CIV6707A - Leçons apprises sur la génération artificielle de données OD

5 décembre 2024

Table des matières

1 Introduction

L'objectif de cette portion du projet est de pouvoir générer des déplacements artificiellement, en se basant sur des données O-D. Pour simplifier notre tâche, nous sélectionnons les informations suivantes disponibles dans les données:

- Sexe de la personne
- Groupe d'âge de la personne
- Heure de départ du déplacement
- 4 coordonnées, soit les coordonnées x et y du départ et de l'arrivée
- Mode de transport utilisé

Le but de notre réseau de neurones est donc d'utiliser ces données, comprendre la structure derrière celles-ci, et pouvoir générer artificiellement des nouveaux déplacements conservant cette structure.

2 Utilisation d'un GAN

Cette section documente la méthode derrière le fichier **GAN.ipynb** qui est utilisé pour générer les données.

Nous avons essayé d'utiliser un *Generative Adverserial Network* (GAN). Un GAN fonctionne en utilisant deux réseaux de neurones. Le premier, le discriminateur, a pour but de pouvoir différencier de vraies et de fausses données. Ainsi, son entrée est un vecteur de données et sa sortie est une valeur binaire déterminant si le discriminateur pense que ce vecteur correspond à la catégorie des fausses ou des vraies données.

Le deuxième réseau de neurones s'agit du générateur. Son but est de synthétiser des données à partir d'un vecteur de bruit de l'espace latent. Ainsi, son entrée est un vecteur de bruit et sa sortie est un vecteur de données.

Durant l'entrainement, le générateur essaie de faire croire au discriminateur qu'il est en train de générer de vraies données. Les vecteurs de données sont tous associés à une valeur binaire dépendant de s'ils sont générés ou originaux. Ainsi, lors de l'entrainement du discriminateur, on lui associe une valeur de perte en fonction du nombre de d'entrées de données qu'il arrive a différencier. Plus il arrive à bien déterminer d'ou viennent les données, plus la valeur de perte est basse. Le générateur, lui, se fait associer une perte à l'inverse de cela. Moins le discriminateur arrive à différencier les données, plus sa valeur de perte est petite, car cela veut dire qu'il arrive à générer des données qui se confondent avec les données originales.

Afin de pouvoir entraîner notre GAN, nous devons premièrement standardiser les vecteurs d'entrée. Pour ce faire, nous convertissons linéairement les données en valeurs entre 0 et 1 pour chaque colonne de nos données O-D originales.

Il faut par la suite définir l'architecture du modèle. En s'inspirant de cette page, nous essayons de reproduire des données O-D avec le GAN. Les architectures respectives du discriminateur et du générateur sont disponibles dans les Table 1 et Table 2.

Pour avoir une idée de la fonctionnalité du modèle, il est important de faire une comparaison entre les déplacements générés par le modèle et ceux provenant des données originales. Pour ce faire, nous comparons les distributions de chaque variable provenant des deux sources de données.

Après les premiers entraînements (300 époques), nous obtenons des distributions pour chacunes des variables ressemblant à celles de la Figure 1.

On observe que les distributions des variables générées ne sont pas représentatives des données sur lesquelles le modèle a été entraîné. Cependant, certaines distributions semblent se rapprocher, comme celle des modes de transport, ou le modèle à bien surreprésenter la catégorie 1, qui correspond au mode «auto conducteur». Les distributions des coordonnées peuvent aussi, à première vue, paraître de se rapprocher des distributions originales. Cependant, quand on regarde la distribution 2 dimensionnelle des origines/destinations, on observe le résultat dans la Figure 2.

Cette structure est évidemment non-représentative de la réalité. Pour se pencher plus sur ce problème, nous essayons de voir s'il est possible de paramétrer le réseau de neurones de sorte à ce que les coordonnées soient mieux représentées. Nous essayons donc d'entraîner le modèle sur les colonnes de coordonnées seulement. Pour les mêmes paramètres d'entraînement, on obtient le résultat de la Figure 3. On observe une nette amélioration du

Couche	Nombre de neurones	Nombre de paramètres
Linéaire	256	2,304
ReLU	256	0
Dropout	256	0
Linéaire	128	32,896
ReLU	128	0
Dropout	128	0
Linéaire	64	8,256
ReLU	64	0
Dropout	64	0
Linéaire	1	65
Sigmoïde	1	0

Table 1: Architecture du discriminateur

Couche	Nombre de neurones	Nombre de paramètres
Linéaire	16	144
ReLU	16	0
Linéaire	32	544
ReLU	32	0
Linéaire	8	264

Table 2: Architecture du générateur

modèle. Cela suggère que les structures présentes dans les données originales requièrent beaucoup de paramètres dans le modèle pour les décrire. En enlevant les autres colonnes, tout le modèle est dédié à la génération de coordonnées. Ainsi, il semble que le modèle ait plus de facilité à détecter et comprendre les structures. Cependant, on observe toujours des droites sur lesquelles le modèle génère anormalement trop de points. En faisant l'exercice inverse, c'est-à-dire, en gardant les colonnes autres que les colonnes de coordonnées, on obtient les résultats de la Figure 4. Encore une fois, on observe que la simplification des données mène à une meilleure distribution des résultats générés, sans toutefois que cela soit génial.

Ces problèmes sont sans-doute dus à un mauvais entraînement du modèle. Les GAN sont connus pour être difficiles à ajuster car ils requièrent une balance précise entre les performances du générateur et du discriminateur.

3 Utilisation de ChatGPT pour créer un modèle

Cette section documente la méthode derrière le fichier **Model_fromGPT.ipynb** qui est utilisé pour générer les données.

Afin d'explorer des alternatives, nous avons utilisé ChatGPT afin qu'il nous donne une base de code pour créer un réseau de neurones pouvant solutionner notre problème. L'idée pour guider ChatGPT à nous mener à un code fonctionnel était de commencer avec un problème plus simple que le notre, puis graduellement complexifier la chose afin de pouvoir générer des données O-D artificielles. La première tâche que nous lui ayons demandée fût de créer un réseau de neurones pouvant apprendre à générer des données lorsque le jeu de données original comprend deux colonnes: une contenant des 0 et des 1, puis l'autre contenant 3 valeurs possibles. Après quelques modifications, ChatGPT était capable d'effectuer cette tâche. Son approche n'utilise pas de GAN. Son code consiste à faire un réseau de neurones prédictif basé sur du bruit. L'idée est d'entraîner un modèle de classification en lui donnant les données elles mêmes en tant qu'identifiant. Par la suite, on demande au modèle de prédire les variables d'un déplacement particulier en lui donnant un vecteur de bruit.

Les résultats obtenus sont similaires à ceux obtenus par le GAN, certaines classes sont très surreprésentées, d'autres très sous-représentées, et la distribution des coordonnées semble être passable, mais la corrélation entre les latitudes et longitudes ne semble pas être apprise par le modèle. On peut voir les résultats générés par ce modèle dans la Figure 5.

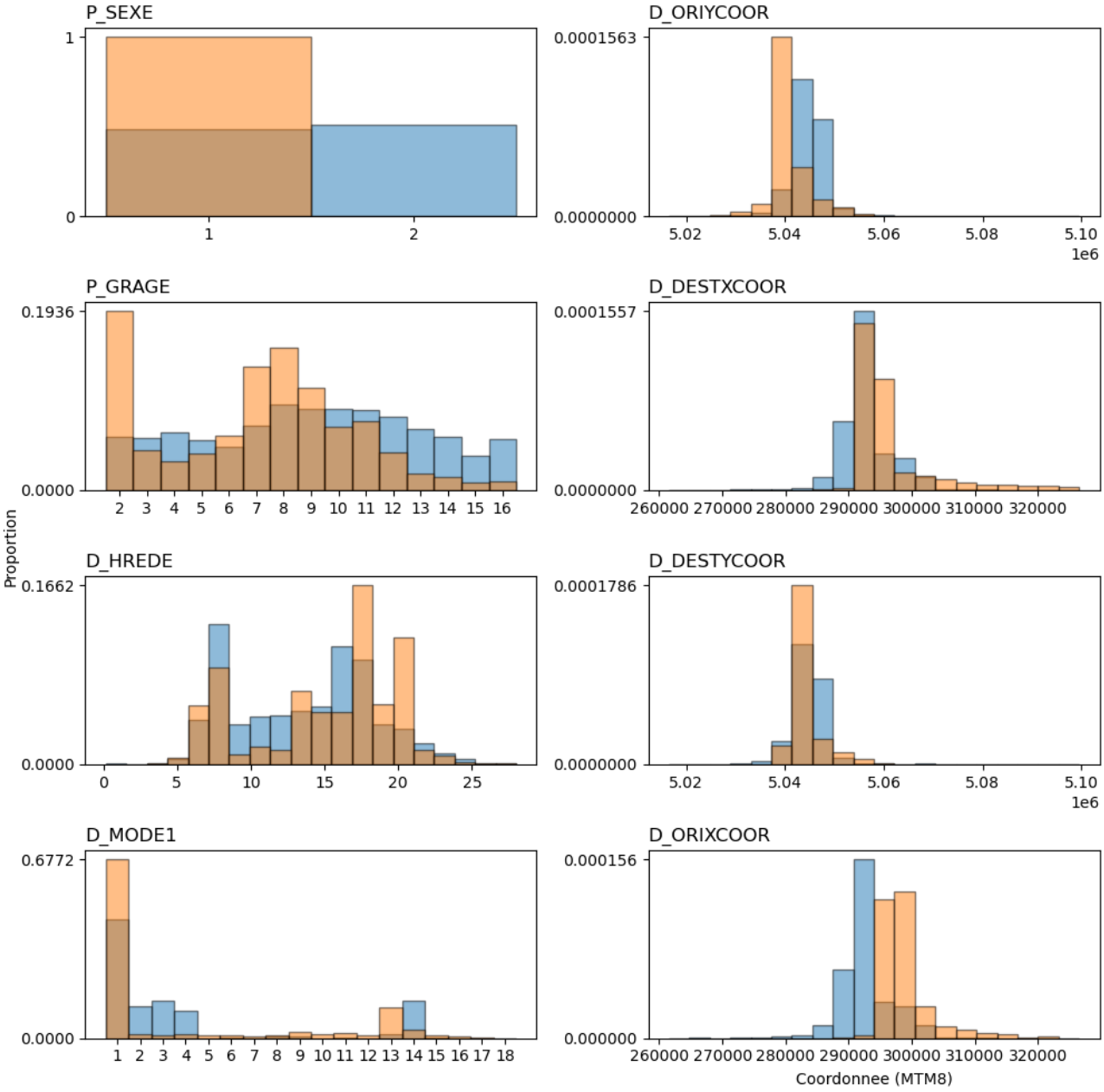

Figure 1: Variables d'intérêt générées par le GAN. Les distributions en bleu correspondent aux données OD originales et les distributions oranges correspondent aux données générées artificiellement

4 Augmentation de données tabulaires

Cette section documente la méthode derrière le fichier **DeepLearning_DataAugmentation.ipynb** qui est utilisé pour générer les données.

Une autre option de réseaux de neurones considérée fût celle de l'augmentation de données tabulaires. Cette méthode est généralement utilisée afin de surreprésenter des classes de données sous représentées dans des ensembles de données. Cependant, en incluant toutes nos données, nous espérons pouvoir recréer des données synthétiques représentant l'ensemble de nos données. En s'inspirant de cette page, nous synthétisons de nouvelles données OD. On peut observer les résultats comme avec les autres modèles, dans la Figure 6.

Pour ce type de modèle, nous observons le problème inverse aux deux autres modèles. Il semble incapable de

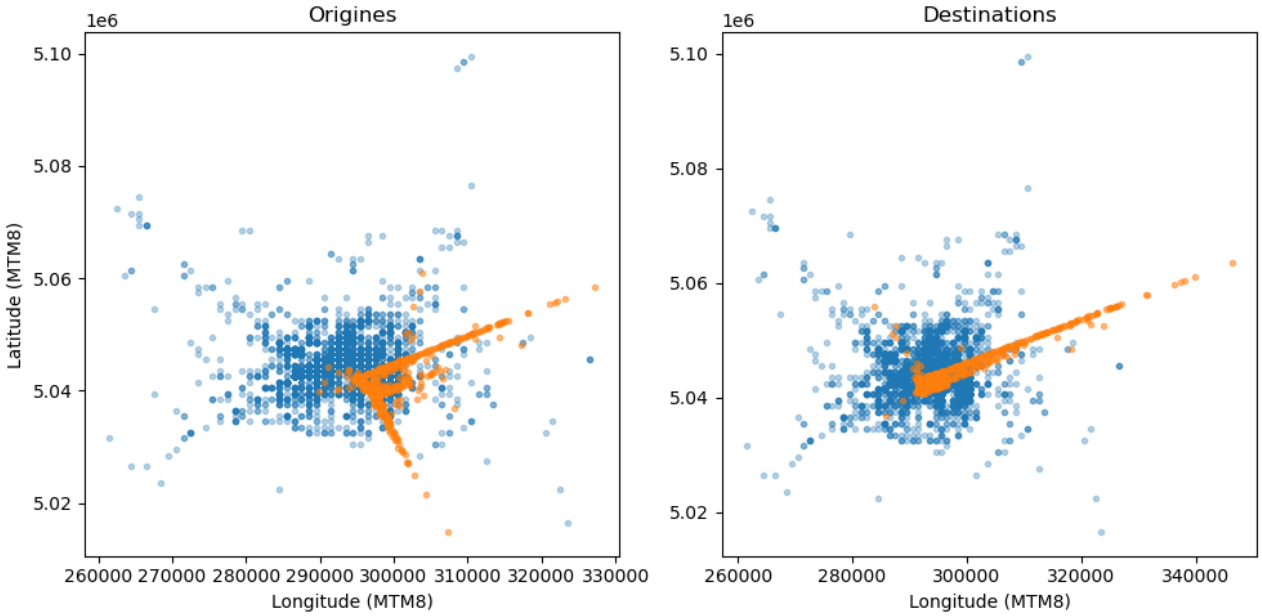

Figure 2: Distributions des origines et destinations générées par le GAN

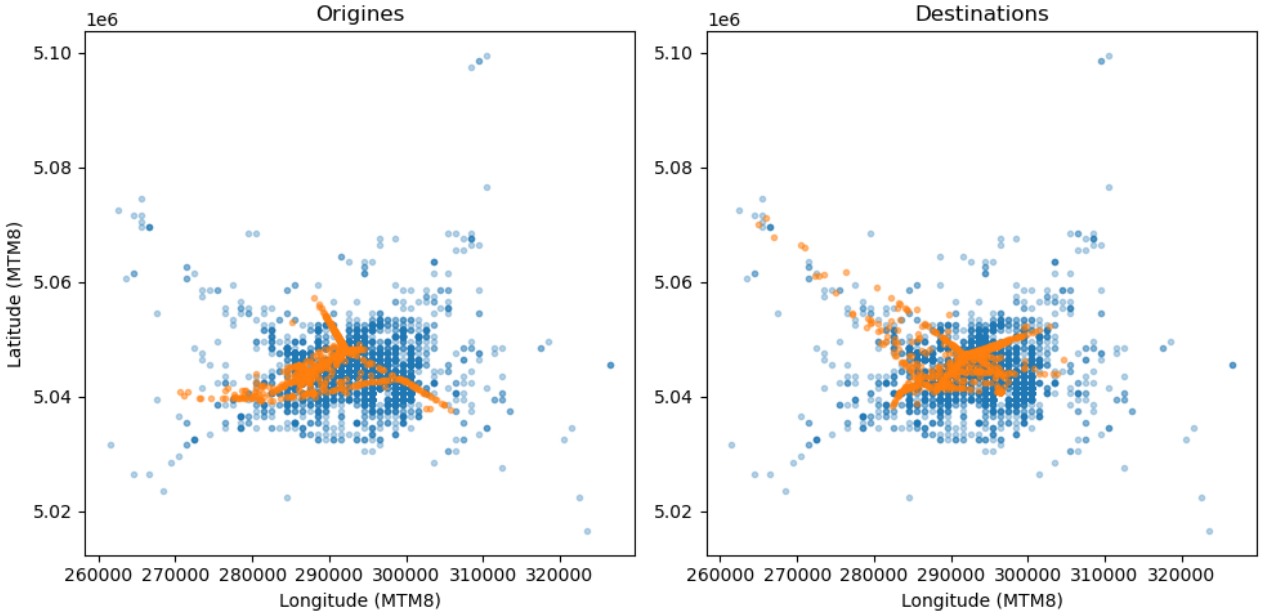

Figure 3: Origines et destinations générées par le GAN lorsqu'on l'entraîne uniquement sur les coordonnées

générer des distributions avec des classes très sur-représentées. Il semblerait que le réseau de neurone apprend seulement à générer des distributions centrées autour de la moyenne de chaque variable. Cela se remarque particulièrement dans la variable des modes de transport, ou les modes 5 à 13 sont très sur-représentés. On obtient cependant une très bonne distribution générée pour le sexe. On remarque par contre un phénomène étrange: toutes les distributions semblent respecter une loi quasi-normale centrée sur la moyenne de la plage de données possibles. Si on regarde le mode, notamment, qui est loin d'être normalement distribué, on observe que la sur-représentation du mode numéro 1 tire la distribution générée vers des petites valeurs. Dans les heures de départ, on observe que les heures de pointes sont inexistantes dans les données artificielles. Pour les données

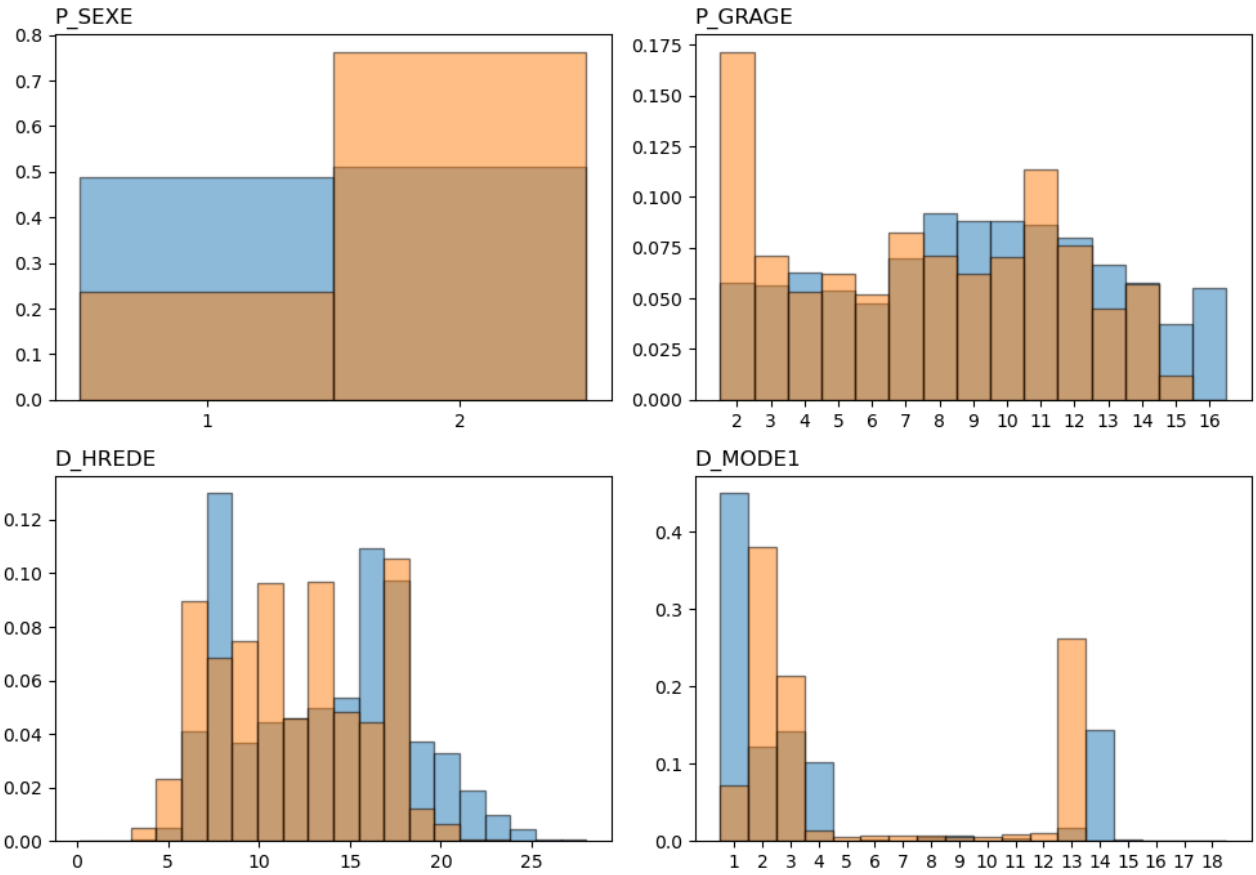

Figure 4: Distribution des variables autres que les coordonnées quand on entraîne le GAN sur celles-ci uniquement

des coordonnées, on voit également que les densités sur-représentées dans les données originales sont gravement sous-représentées dans les données artificielles.

5 Conclusions

5.1 Encodage des données

Pour toutes les données générées ci-haut, les variables sont converties en nombres entre 0 et 1. Cependant, les données OD contiennent un mélange de données catégorielles (ex.: mode de transport) et de données continues (ex.: coordonnées). Cependant, il est normalement plus approprié pour les variables catégorielles de faire de l'encodage unique pour chacune des variables. Si on prend l'exemple de la variable du sexe, comme les données sont composées de 2 valeurs possibles pour cette variable, les encodages uniques seraient [1,0] et [0,1]. Cependant, dans les réseaux de neurones, ces deux types de variables ont besoin d'un traitement différent. Plus précisément, il est nécessaire de calculer la perte différement pendant l'entraînement. Cependant, dans le cadre de ce projet, nous ne sommes pas arrivés à avoir une fonction de perte qui achevait ce but. Idéalement, il serait nécessaire de pouvoir calculer une perte à entropie croisée catégorielle (*categorical cross-entropy*) pour les variables catégorielles, et une perte d'entropie croisée binaire (*binary cross-entropy*) pour les variables continues. Cette modification d'encodage ne s'appliquerait cependant pas au modèle utilisant un GAN, car les pertes considérant ce dernier ne sont pas calculées par rapport aux données elles-mêmes, mais bien si le discriminateur arrive à différencier les vraies des fausses données.

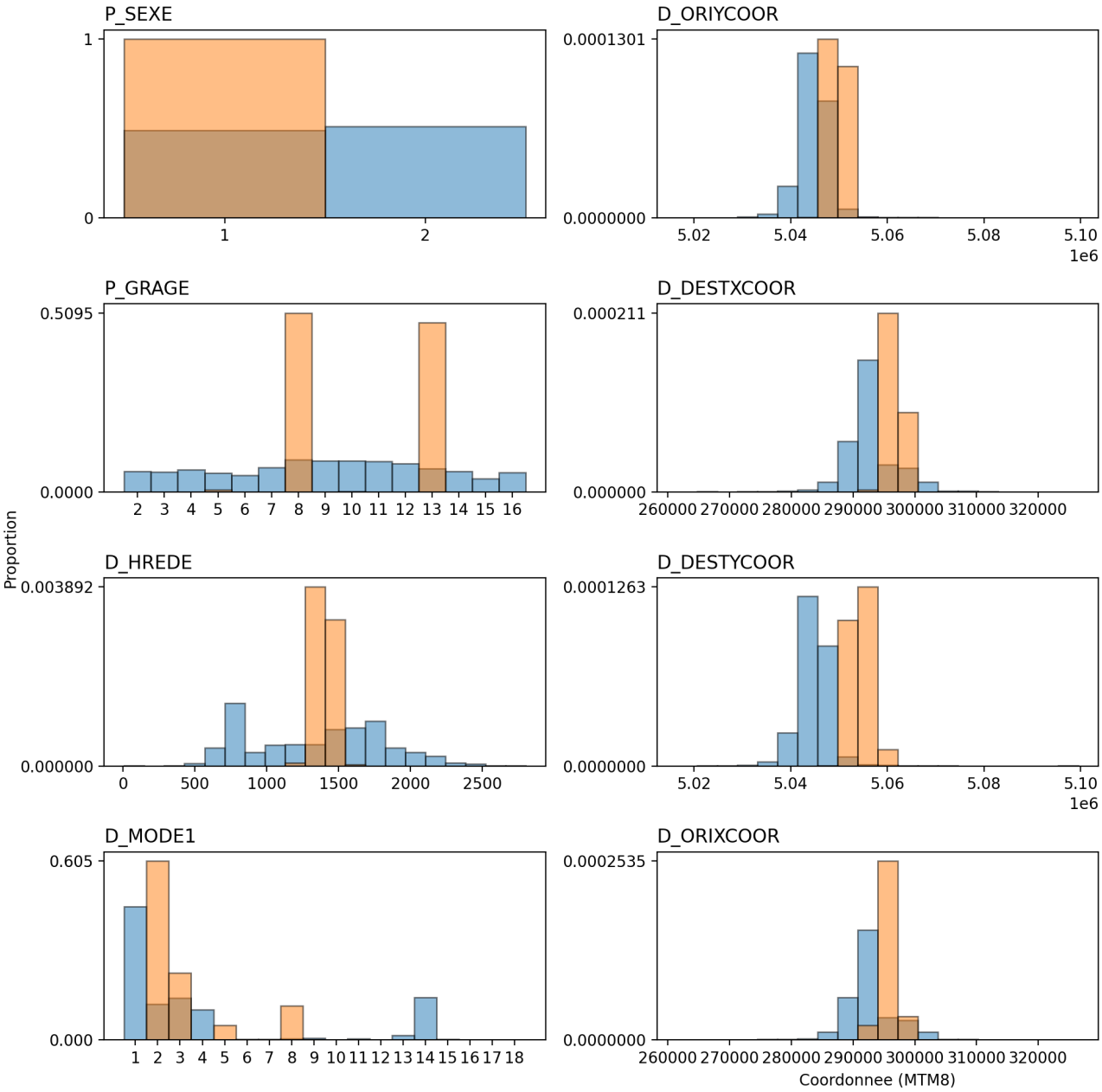

Figure 5: Variables d'intérêt générées par le modèle prédictif inspiré de chatGPT.

5.2 Modification des conditions d'entraînement

Les paramètres utilisés lors de l'entraînement sont critiques à la fonctionnalité du modèle. Hors, nos compétences en ce sujet et le temps disponible ne nous ont pas permis de trouver des combinaisons nous permettant de générer des déplacements représentant avec précision ceux de l'enquête OD. Une approche plus sytématique de sélection des paramètres, soit l'architecture du modèle, les rythmes d'apprentissages, temps d'entraînement et autres nécéssiteraient d'être mieux explorés.

Il pourrait également être bénéfique de voir si la complexité des structures dans les données originales pourrait être limitée en entraînant différents modèles sur différentes typologies de déplacement. Cela pourrait être, par exemple, en séparant les données originales par modes de transport, ou par distances de déplacement. Ainsi, les modèles pourraient avoir plus de facilité à cerner ce qui constitue les structures derrières ces données.

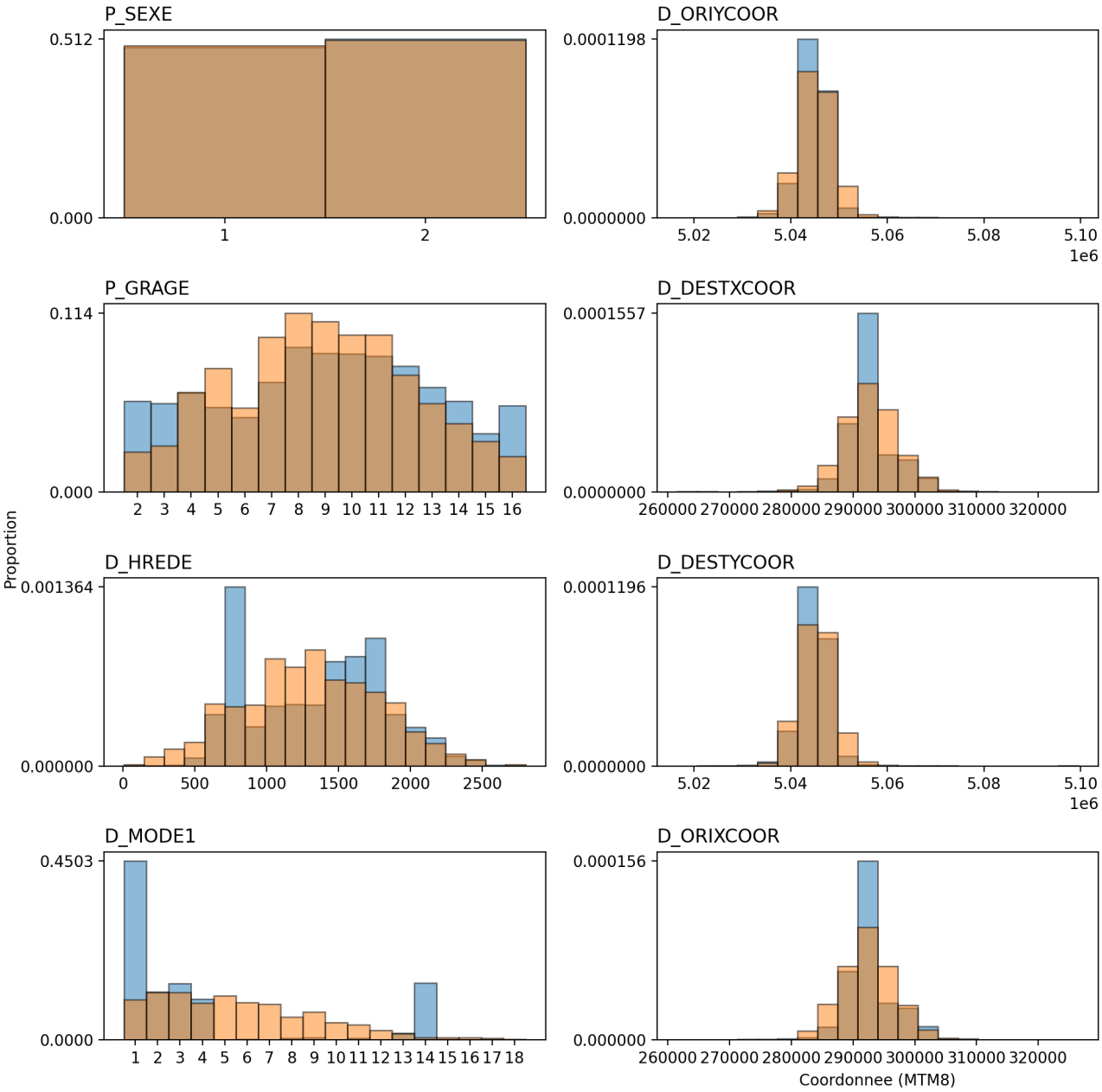

Figure 6: Variables d'intérêt générées par augmentation de données tabulaires

6 Notes finales

Après tous ces tests, nous pensons que l'utilisation d'un GAN est la méthode la plus prometteuse pour générer des données OD artificielles. Le modèle créé par ChatGPT semble mélanger des idées de modèles de classification et de modèles de génération, ce qui n'est pas souhaitable dans notre cas. Par ailleurs, ses résultats sont également très mauvais. Le modèle d'augmentation de données semble également avoir du potentiel, mais il repose sur des hypothèses inconnues quant à son fonctionnement. Ainsi, le GAN semble être la méthode la plus robuste pour accomplir notre tâche.

Notre recommandation face à des gens qui voudraient poursuivre ce projet serait de faire usage d'un GAN, mais de s'attarder sur comment construire celui-ci plus efficacement afin d'arriver à des résultats plus représentatifs des données originales.