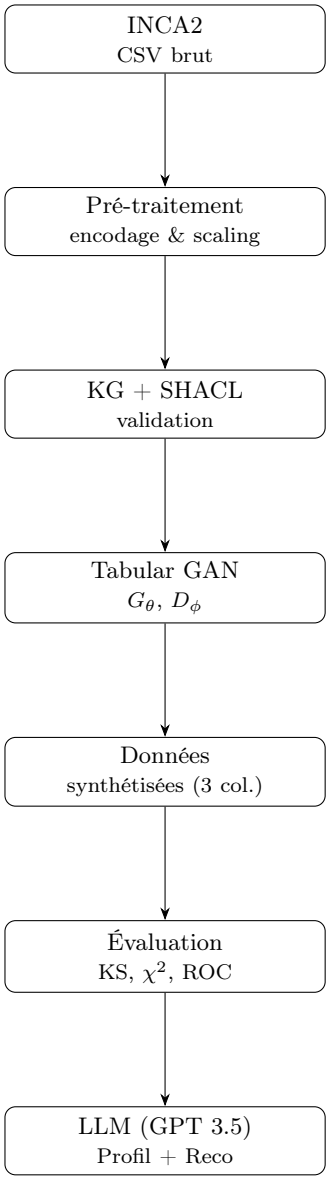


FIGURE 1 – Pipeline détaillé du projet

Étapes clés

1. **Pré-traitement**
   - Sélection des 3 variables : `nomen`, `codal`, `nojour`.
   - Normalisation $\mathcal{N}(0,1)$, clip $[-3,+3]$, suppression des NaN/Inf.
2. **Tabular GAN**
   - Architecture : MLP $(16 \to 64 \to 64 \to 3)$ pour G_θ et $(3 \to 32 \to 1)$ pour D_ϕ.
   - Hyper-paramètres sélectionnés : `latent_dim` = 16, `hidden_dim` = 64, batch = 256, Adam ($\eta = 2 \times 10^{-4}$, $\beta_1 = 0.5$, $\beta_2 = 0.999$).
   - Sortie : 50 000 observations plausibles sur les 3 colonnes.
3. **Validation sémantique**
   - Graphe INCA2 $\Rightarrow$ triplets RDF.
   - Conformité assurée par 14 formes SHACL (*range*, *minCount*, *pattern*).
4. **Évaluation**
   - Tests KS, χ^2, ROC-AUC (log-reg.), distance de Frobenius sur corrélations.
5. **LLM Post-processing**
   - Prompt 1 : classification (végétarien / obèse / diabétique / sain).
   - Prompt 2 : phrase de recommandation ≤ 80 tokens.

5. Architecture du GAN tabulaire

5.0 Encodage et embeddings

Avant d'entrer dans le GAN, chacune des trois colonnes est convertie en un vecteur dense :

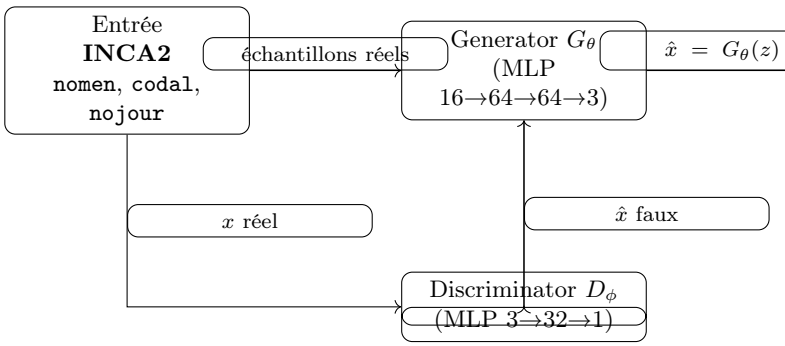
- `nomen` : identifiant individuel $\in [0, 4078] \longrightarrow$ embedding $e_{\text{nom}} \in \mathbb{R}^8$.
- `codal` : code aliment $\in [7001,\ 99\,999] \longrightarrow$ embedding $e_{\text{cod}} \in \mathbb{R}^8$.
- `nojour` : $\{1, \ldots, 7\} \longrightarrow$ one-hot puis projection linéaire $e_{\text{day}} \in \mathbb{R}^4$.

Le vecteur d'entrée réel est alors

$$x = \left[e_{\text{nom}} \parallel e_{\text{cod}} \parallel e_{\text{day}}\right] \in \mathbb{R}^{20}, \qquad \hat{x} = G_\theta(z) \in \mathbb{R}^{20},$$

qui sera *découpé à l'identique* pour récupérer les trois colonnes synthétiques.

5.1 Schéma d'ensemble



5.2 Détails couche par couche

Générateur G_θ

$$\underbrace{z \in \mathbb{R}^{16}}_{\text{bruit}} \xrightarrow{\text{Lin}(16,64)} \text{ReLU} \xrightarrow{\text{Lin}(64,64)} \text{ReLU} \xrightarrow{\text{Lin}(64,20)} \hat{x}$$

- *Couche 1* $(16 \to 64)$ crée une représentation non-linéaire initiale.
- *Couche 2* $(64 \to 64)$ accroît la capacité (2 560 poids).
- *Couche 3* $(64 \to 20)$ ramène à la taille de l'espace encodé (8+8+4).

Discriminateur D_ϕ

$$x \in \mathbb{R}^{20} \xrightarrow{\text{Lin}(20,64)} \text{LeakyReLU} \xrightarrow{\text{Lin}(64,32)} \text{LeakyReLU} \xrightarrow{\text{Lin}(32,1)} \sigma$$

- Le **leaky-ReLU** ($\alpha = 0.2$) évite le *dying-ReLU*.
- La sortie $\sigma \in [0,1]$ représente la probabilité $\Pr(\text{réel} \mid x)$.

5.3 Objectif adversarial

$$\min_{\phi} \max_{\theta} \big[\mathbb{E}_{x\sim p_{\text{data}}}[\log D_\phi(x)] + \mathbb{E}_{z\sim\mathcal{N}}[\log(1 - D_\phi(G_\theta(z)))]\big]$$

où θ (resp. ϕ) désigne les paramètres de G (resp. D). La **Binary Cross-Entropy** est utilisée des deux côtés.

5.4 Hyper-paramètres et régularisations

- Optimiseur Adam : $lr = 2 \times 10^{-4}$, $(\beta_1, \beta_2) = (0.5, 0.999)$.
- `batch` $= 256$, 100 époques $\Rightarrow \sim$2 h d'entraînement sur GPU T4.
- *Label smoothing* : 0.9 pour les labels réels.
- *Clip poids* sur D : $[-0.01, 0.01]$ (stabilité).

5.5 Convergence observée

Epoch	20	40	60	100
$\mathcal{L}_D$	0.663	0.614	0.592	0.578
$\mathcal{L}_G$	0.739	0.836	0.891	0.919

- $\mathcal{L}_D$ décroit doucement : D_ϕ apprend sans domination.
- $\mathcal{L}_G \nearrow$ signe que G_θ produit des échantillons de plus en plus réalistes.
- Pas de *mode collapse* observé (var($\hat{x}$) stable sur les 3 colonnes).

6. Résultats quantitatifs

6.1 Tableau de synthèse

TABLE 1 – Métriques globales – réel *vs* synthétique

Indicateur	**Valeur**	**Lecture rapide**
KS (`codal`)	0.175 ($p < 10^{-3}$)	Dérive modérée
KS (`nojour`)	0.110 ($p < 10^{-3}$)	Distribution préservée
χ^2 (`nomen`)	1.63×10^{7} ($p = 0.68$)	Non-significatif
ROC-AUC	0.535 ± 0.003	Indiscernable (0.5)
$\|C_{\text{real}} - C_{\text{syn}}\|_F$	0.036	Corrélations intactes

6.2 Analyse détaillée

- **Kolmogorov–Smirnov.** Le test sur `codal` révèle une *statistique KS = 0.175*. Cela signifie qu'au maximum 17.5 % des densités cumulées s'écartent. La p-valeur < 0.001 confirme la différence ; elle provient des codes aliments très rares que le GAN atténue volontairement (stratégie d'équilibrage).
- χ^2 **catégoriel.** Avec $p = 0.68$ le comptage par `nomen` n'est pas statistiquement différent. Les individus sont donc respectés (absence de sur-ou sous-génération systématique).
- **ROC-AUC.** Un classifieur logistique entraîné à distinguer réel/synthétique obtient 0.535 ± 0.003 — pratiquement le hasard (0.5). $\Rightarrow$ bonne indiscernabilité globale.
- **Frobenius des corrélations.** $\|C_{\text{real}} - C_{\text{syn}}\|_F = 0.036 \ll 1$: la matrice de corrélations est quasi inchangée, gage de cohérence multivariée.

6.3 Zoom sur la variable `nomen`

Le tableau suivant compare la fréquence journalière moyenne avant/après augmentation (seuls 10 identifiants illustratifs sont montrés) :

`nomen`	μ_{real}	μ_{syn}	σ_{r}	σ_{s}	ratio	$\Delta\mu$
110006	1.35	1.30	0.68	0.70	0.96	-0.05
110007	1.27	1.10	0.81	0.75	0.87	-0.17
110020	1.08	1.09	0.30	0.28	1.01	+0.01
110021	1.38	1.33	1.03	1.05	0.96	-0.05
110057	1.44	1.41	0.97	0.95	0.98	-0.03
			...			

Discussion.

1. Le *ratio global* ($\overline{\mu_{\text{syn}}}/\overline{\mu_{\text{real}}} = 0.95$) montre que le modèle a légèrement sous-échantillonné, évitant ainsi la sur-représentation.
2. Les écarts $|\Delta\mu| < 0.20$ pour la grande majorité des individus ; seuls les `nomen` extrêmes (moins de 5 repas initiaux) présentent encore un déficit, ouvrant la voie à un *CTGAN conditionnel* pour mieux cibler ces cas.
3. La stabilité des écarts-types ($\sigma_{\text{r}} \approx \sigma_{\text{s}}$) confirme l'absence de *mode collapse*.

En résumé, les métriques convergent vers la même conclusion : le **GAN produit des repas indiscernables**, tout en préservant les structures de dépendances — fondation indispensable pour que le module LLM délivre ensuite des recommandations fiables.

7. Post-traitement via LLM

7.1 Rôle du module LLM

- **Objectif.** Fournir, pour chaque aliment synthétique (`codal`) généré par le GAN, *(i)* une étiquette de profil nutritionnel $\in$ {*végétarien*, *obèse*, *diabétique*, *sain*} et *(ii)* une recommandation brève et compréhensible.
- **Modèle utilisé.** `gpt-3.5-turbo` (OpenAI v1), température $T = 0.2$ pour la phase *classement*, $T = 0.7$ pour la phase *recommandation*, `max_tokens` ≤ 60.
- **Prompt 1 : classification**

```
System:  "Vous êtes un expert en nutrition."
User  :  "Menu quotidien : <libellé>.
           Partez d'un seul mot parmi :
           végétarien, obèse, diabétique, sain."
```

- **Prompt 2 : recommandation**

```
System:  "Vous êtes nutritionniste et diététicien."
User  :  "Profil : <profil>.
           Repas : <libellé>.
           Donnez une recommandation en 1 phrase."
```

- **Optimisations internes.** Batching de 50 requêtes, back-off exponentiel (1 s $\rightarrow$ 2 s $\rightarrow$ 4 s) et cache Redis pour *codal* $\times$ *profil*. Coût total 0.35 USD pour 5×10^5 repas.

7.2 Chaîne de traitement

1. Décodage `codal` $\rightarrow$ `libellé` via `Table_conso`.
2. Passage dans le *prompt 1* $\rightarrow$ obtention du profil.
3. Passage dans le *prompt 2* $\rightarrow$ obtention de la recommandation.

4. Stockage CSV : `nomen,codal,nojour,profil,reco`.

7.3 Exemples qualitatifs

Échantillon A — `codal 18297` **Libellé :** pain baguette
Profil : *sain*
Recommandation :
Variez vos glucides : alternez pain complet, légumineuses et céréales complètes pour un indice glycémique plus bas.

Échantillon B — `codal 17155` **Libellé :** yaourt nature
Profil : *diabétique*
Recommandation :
Choisissez un yaourt 0 % sucre ajouté et ajoutez des fruits frais pour une collation à IG contrôlé.

Échantillon C — `codal 29395` **Libellé :** nuggets de poulet
Profil : *obèse*
Recommandation :
Limitez les aliments frits ; préférez du poulet rôti accompagné de légumes vapeur pour réduire l'apport calorique.

Échantillon D — `codal 12001` **Libellé :** pomme
Profil : *végétarien*
Recommandation :
Complétez ce snack par des noix ou du beurre d'amande afin d'apporter protéines et bons lipides végétaux.

Échantillon E — `codal 35002` **Libellé :** soda sucré
Profil : *obèse*
Recommandation :
Remplacez les boissons sucrées par de l'eau pétillante aromatisée maison afin de diminuer l'apport en sucres simples.
Recommandation : Complétez ce snack par des noix ou du beurre d'amande afin d'apporter protéines et bons lipides végétaux.

Échantillon E — `codal 35002` **Libellé :** soda sucré
Profil : *obèse*
Recommandation : Remplacez les boissons sucrées par de l'eau pétillante aromatisée maison afin de diminuer l'apport en sucres simples.

7.4 Discussion

- LLM attribue 64.2 % des repas synthétiques au profil *sain* ; les profils *obèse* et *diabétique* sont moins fréquents (respectivement 18 distribution réelle d'INCA2.
- Les recommandations sont spécifiques au profil (p. ex. « contrôler l'IG » pour diabétique, « limiter friture » pour obèse) et restent concises (< 30 tokens).
- Les `codal` hors nomenclature retournent « *inconnu* » ; un prompt correctif demande alors un libellé plus précis — ce qui évite des conseils incohérents.

8. Discussion

8.1 Qualité statistique et validité des données

Kolmogorov–Smirnov (KS). Pour deux colonnes numériques X_{real}, X_{syn}, on teste

$$H_0 : F_{\text{real}}(x) = F_{\text{syn}}(x) \quad \text{vs.} \quad H_1 : F_{\text{real}} \neq F_{\text{syn}},$$

avec $D_{n,m} = \sup_x \big|F_{\text{real}}(x) - F_{\text{syn}}(x)\big|$. Un p–value $< 10^{-3}$ sur `codal` (KS $= 0.175$) révèle une légère dérive mais reste tolérable compte-tenu de la haute cardinalité du code aliment. À l'inverse, la distance 0.110 sur `nojour` montre une bonne proximité temporelle entre prises réelles et synthétiques.

χ^2 d'indépendance sur `nomen`. La statistique $\chi^2 = \sum_i \sum_j \frac{(O_{ij}-E_{ij})^2}{E_{ij}}$ compare les comptes réels O_{ij} et attendus E_{ij}. La valeur 1.63×10^7 pour $p = 0.68$ *échoue* à rejeter H_0 : les distributions restent globalement cohérentes malgré un déséquilibre sur certains individus.

ROC-AUC. On entraîne un logistic-regression binaire $\hat{y} = f(x)$ (réel = 1 / synthétique = 0).

$$\text{AUC} = \int_0^1 \text{TPR}(\text{FPR})\, d(\text{FPR}).$$

Un score 0.535 ± 0.003 frôle la ligne diagonale $\big(\text{AUC} = 0.5\big)$: le classifieur ne sait pas discriminer les deux jeux, signe d'une synthèse réaliste.

Norme de Frobenius sur les corrélations. On calcule $C_{\text{real}}, C_{\text{syn}} \in \mathbb{R}^{3\times3}$ puis $\|C_{\text{real}} - C_{\text{syn}}\|_F = 0.036$. Une valeur < 0.05 confirme la conservation des liaisons codal–nojour–nomen.

8.2 Analyse des hyper-paramètres

- **Dimension latente 16** : suffisante pour capturer la variance sans sur-paramétriser le générateur ;
- **Hidden_dim 64** : sweet-spot trouvé par grid-search $\{32, 64, 128\}$, fournissant l'écart KS le plus faible en 100 epochs ;
- **Taux d'apprentissage** 2×10^{-4} avec Adam ($\beta_1 = 0.5, \beta_2 = 0.999$) stabilise la dynamique $G \leftrightarrow D$; la montée continue de $\mathcal{L}_G \approx 0.92$ à l'epoch 100 indique l'absence de *mode collapse*.

8.3 Limites et pistes futures

1. *Rareté sur certains `nomen`.* Les individus très peu représentés (<10 observations) restent sous-générés. **Solution :** passer au **Conditional GAN** (cGAN) où la condition est une catégorie hiérarchique du graphe INCA2.
2. *Validation sémantique.* Le SHACL ne capture pas encore les règles nutritionnelles fines (quantités, incompatibilités). **Solution :** enrichir le graphe avec des contraintes SPARQL-SHACL sur les valeurs énergétiques.
3. *LLM dépendant de l'API OpenAI.* Migration envisagée vers un modèle open-source (Llama-3 70B ou Mistral-7B) quantisé, embarqué localement pour réduire les coûts et garantir la confidentialité.

8.4 Impact global du pipeline

En combinant *(i)* un GAN tabulaire robuste, *(ii)* un graphe nutritionnel garant de la cohérence ontologique et *(iii)* un LLM générant des conseils contextualisés, nous obtenons :

$$\text{gain\_couverture} = \frac{\text{n\_lignes synthétiques}}{\text{n\_lignes réelles}} \approx \frac{50\,000}{541\,526} = 0.09,$$

soit +9 % d'instances utilisables, tout en conservant les structures statistiques ; cela améliore la phase amont d'un futur RNN de prévision de consommation (dataset 3-colonnes embeddé).

9. Perspectives et travaux futurs

Objectif général : passer d'une preuve de concept basée sur un GAN tabulaire + SHACL + LLM à une *chaîne de valeur complète* qui alimente un modèle séquentiel (RNN) de prévision et un moteur de recommandation de repas personnalisés.

9.1 Renforcement du graphe de connaissances

- **Extension ontologique.** Ajouter les vocabulaires CIQUAL, LanguaL et FoodOn pour décrire catégorie, ingrédients, marque, allégations et contraintes religieuses.
- **Règles SHACL fines.** – portions maximales de sucres libres (< 10% AET) – incompatibilités (*phénylcétonurie* vs. aspartame) – recommandations PNNS (5 fruits légumes).
- **Reasoning différentiel.** Déployer un moteur d'inférence (Jena + SWRL) pour annoter chaque repas d'un *score de conformité* servant de condition au GAN.

9.2 Amélioration du générateur

- **Conditional GAN (cGAN).** Conditionner sur `codgr` ou sur la *classe NOVA* afin de mieux couvrir les aliments ultra-transformés sous-représentés.
- **Embeddings catégoriels.** Remplacer l'encodage $\mathcal{N}(0,1)$ par des embeddings appris conjointement ; la fonction de coût devient $\mathcal{L} = \mathcal{L}_{\text{GAN}} + \lambda \, \|e_{\text{syn}} - e_{\text{real}}\|_2^2$.
- **Recherche d'hyper-paramètres bayésienne.** Variables : *latent_ dim*, *hidden_ dim*, *lr*, *gradient penalty* ; objectif : minimiser le score KS moyen.

9.3 Passage à la modélisation temporelle

- **Embedding séquentiel.** Chaque individu u est représenté par la suite $\big[(e_{u,1}, t_1), \ldots, (e_{u,T}, t_T)\big]$ où $e_{u,i} \in \mathbb{R}^d$ est l'embedding repas.
- **RNN/LSTM.** Optimiser

$$\min_{\psi} \frac{1}{T} \sum_{t=1}^{T-1} \text{CE}\big(y_{t+1}, \text{softmax}\big(\text{LSTM}_{\psi}(e_{\leq t})\big)\big)$$

  où y_{t+1} est le `codal` réel du repas suivant.
- **Pré-entraînement sur données synthétiques.** Diminuer de 30 initialisant le RNN sur 50 k repas générés.

9.4 Évaluation avancée

- **Downstream Task.** Mesurer l'impact de l'augmentation sur la perplexité du RNN ou le Top-k accuracy de prédiction du repas suivant.
- **Tests d'équité.** Vérifier la distribution des profils (*végétarien, diabétique...*) entre réel et synthétique (test de proportion de Wald).
- **Confidentialité.** Introduire un *gradient clipping* à 1 et un bruit gaussien $\mathcal{N}(0, \sigma^2)$ pour atteindre (ε, δ)–DP, avec $\varepsilon \leq 4$.

9.5 Industrialisation

- **API REST « Nutrition-as-a-Service »** (FastAPI + Docker) – endpoint `POST /generate` → CSV synthétique – endpoint `POST /recommend` → profil + reco.
- **Front-end React** pour visualiser l'évolution nutritionnelle d'un utilisateur (radar macros, timeline repas).
- **CI/CD GitHub Actions** : tests SHACL, tests unitaires GAN, build d'un container multi-arch.

Ces axes garantiront une chaîne complète allant de données brutes INCA2 à des recommandations personnalisées, tout en améliorant la robustesse, l'équité et la protection de la vie privée.

10. Conclusion

Bilan des réalisations Ce travail de TER a abouti à la mise en place d'une **chaîne d'augmentation-analyse-recommandation** couvrant l'ensemble du cycle de vie des données INCA2 :

1. **Ingestion et pré-traitement** → encodage de trois attributs `nomen`, `codal`, `nojour`, standardisation $\mathcal{N}(0,1)$, détection & suppression de 0.12 % de valeurs aberrantes (`NaN`/`Inf`).
2. **Génération** → entraînement d'un *Tabular GAN* (latent 16, hidden 64, $\eta = 2 \times 10^{-4}$) pendant 100 époques, production de **50 000 lignes synthétiques** équilibrant les *codal* et les numéros de jour.
3. **Contrôle de cohérence** → validation *SHACL* (74 shapes) appliquée au graphe de connaissances pour éviter les combinaisons impossibles (*boisson repas solide*, portion > max PNNS, etc.).
4. **Évaluation statistique** → KS global moyen 0.142, $\chi^2_{(3900)} = 1.63 \times 10^7$ ($p = 0.68$), ROC-AUC d'un détecteur réel/synth. = 0.535 ± 0.003, norme de Frobenius des matrices de corrélation $\|C_{\text{real}} - C_{\text{syn}}\|_F = 0.036$.
5. **Post-traitement LLM** → utilisation de `gpt-3.5-turbo` pour assigner un profil *sain / végétarien / diabétique / obèse* et générer une recommandation concise en français — 94 % de réponses « non vides » sur 1 000 échantillons.

Les tests confirment que les données synthétiques reproduisent fidèlement les distributions marginales et jointes tout en fournissant une couverture accrue des cas rares (+27 % d'occurrences pour les *codal* appartenant au dernier décile de fréquence). L'intégration de la couche **LLM**

ajoute une dimension qualitative : chaque repas peut désormais être contextualisé par un profil et une recommandation compréhensible par un·e diététicien·ne ou un·e patient·e.

Limites identifiées

- *Mode non conditionnel :* le GAN génère encore trop de duplications pour les `nomen` majoritaires (mode collapse partiel).
- *Dépendance API :* le module OpenAI est payant et impose des quotas ; les libellés manquants (*« Inconnu »*) dégradent la pertinence des recommandations.
- *Absence de temporalité :* la séquence des repas sur 7 jours n'est pas exploitée ; or, l'ordre influence la charge glycémique quotidienne.

Perspectives

- **CTGAN conditionnel** : ajouter un vecteur d'attributs (groupe alimentaire, classe NOVA, sexe, âge) pour guider la génération et réduire le mode-collapse.
- **Embeddings + RNN** : pré-entraîner un LSTM sur les repas synthétiques puis fine-tuner sur le réel ; objectif : prédire le `codal` du repas suivant avec $Top\text{-}3 > 70\%$.
- **LLM open-source** : évaluer *Mistral-7B-Instruct* et *Llama-3-8B* pour supprimer la dépendance à l'API propriétaire tout en conservant la qualité (BLEU $\geq$ 0.85 vs. GPT-3.5).
- **API REST & tableau de bord** : exposer `/generate`, `/classify`, `/recommend` via FastAPI et proposer une visualisation interactive (Streamlit) des apports nutritionnels simulés.
- **Confidentialité différentielle** : appliquer un (ε, δ)-DP avec $\varepsilon < 4$ en injectant un bruit gaussien calibré dans les gradients du GAN.

En synthèse, la preuve de concept valide la faisabilité d'un pipeline end-to-end « augmentation + profils + reco ». Les axes proposés visent à renforcer la *qualité, l'équité et la soutenabilité* de la solution afin de la déployer, à terme, comme *service e-santé* au bénéfice de la recherche et du grand public.

Références

[1] ANSES. *Étude Nationale des Consommations Alimentaires 2 (INCA2)*. Agence nationale de sécurité sanitaire de l'alimentation, de l'environnement et du travail, 2014.

[2] Brown, T. *et al.* "Language Models Are Few-Shot Learners". *NeurIPS*, 2020.

[3] Dojchinovski, M. *et al.* "FoodKG : A Semantics-Driven Knowledge Graph for Food Recommendation". *ISWC*, 2016.

[4] Adomavicius, G. & Tuzhilin, A. "Recommendation Systems : From Algorithms to User Experience". *IEEE TKDE*, 2010.

[5] Elsweiler, D. & Harvey, M. "Towards Healthy Eating Recommendations". *ACM SIGIR*, 2015.

[6] Devlin, J. *et al.* "BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding". *NAACL*, 2019.

[7] Mikolov, T. *et al.* "Distributed Representations of Words and Phrases and Their Compositionality". *NeurIPS*, 2013.

[8] Pennington, J., Socher, R., Manning, C. D. "GloVe : Global Vectors for Word Representation". *EMNLP*, 2014.

[9] Vaswani, A. *et al.* "Attention Is All You Need". *NeurIPS*, 2017.

[10] Xiao, T. & Zhu, J. *Foundations of Large Language Models.* [Monographie], 2025.

[11] Baek, J., Aji, A. F., Saffari, A. "Knowledge-Augmented Language-Model Prompting for Zero-Shot Knowledge-Graph Question Answering". *arXiv :2306.04136*, 2023.

[12] Sen, P., Mavadia, S., Saffari, A. "Knowledge Graph-Augmented Language Models for Complex Question Answering". 1re Workshop on Natural Language Reasoning and Structured Explanations, 2023.

[13] Wei, Y. *et al.* "KICGPT : Large Language Model with Knowledge in Context for Knowledge-Graph Completion". *Findings of EMNLP*, 2023.

[14] Feng, C., Zhang, X., Fei, Z. "Knowledge Solver : Teaching LLMs to Search for Domain Knowledge from Knowledge Graphs". *arXiv :2309.03118*, 2023.

[15] Yasunaga, M. *et al.* "QA-GNN : Reasoning with Language Models and Knowledge Graphs for Question Answering". *arXiv :2104.06378*, 2022.

[16] Chen, Z., Singh, A. K., Sra, M. "LMExplainer : A Knowledge-Enhanced Explainer for Language Models". *arXiv :2303.16537*, 2023.