ECOLE NATIONALE DE LA STATISTIQUE
ET DE L'ANALYSE DE L'INFORMATION

PROJET DE FIN D'ÉTUDES (PFE)

Troisième Année

RÉSEAUX DE NEURONES POUR MODÉLISER L'USURE DES MOTEURS D'AVION

Réalisé par :

Thomas CAMPOLUNGHI

David OUBDA

Caleb Carloss AGUIDA

Encadrant :

Responsable :

Arthur KATOSSKY

Encadrant :

Delphine BAY

Année académique : 2020-2021

Remerciements

Tout d'abord, nous tenons à remercier Mme BAY Delphine, qui nous a accompagné tout au long du projet et qui s'est montrée réactive pour répondre à nos questions. Également, merci à nos professeurs Mr GAUDEL Romaric et Mme MALLART Cyrielle qui ont pris du temps pour nous aider à étudier la problématique.

Table des matières

Glossaire

Marge EGT : Indicateur représentant l'usure du moteur. Plus un moteur est usé, plus il est chaud. A partir d'un certain seuil de température, le moteur n'est plus fonctionnel. La marge EGT correspond à l'écart de température du moteur avec ce seuil. La marge EGT diminue donc à l'usure du moteur.

Water wash : Lavage moteur comme opération de maintenance. La marge EGT augmente fortement après cette opération.

Liste des tableaux

Table des figures

⋆ ⋆ Chapitre Un ⋆ ⋆

Résumé

Le projet consiste en la modélisation de l'usure des moteurs d'avion au travers d'un réseau de neurones convolutif. La variable qui sert d'indicateur de cette usure est la marge EGT. C'est une mesure d'écart à un certain seuil trop élevé de température du moteur. Nous nous intéressons aux variations de celle-ci. Les réseaux de neurones convolutifs prenants des images en entrée, deux méthodes on été mises en oeuvre pour la construction de ces images. La première consiste à prendre directement les matrices $n\ vols \times p\ variables$. La deuxième, nommée DeepInsight[Sha+19], permet de donner à nos matrices d'entrées les caractéristiques propres aux images. Aucune des deux méthodes n'a mené à des résultats satisfaisants en validation, et ce malgré de nombreux essais de modifications de la structure du réseaux ou des hyperparamètres. La deuxième partie de la problématique consistait à analyser les facteurs explicatifs de l'usure des moteurs, par exemple à travers l'interprétation des features du modèle, mais elle supposait que celui-ci ait de bons résultats en généralisation. Cette partie n'a donc pas pu être traitée.

Abstract

The project consists in modeling the wear of aircraft engines through a convolutional neural network. The variable that serves as an indicator of this wear is the EGT margin. It is a measure of deviation from a certain threshold of engine temperature. We are interested in the variations of this one. Since convolutional neural networks take images as input, two methods have been implemented for the construction of these images. The first one consists in taking directly the matrices n flights × p variables. The second one, named DeepInsight [Sha+19], allows us to give to our input matrices the characteristics of images. Neither of the two methods led to satisfactory results in validation, despite numerous attempts to modulate the structure of the network or the hyper-parameters. The second part of the problem consisted in analyzing the explanatory factors of the wear of the motors, for example through the interpretation of the features of the model, but it supposed that this one has good results in generalization. This part could not be treated.

Chapitre Deux

Introduction

2.1 Problématique

À chaque vol, les moteurs d'avions fournissent des données qui informent sur les conditions du vol : environnement, type d'opération, routes, état du moteur, etc. Ainsi, nous disposons d'une base de données de plusieurs millions de vols avec de nombreux paramètres de vol. Les données nous permettent de suivre l'évolution d'un paramètre en particulier, appelé "marge EGT", représentant l'usure du moteur.

La figure 2.1 présente l'évolution de la marge EGT au cours du temps pour un moteur donné. Nous remarquons que celle-ci a tendance à diminuer avec le temps. Cela représente l'usure du moteur. Nous nous apercevons également que les opérations de maintenance ont un fort impact sur la marge EGT, celle-ci augmente fortement après chacune d'elles.

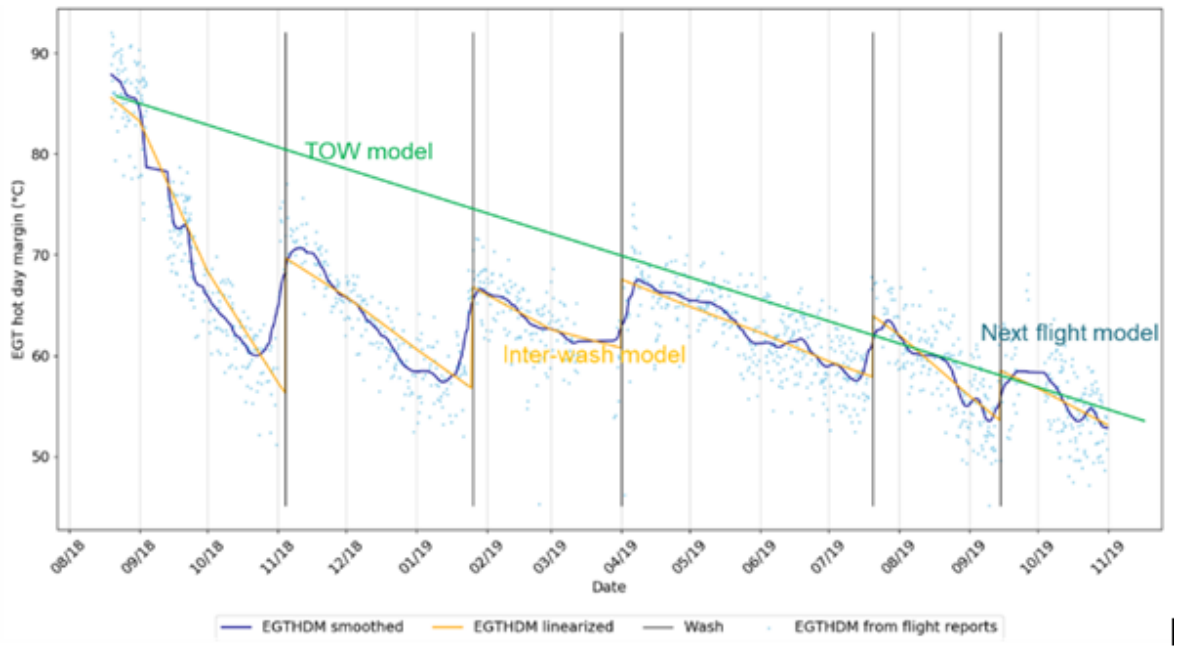


Figure 2.1: Évolution de la marge EGT au cours du temps

Le premier objectif du projet est de modéliser les pentes de la marge EGT entre deux événements de maintenance du moteur (water washes, shop visit etc.) en fonction des paramètres d'usage du moteur.

La question qu'on se pose est donc : « Entre deux évènements de maintenance, sachant les conditions dans lesquelles le moteur a volé (routes, environnement, type d'utilisation etc ...), quelle est la vitesse de dégradation de la marge EGT. »

Nous pourrons par la suite analyser les facteurs qui influent sur cette vitesse de dégradation.

Nous nous restreindrons à des réseaux de neurones pour cette modélisation. Des essais utilisant des réseaux récurrents ont déjà été réalisés et n'ont pas mené à des résultats satisfaisants. C'est pourquoi nous étudierons pour ce projet la piste des réseaux de neurones convolutifs.

2.2 Description des données

Nous disposons pour nos travaux d'une base de données de 2 913 834 lignes et 24 colonnes. Chaque ligne correspond à un vol et les colonnes sont des variables permettant de décrire chacun des vols. Parmi elles il y en a qui identifient le moteur qui a effectué le vol (numéro d'immatriculation, famille, type et configurations du moteur). D'autres donnent des informations sur le vol : la durée du vol, la date et la variable d'utilisation du moteur (*var_mot_1*). Enfin, il y'a des variables correspondant à l'historique du moteur : le nombre de vols effectués précédemment par le moteur et les variables de type rank (*WW_rank*, *SV_rank*, *Config_B_rank*). Ces dernières correspondent au nombre respectivement de water washes, de shop visits et de modifications de configuration subis par le moteur. La variable *Event_rank* résume ces informations puisqu'elle est incrémentée chaque fois qu'une opération de maintenance a lieu. C'est elle qui servira par la suite à définir nos groupes. Il y a également cinq variables environnementales servant à connaître par exemple les conditions météorologiques dans lesquelles le vol a été effectué. Enfin, nous disposons d'une variable sur la marge EGT du moteur au moment du vol ainsi que la pente de la marge EGT sur la période considérée. Une description détaillée des variables est fournie en annexe (Table 6.1).

2.2.1 Statistiques descriptives

Les tableaux 2.1 et 2.2 nous donnent des informations sur la distribution des variables de notre base de donnée. Nous disposons de dix variables quantitatives et de treize variables qualitatives. Nous constatons que certaines des variables sont réduites et d'autres non et que la variable engine_family ne dispose que d'une modalité.

Variable	Moyenne	Médiane	Ecart type
cycles	1.6e+03	1.3e+03	1.2e+03
cycles_counter	1.6e+03	1.3e+03	1.3e+03
egt_margin	5.5e-17	0.31	1
var_mot_1	1.7e-16	0.24	1
flight_leg_hours	2.1	1.9	1.3
var_env_1	1.5e-17	-0.35	1
var_env_2	-1.6e-18	-0.32	1
var_env_3	1.2e-16	0.24	1
var_env_5	2.5e-16	0.15	1
egt_slope	-0.029	-0.023	0.2

TABLE 2.1: Variables quantitatives

Variable	Nombre de modalités	Mode
engine_serial_number	1397	ESN_2
engine_family	1	Engine_family_1
engine_series	7	Engine_series_1
event_rank	26	0
SV_indicator	2	0
SV_rank	5	0
Config_B_indicator	2	0
Config_B_rank	12	0
WW_indicator	2	0
WW_rank	20	0
config_A	5	Config_A_1
config_B	4	Config_B_1
var_env_4	5	0

TABLE 2.2: Variables qualitatives

2.2.2 Valeurs manquantes

Onze variables de la base de données comportaient des données manquantes. La table ci-après donne un aperçu du nombre de valeur manquantes pour chacune de ces variables.

Variable	Nombre de valeurs manquantes
egt_margin	992 605 (34%)
var_mot_1	992 607 (34%)
flight_leg_hours	477 302 (16%)
SV_rank	992 605 (34%)
Config_B_rank	992 605 (34%)
WW_rank	992 605 (34%)
var_env_1	451 389 (15%)
var_env_2	258 150 (9%)
var_env_3	625 419 (21%)
var_env_4	622 175 (21%)
var_env_5	625 530 (21%)

TABLE 2.3: Nombre de valeurs manquantes par variable

Nous constatons que les variables egt_margin , var_mot_1 , SV_rank , Config_B_rank, WW_rank ont une forte proportion de valeurs manquantes.

2.2.3 Corrélations

Dans cette section, nous nous intéressons à la corrélation qui peut exister entre nos variables explicatives et la marge EGT.

Afin de mettre à l'écart l'effet des opérations de maintenance sur la marge EGT, nous devons nous intéresser aux corrélations internes aux groupes, à savoir, entre deux opérations de maintenance. La table 2.4 présente la moyenne et l'écart type des coefficients de corrélation linéaire de chaque groupe entre les variables explicatives quantitatives et la marge EGT.

Variable	egt_margin	
	Moyenne	Ecart type
var_mot_1	0.05	0.30
flight_leg_hours	0.01	0.24
var_env_1	0.026	0.23
var_env_2	0.024	0.29
var_env_3	0.18	0.32
var_env_5	0.17	0.26

TABLE 2.4: Coefficients de corrélation linéaire entre les variables explicatives quantitatives et la marge EGT. Moyenne et écart-type des groupes.

Les coefficients sont faibles. Les variables environnementales 3 et 5 sont les variables environnementales les plus corrélées avec la marge EGT avec des coefficients à 0.18 et 0.17.

Il est important de noter que nous n'avons analysé que les corrélations linéaires. Il peut exister des liens non-linéaires plus importants entre les variables.

⋆ ⋆ Chapitre Trois ⋆ ⋆

Description de la méthodologie

Il est important de comprendre que, pour un moteur et un event rank donnés, tous les vols ont la même valeur de pente. Celle-ci a été estimée par régression linéaire par morceaux de l'egt_margin. Un individu statistique est ici identifié par un event rank pour un moteur particulier. Il y a donc plusieurs vols pour chaque individu statistique. En tenant compte de cela, nous avons représenté nos individus par des matrices grâce à deux différentes méthodes.

3.1 Méthode 1

3.1.1 Description

Pour cette première méthode, nous avons dans un premier temps regroupé les vols, chaque groupe étant défini par un nom de moteur et un event rank. De chaque groupe nous échantillonnons par la suite autant que possible des blocs de n vols pour le représenter. Nous avons effectué pour cela des tirages aléatoires uniformes sans remise. Comme indiqué dans la section précédente, il y a énormément de valeurs manquantes pour un certain nombre de variables. Une première stratégie de traitement des valeurs manquantes a consisté à supprimer tous les vols ayant des valeurs manquantes. Cette stratégie nous a fait perdre une grande quantité de données. Nous avons dénombré 1 592 457 vols supprimés. Il est également à noter que le nombre de vols par groupe est assez variable. La taille moyenne des groupes était de 312, l'écart type de 234 et la médiane de 300. En fonction du nombre de vols que l'on décide de prendre, il arrive donc que certains groupes ne puissent pas être représentés. Nous avons choisi d'ignorer ce genre de groupes pour ne considérer que les groupes ayant au moins n vols. Par exemple, pour $n = 100$ vols, 3004 groupes de vols ont ainsi été ignorés, soit 155 071 vols supprimés. Le nombre total de vols supprimés est donc de 1 747 528 pour $n = 100$.

Au regard de l'importance du nombre de vols supprimés, nous avons mis en oeuvre une deuxième stratégie dans laquelle plutôt que de supprimer les valeurs manquantes, nous les avons imputées en fonction du groupe dans lequel elle se trouvaient. Pour les variables quantitatives, les valeurs manquantes ont été imputées par la moyenne de la variable concernée

au sein du groupe. Quant aux variables qualitatives, c'est le mode dans le groupe qui a été utilisé. Notons qu'il y a des groupes dans lesquels certaines variables contenaient beaucoup de données manquantes. Nous avons choisi de supprimer les vols situés dans des groupes pour lesquels il existe au moins une variable avec plus de 50% de données manquantes, ce qui nous a fait un total de 1 225 907 vols supprimés. Avec cette stratégie, nous avons finalement perdu 1 524 334 vols pour $n = 100$.

Les matrices construites sont de taille $n \times p$, p étant le nombre de variables. La procédure de construction des matrices est représentée par la figure ci-après. Pour $n = 100$ nous avons construits avec la première stratégie 9 626 matrices et 13 895 avec la seconde.

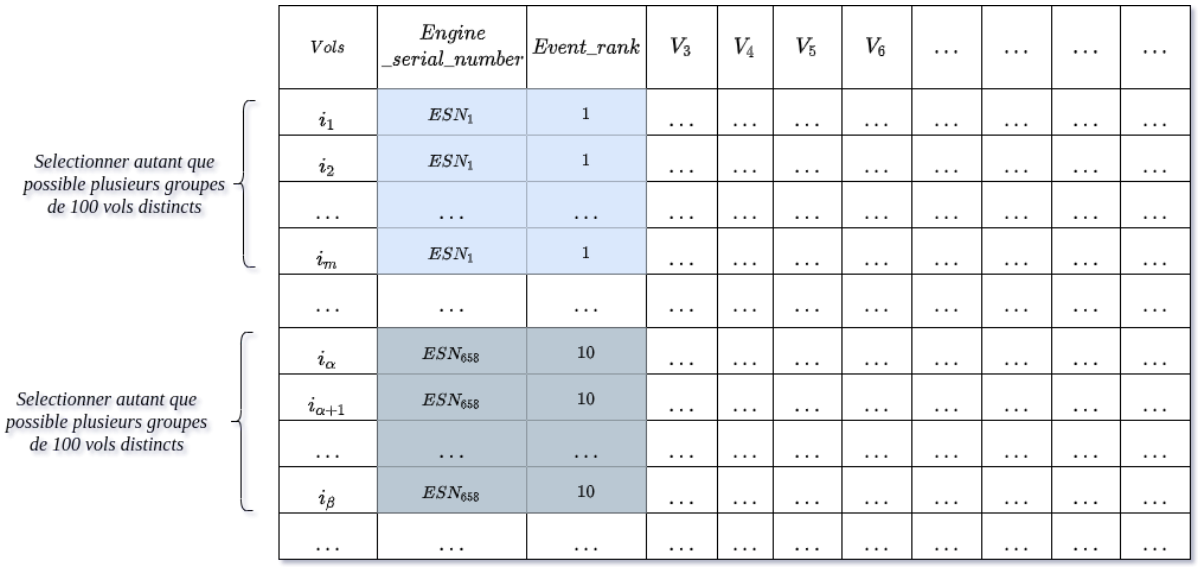


FIGURE 3.1: Illustration de la première méthode de construction des images avec une sélection de 100 vols.

3.1.2 Limites

L'algorithme que nous avons décidé d'utiliser dans nos travaux est un réseau de neurones convolutionnel (CNN). Les CNN sont connus pour être efficaces dans le traitement de données sous forme d'image. En choisissant de fournir en entrée les matrices précédemment construites à notre algorithme, nous les considérons comme des images. Cependant dans une image, des pixels proches les uns des autres partagent de l'information. Il y a donc une certaine dépendance entre les valeurs des pixels voisins [Sha+19]. Cela n'est a priori pas vrai dans notre situation. En effet les calculs de corrélations entre les variables que nous avons réalisés ont montré que très peu de variables sont vraiment corrélées, ce qui veut dire qu'il n'y a a priori aucune relation entre les pixels, même voisins sur les images construites par cette première méthode. De plus, même si les variables étaient dépendantes il aurait fallu choisir un ordre

fixe pour les variables dans les matrices construites. Nous avons représenté deux matrices construites à l'aide de la méthode 1 dans la figure ci-dessous.

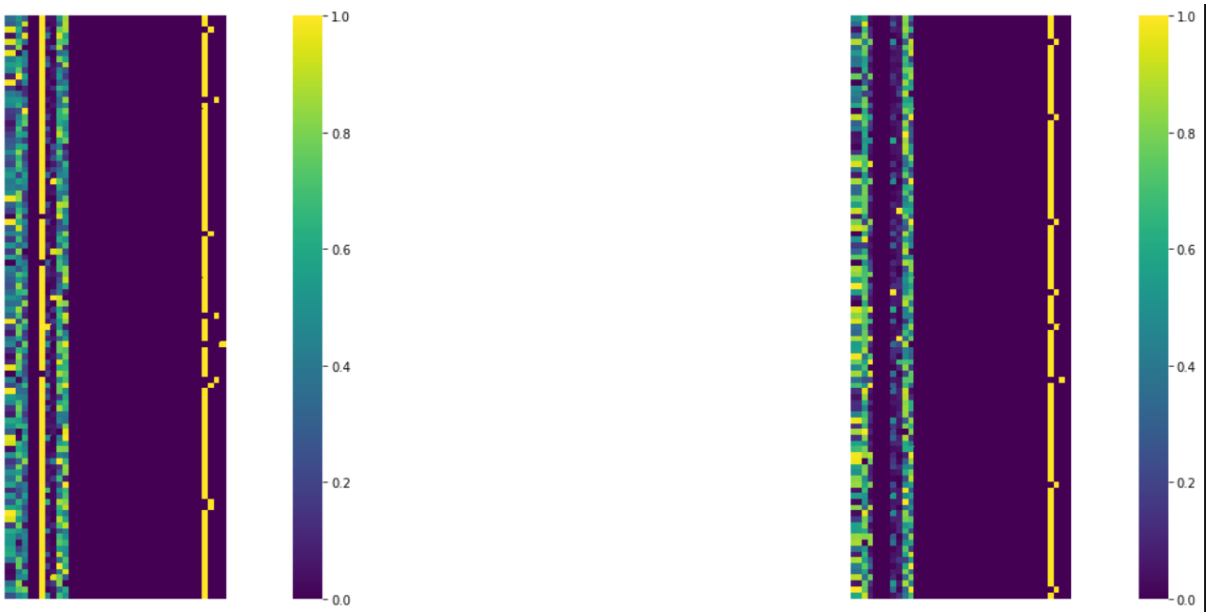


FIGURE 3.2: Images méthode 1

3.2 Méthode 2 : DeepInsight

3.2.1 Description

Dans la section précédente, nous avons évoqué les limites de la première méthode dans la construction des données d'entrée, notamment le fait que les matrices construites n'ont a priori pas des caractéristiques d'images. Nous explorons dans cette section une méthode permettant de transformer des données non-image en image pour l'architecture d'un réseau neuronal à convolution, la méthode DeepInsight présentée par Alok Sharma et al [Sha+19] en 2019.

Dans leurs travaux, Alok Sharma et al. ont construit une procédure de transformation de vecteurs de données (individus) en tenseurs ayant des caractéristiques d'images. La différence fondamentale entre l'application initiale de cette méthode et notre application se trouve au niveau des données. En effet, comme indiqué ci-dessus DeepInsight transforme chaque vecteur de données alors que nos unités de données sont ici des matrices. Nous avons donc transformé nos données matricielles en données vectorielles en appliquant une opération d'aplanissement (flattening) sur chacune des matrices. Ainsi, un individu de taille $n \times p$ est représenté par un vecteur de taille $1 \times (n \times p)$. Une fois que nous avons nos données sous ce format, nous appliquons la méthode DeepInsight de la même manière que celle présentée dans l'article.

La méthode consiste à :

☞ Projeter chacune des variables (colonnes de la base de données) dans un plan. Cela se fait par une ACP simple ou avec noyau ou encore par un t-SNE.

☞ Utiliser l'algorithme de l'enveloppe convexe (Convex Hull Algorithm) pour trouver le plus petit rectangle contenant toutes les variables représentées dans le plan puis effectuer une rotation de ce rectangle. A l'issue de cette étape, on récupère les coordonnées cartésiennes de chacune des variables.

☞ Convertir les coordonnées cartésiennes (x_c, y_c) des variables en coordonnées pixels (x_p, y_p) selon les formules suivantes :

$$x_p = round(1 + \frac{(x_c - x_{min}) \times A_p}{x_{max} - x_{min}}) \tag{3.1}$$

$$y_p = round(1 + \frac{(y_c - y_{min}) \times B_p}{y_{max} - y_{min}}) \tag{3.2}$$

$$o\text{ù :} \quad A_p = ceil(A_c \times \frac{precision}{d_{min}}), \quad B_p = ceil(B_c \times \frac{precision}{d_{min}}) \tag{3.3}$$

avec :

- precision = la résolution de l'image (nombre de pixels)
- d_{min} = la plus petite distance entre deux variables dans le plan.
- A_p resp B_p = les dimensions d'un pixel (largeur resp. hauteur). Notons, que les dimensions d'un pixel peuvent aussi fixées indépendamment des données. Nous avons utilisé le calcul automatique des dimensions.
- A_c resp B_c = les dimensions du rectangle obtenu après l'algorithme de de coque convexe (largeur resp. hauteur).
- $ceil(x)$ = le plus petit entier $\geq x$ et $round(x)$ = valeur entière arrondie de x

☞ Pour un individu donné, associer à chaque pixel la valeur de la variable positionnée sur ce pixel. Si plusieurs variables se trouvent sur le même pixel, la moyenne des valeurs prises par ces variables est affectée au pixel.

La figure ci-dessous illustre toutes les étapes de la méthode :

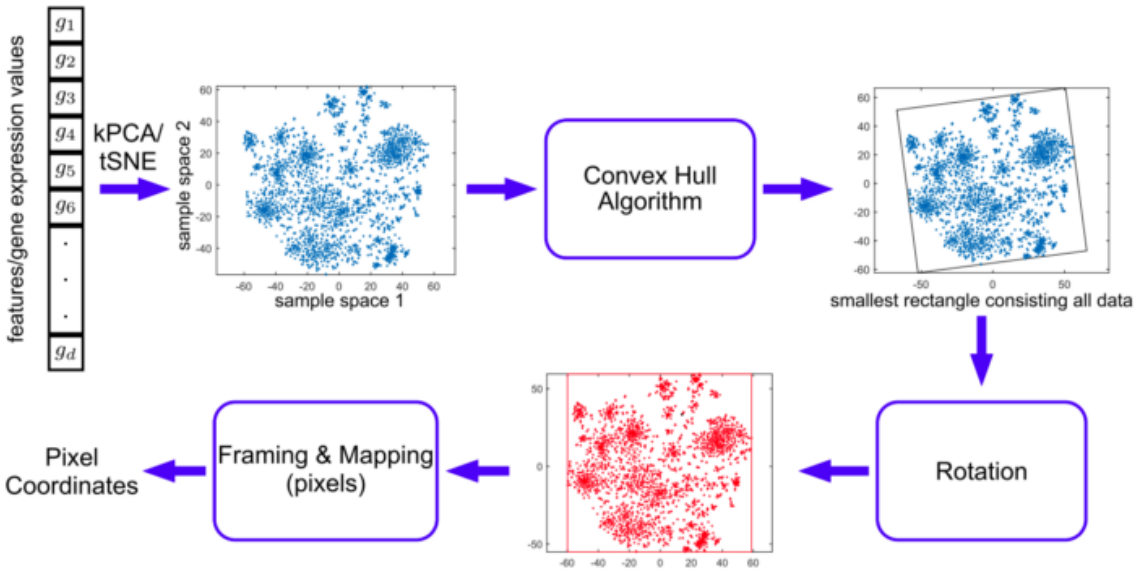


FIGURE 3.3: DeepInsight pipeline tiré de [Sha+19]

3.2.2 Limites

Même si elle permet d'avoir des matrices avec des caractéristiques plus proches de celles d'une vraie image que celles de la première méthode, la méthode du DeepInsight présente des limites dans notre application.

La première grande limite est causée par la vectorisation des matrices que nous avons effectuée avant l'application du DeepInsight. Cela a considérablement augmenté le nombre de variables à représenter dans le plan. En prenant par exemple $n = 100$ comme nombre de vols et $p = 35$ variables, on se retrouve avec 3500 variables à représenter dans le plan, ce qui fait que plusieurs variables se retrouvent à la même position après la conversion des coordonnées cartésiennes en coordonnées pixels. Une solution à ce problème serait de considérer des résolutions d'images plus grandes pour conserver les informations de la majorité des variables mais les capacités de stockage et de calculs dont nous disposions limitent la résolution que nous pouvons atteindre.

Une autre limite à souligner concerne la qualité de projection des variables. En effet, toute la procédure de transformation en image dépend de la représentation des variables sur le plan et une mauvaise représentation est synonyme de perte d'information. En utilisant par exemple l'ACP comme méthode de projection avec n = 100, nous avions autour de 73% de variance expliquée par le premier axe principal et 0.05% pour le second. Même si la quantité de variance totale expliquée est à peu près satisfaisante, on note que les variables sont très bien représentées sur l'axe 1 mais pas sur l'axe 2 alors que les coordonnées suivant chacun des axes

sont toutes importantes dans la procédure. Dans la figure ci-après sont représentées quelques images issues de la méthode DeepInsight.

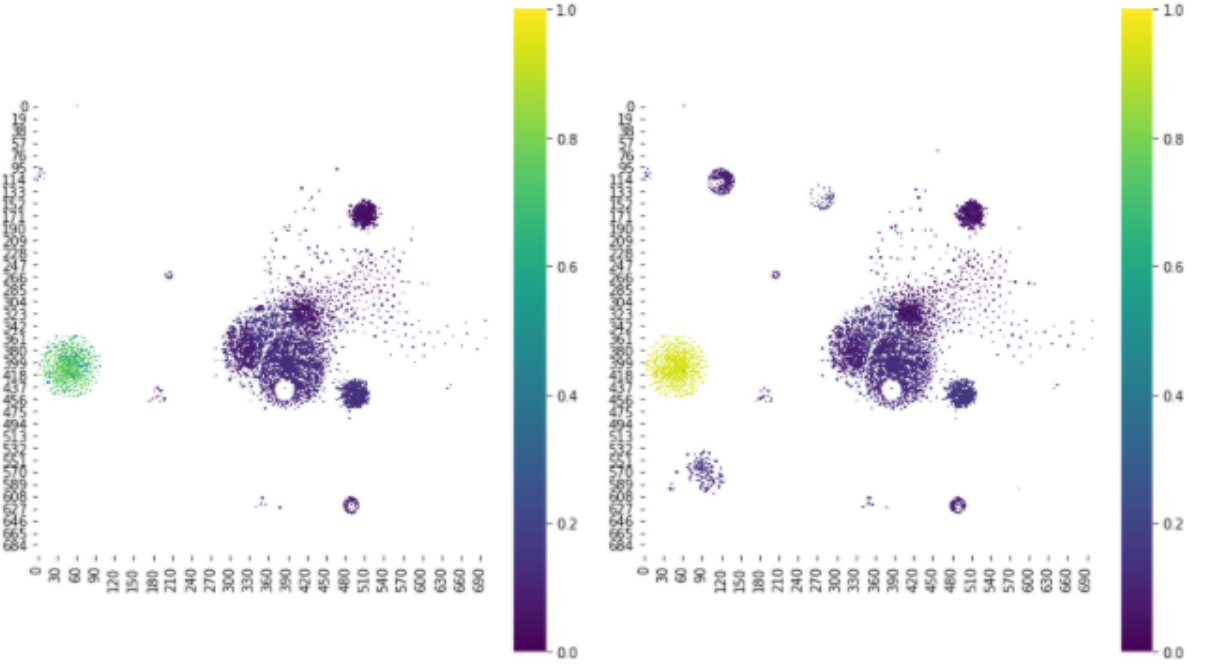


FIGURE 3.4: Images méthode 2

3.3 Le réseau de neurones

Une fois les matrices construites par l'une ou l'autre méthode, il ne nous reste plus qu'à trouver un algorithme d'apprentissage capable de prendre en entrée ces matrices. Les réseaux de neurones convolutifs peuvent être utilisés pour cela. Ils sont réputés efficaces pour la classification d'images, c'est-à-dire de tenseurs de données. Dans notre cas il s'agit plutôt d'un problème de régression. Ce type de problème est connu sous le nom de "vanilla deep regression" [IH20]. L'objectif du réseau à construire est ici de prédire la valeur de la pente de la marge egt. L'architecture du réseau reste un élément qu'il est important d'étudier. Outre l'architecture, les paramètres d'optimisation peuvent influencer les résultats lors d'un entraînement. Il s'agit notamment de la fonction de coût, et de la variante de l'algorithme de descente du gradient utilisée. Pour ce qui est de la fonction de coût, la nature de notre variable de sortie nous offrait le choix entre le *mse* (mean square error), le *mae* (mean absolute error) et le *mape* (mean absolute percentage error). Dans notre cas, nous avons préféré le MSE car il nous donnait de meilleurs résultats.

Nous avons essayé plusieurs structures de réseau en jouant sur le nombre de couches et le nombre de neurones par couche. L'architecture retenue est présentée en annexe 6.1 et la description dans le tableau 3.1. Le réseau est composé de trois couches de convolution, trois couches de MaxPooling et de deux couches denses. Au total, 36 928 caractéristiques ont été

extraites du modèle 1 et 1 982 528 du modèle 2. Aussi, pour palier aux problèmes de sur-apprentissage éventuels étant donné le grand nombre de paramètres à estimer, nous avons rajouté deux couches dropout. Compte tenu des ressources de calculs dont nous disposons et de la taille de la base de donnée, nous avons fourni des images ne possédant qu'une seule couche.

	Modèle 1			Modèle 2	
Layer (type)	Output Shape	Param		Output Shape	Param
Input Images	(100,38,1)			(200, 200,1)	
Conv2D	(None, 86, 24, 32)	7232		(None, 186, 186, 32)	7232
MaxPooling2	(None, 43, 12, 32)	0		(None, 93, 93, 32)	0
Dropout	(None, 43, 12, 32)	0		(None, 93, 93, 32)	0
Conv2D	(None, 40, 9, 64)	32832		(None, 90, 90, 64)	32832
MaxPooling2	(None, 20, 4, 64)	0		(None, 45, 45, 64)	0
Dropout	(None, 20, 4, 64)	0		(None, 45, 45, 64)	0
Conv2D	(None, 19, 3, 64)	16448		(None, 44, 44, 64)	16448
MaxPooling2	(None, 9, 1, 64)	0		(None, 22, 22, 64)	0
Flatten	(None, 576)	0		(None, 30976)	0
Dense	(None, 64)	36928		(None, 64)	1982528
Dense	(None, 1)	65		(None, 1)	65
Total params :	93505			2039105	
Trainable params :	93505			2039105	
Non-trainable params :	0			0	

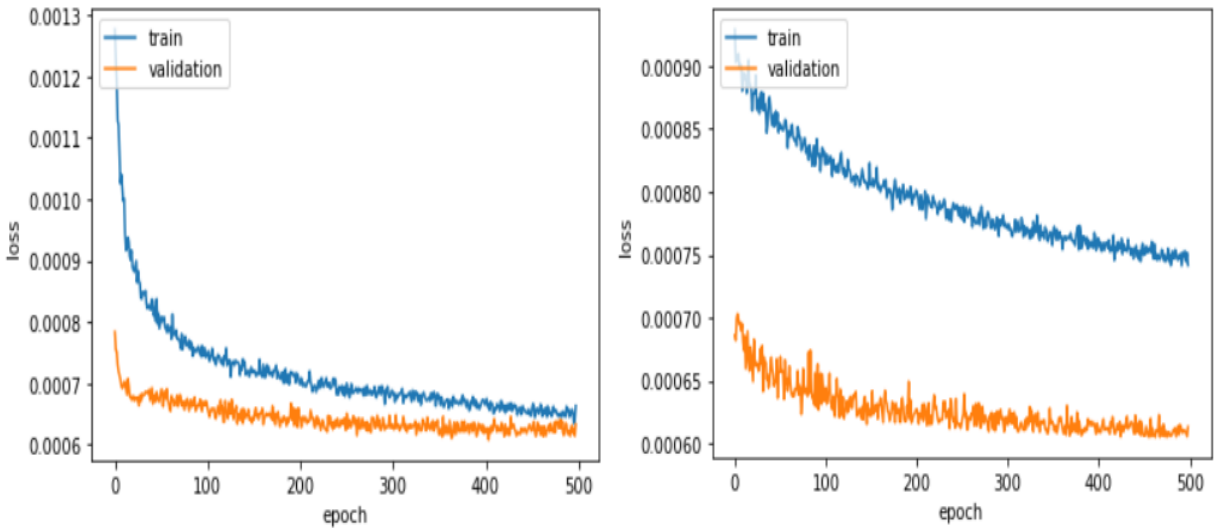
TABLE 3.1: Description de l'architecture du modele 1 et 2

Chapitre Quatre

Résultats

Nous présentons dans cette section les résultats obtenus avec chacune des deux méthodes.

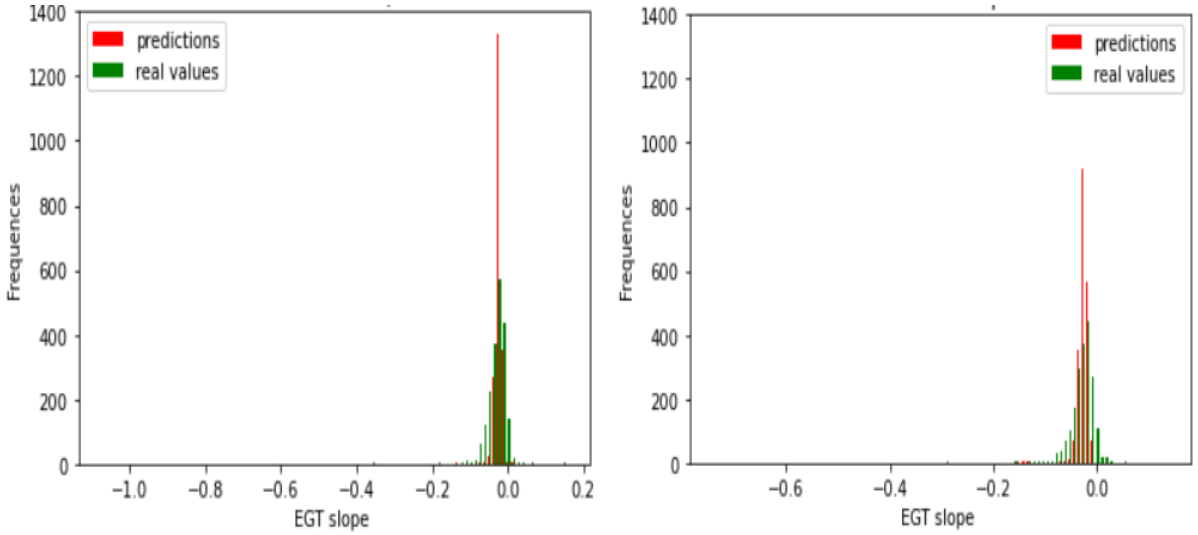
Nous avons entraîné le réseau retenu sur les données issues de chacune des deux méthodes sur 500 époques. Le réseau présente pour les deux méthodes des performances meilleures sur les données d'apprentissage que sur les données de test en matière de R^2. Cela est cohérent avec l'évolution observée des valeurs de la fonction de coût, présentée sur les figures 4.1a et 4.1b.



(a) Méthode 1 (b) Méthode 2

FIGURE 4.1: Évolution de la fonction de coût

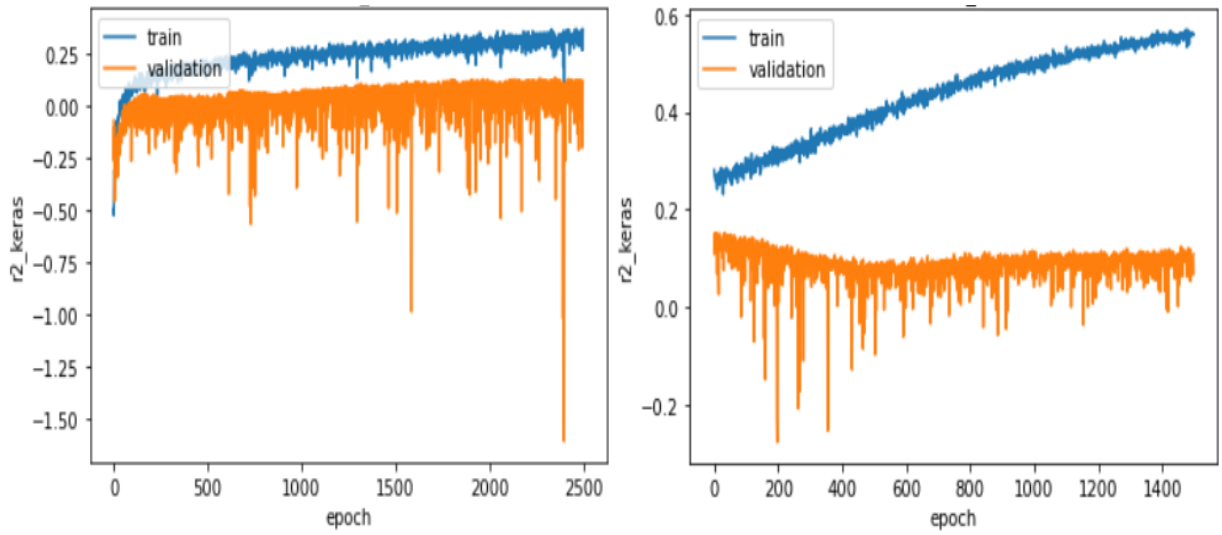
Nous avions pour la première méthode 0.05 et -0.03 en R^2 respectivement pour les données d'apprentissage et de test. La valeur négative du R^2 peut s'expliquer par le fait que le modèle est arbitrairement mauvais. Quant à la seconde méthode, les valeurs de R^2 étaient de 0.14 et 0.04. Malgré ces mauvais résultats obtenus en terme de R^2 nous avons néanmoins noté que la distribution des valeurs prédites n'était pas très différente de celle des valeurs réelles (cf figures 4.2a et 4.2b).



(a) Méthode 1 (b) Méthode 2

FIGURE 4.2: Prédictions VS valeurs réelles

Nous pouvons voir sur ces figures que l'algorithme semble continuer d'apprendre sur les données d'entraînement, ce qui nous a amené à considérer des nombres d'époques plus grands. Avec ces valeurs, le même constat est fait. Le modèle continue d'apprendre sur les données d'entraînement et pas sur les données de validation. L'évolution du R^2 est présentée sur les figures 4.3a et 4.3b.



(a) Méthode 1 (b) Méthode 2

FIGURE 4.3: Évolution du R^2 lors de l'entraînement.

Nous observons que la méthode 2 permet d'avoir des valeurs de R^2 supérieures en apprentissage en moins d'époques qu'avec la méthode 1. Il en est de même avec les données de validation. Cependant la présence de valeurs de R^2 négatives sur les données de validation même avec 2500 époques montre que le modèle a beaucoup de mal avec de nouvelles données.

⋆ ⋆ **Chapitre Cinq** ⋆ ⋆

Discussion

Malgré les variations que nous avons apportées aux hyper-paramètres, à la structure du réseau (nombre de couches et de neurones par couche), ou encore sur les matrices que nous fournissons en entrée, nous n'avons pas trouvé un modèle ayant progressé de manière satisfaisante sur les données de validation. Les valeurs du R^2 sur les données de validation sont souvent très faibles (≤ 0.1) ou parfois négatives. Cependant, celles sur les données d'apprentissages atteignent sur plusieurs époques un niveau satisfaisant pour la méthode 2 (≥ 0.6). D'une époque à une autre les pertes sont également très faibles, aussi bien en apprentissage qu'en validation. Au vu de ces observations, nous avons adopté plusieurs approches dont l'ajout/retrait de couches et noeuds, l'augmentation de données mais nous n'avons pas pu améliorer significativement le modèle.

Notre deuxième problématique, celle consistant à analyser les principaux facteurs influents sur la dégradation du moteur, ne peut pas non plus être réalisé en observant les poids et les features créés par le réseau, puisque ceux-ci ne permettent pas de prédire correctement notre variable cible.

Plusieurs pistes d'explication sont possibles pour les résultats obtenus.

- ☞ Premièrement, les données comportent énormément de valeurs manquantes. Ceci nous a contraint à supprimer quasiment la moitié des vols de notre base initiale lors de la constitution des images. De plus, les valeurs manquantes apparaissent sur des variables importantes comme les variables environnementales, la durée du vol ou la variable d'utilisation du moteur. Par ailleurs, la suppression des données manquantes, telle qu'elle a été faite dans un premier temps et lorsque l'imputation n'était pas possible, peut induire un biais sur les résultats trouvés.
- ☞ Aussi, la variable à prédire *egt_slope* est elle même une variable dont les valeurs ont été estimées précédemment par régression linéaire par morceaux. Elle peut donc également être une source de biais.
- ☞ De plus, cela peut provenir d'un défaut de méthodologie. En effet, il se peut que l'approche par un réseau de neurones convolutif ne soit pas adapté. Le sujet était assez expérimental, nous connaissions à l'avance la difficulté de retrouver les caractéristiques d'une image en gardant toute l'information de base.
- ☞ Enfin, les difficultés à obtenir une modélisation correcte peuvent provenir de la difficulté du problème initial. Il se peut que les liens de dépendance entre les variables explicatives

et la variable cible soient tout simplement très faibles et difficiles à détecter pour le réseau. Les corrélations linéaires, bien qu'elles n'analysent pas les relations non-linéaires sont déjà un premier indicateur que ce lien est faible.

⋆ ⋆ **Chapitre Six** ⋆ ⋆

Conclusion

L'objectif de ce travail consistait à modéliser la vitesse d'usure d'un moteur d'avion en fonction de son environnement de vol et d'autres facteurs. Cette vitesse d'usure du moteur est quantifiée par une variable nommée *egt slope*. Pour appréhender ce lien, nous avons testé un modèle de type réseau de neurone convolutif après un pré-traitement de notre base de donnée. Mais les résultats du modèle ne sont pas convaincants en vue de déterminer les variables explicatives les plus influentes.

Bibliographie

[Sha+19] Alok Sharma et al. *DeepInsight: Transforming Non-image data to Images for CNN Architectures*. https://www.nature.com/articles/s41598-019-47765-6.pdf. 2019.

[IH20] Stéphane Lathuilière Pablo Mesejo Xavier Alameda-Pineda Member IEEE et Radu Horaud. *A Comprehensive Analysis of Deep Regression*. https://arxiv.org/pdf/1803.08450.pdf. 24 Sep 2020.

[21] *Lien github DeepInsight*. https://github.com/alok-ai-lab/DeepInsight. consulté le 08 Mars 2021.

Annexes

Statistiques

Variable	Description
engine_serial_number	Numéro d'immatriculation
engine_family	Famille moteur
engine_series	Type moteur
date	Date du vol
cycles	Cycles estimés methode1
cycles_counter	Cycles estimés méthode2 (plus fiable)
marge_egt	Marge egt
var_mot_1	Variable d'utilisation du moteur
flight_leg_hours	Durée du vol
event_rank	Numéro d'évènement
SV_indicator	=1 si le moteur passe en Shop Visit (passage en atelier de réparation)
SV_rank	Nombre de SV effectuées
Config_B_indicator	=1 si la config B a été modifiée
Config_B_rank	Nombre de changements de la config B
WW_indicator	=1 si un waterWash est effectué
WW_rank	Nombre de WW déjà effectués
config_A	Type de config moteur
config_B	Type de config moteur
var_env_1	Variable environnementale[1]
var_env_2	Variable environnementale
var_env_3	Variable environnementale
var_env_4	Variable environnementale
var_env_5	Variable environnementale

Table 6.1: Description des variables

Variable	Nombre de modalités
engine_serial_number	1397
engine_family	1
engine_series	7
event_rank	26
SV_indicator	2
SV_rank	4
Config_B_indicator	2
Config_B_rank	11
WW_indicator	2
WW_rank	19
config_A	5
config_B	4
var_env_4	4

Table 6.2: Variables qualitatives

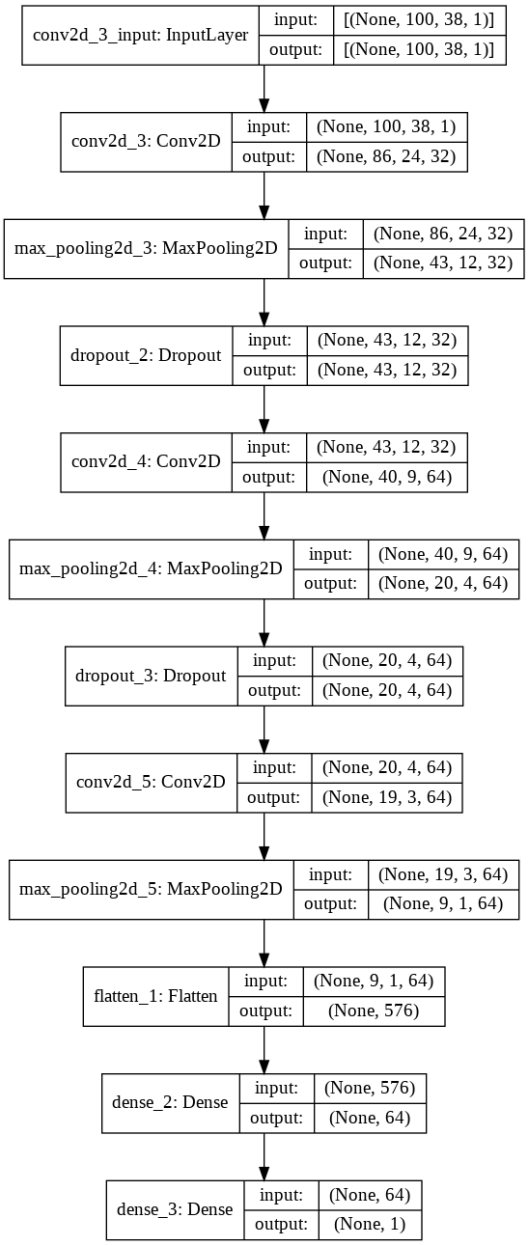


FIGURE 6.1: Architecture du modèle 1 CNN