ΔΗΜΟΚΡΙΤΟΣ
ΕΘΝΙΚΟ ΚΕΝΤΡΟ ΕΡΕΥΝΑΣ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
UNIVERSITY OF PIRAEUS

Δ.Π.Μ.Σ.
ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

Report εργασίας DL

Αλεξίου Κυριάκος ΑΜ: 2303

12 Ιουνίου, 2024

Η παρούσα εργασία σκοπό έχει να συγκρίνει τα αποτελέσματα ML τεχνικών missing data imputation με τις επιδόσεις μοντέλων DL όπως οι Autoencoders, οι Variational Autoencoders καθώς και τα Generative Adversarial Networks. Συνολικά εκτός από τα προαναφερθέντα μοντέλα εξετάστηκαν 5 διαφορετικές τεχνικές οι οποίες συγκρίθηκαν με τις επιδόσεις της naive mean missing data imputation method. Η μέθοδος αυτή υπολογίζει την μέση τιμή κάθε στήλης των δεδομένων και κάνει impute την τιμή αυτή στα κενά δεδομένα αυτής της στήλης και αποτέλεσε το baseline της παρούσας εργασίας. Οι εναπομένουσες μέθοδοι που χρησιμοποιήθηκαν είναι η μέθοδος kNN imputation καθώς και η πειραματική μέθοδος iterative imputer που διατίθενται από την βιβλιοθήκη scikit learn της Python. Το domain του πειράματος αφορά δεδομένα που προέρχονται από onboard μετρήσεις λειτουργικών παραμέτρων πλοίου τα οποία χρησιμοποιούνται για εκτίμηση της λειτουργικής του συμπεριφοράς.

Το πρόβλημα των χαμένων τιμών στο συγκεκριμένο domain είναι ιδιαίτερα αισθητό κυρίως λόγω των παρακάτω λόγων:

- Της πολυπλοκότητας των συστημάτων καταγραφής δεδομένων.
- Του ιδιαίτερα αφιλόξενου περιβάλλοντος στο οποίο λειτουργούν οι διάφοροι αισθητήρες που καταγράφουν λειτουργικές παραμέτρους μηχανημάτων.
- Της έλλειψης εξειδικευμένου προσωπικού που να είναι σε θέση να αποκαταστήσει προβλήματα της εγκατάστασης κατά την διάρκεια του ταξιδίου.
- Της "σύνθεσης" των δεδομένων από διαφορετικούς vendors (π.χ. weather data providers) που εντείνουν την προαναφερθείσας πολυπλοκότητα της εγκατάστασης.

Για να είναι αξιόπιστη η αξιολόγηση των λειτουργικών παραμέτρων απαιτείται η πληρότητα τους. Η εύρεση επομένως μιας αξιόπιστης διαδικασίας missing data imputation είναι σημαντική ώστε να μην «χάνονται» δεδομένα προς αξιολόγηση.

Η εφαρμογή δημιουργήθηκε σε περιβάλλον python. Όλα τα dependencies καθώς και βοηθητικά προγράμματα που θα αναλυθούν παρακάτω, βρίσκονται αναρτημένα στο παρακάτω repo:

https://github.com/kiriakos2004/dl_assign_Democritos.git

Περιγραφή πλοίου / δεδομένα εκπαίδευσης

Τα δεδομένα του πλοίου που χρησιμοποιήθηκαν για την εκπαίδευση του μοντέλου παραχωρήθηκαν από την εταιρεία Laskaridis Shipping Co. Ltd. και αφορούν ένα Dry bulk Carrier εκτοπίσματος 210000DWT. Τα βασικά χαρακτηριστικά του πλοίου φαίνονται στον πίνακα1.

ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΠΛΟΙΟΥ	
Εκτόπισμα	210000 DWT
Συνολικό μήκος	299
Μέγιστο πλάτος	50

Πιν.1 Βασικά χαρακτηριστικά πλοίου

Τα δεδομένα εκπαίδευσης αφορούν την περίοδο από 07/06/2022 έως 27/1/2022. Λήφθηκαν με συχνότητα δειγματοληψίας 0,016Hz (ένα instance ανά λεπτό). Στο χρονικό αυτό διάστημα συλλέχθηκαν συνολικά 336952 δείγματα από 77 διαφορετικές παραμέτρους

(attributes). Τα ανωτέρω δεδομένα συλλέγονται από το αυτόματο σύστημα επιτήρησης της εγκατάστασης πρόωσης του πλοίου, από ναυτιλιακά βοηθήματα καθώς και από weather providers με τους οποίους η πλοιοκτήτρια εταιρεία έχει σύμβαση. Στον πίνακα 2 φαίνονται τα διαφορετικά attributes του dataset.

DATETIME	MAIN ENGINE SCAVENG AIR RECEIVER TEMP	ME CYL 1 EXH GAS OUTLET TEMP
LATITUDE	AIR PRESSURE AT SEA LEVEL	ME CYL 2 EXH GAS OUTLET TEMP
LONGITUDE	WINDWAVE WAVE HEIGHT	ME CYL 3 EXH GAS OUTLET TEMP
COURSE OVER GROUND	WINDWAVE WAVE DIRECTION	ME CYL 4 EXH GAS OUTLET TEMP
MAGNETIC VARIATION	WINDWAVE WAVE LENGTH	ME CYL 5 EXH GAS OUTLET TEMP
MAGNETIC COURSE OVER GROUND	AIR TEMPERATURE AT 10M	ME CYL 6 EXH GAS OUTLET TEMP
WATER DEPTH	SEA TEMPERATURE	FUEL OIL DENSITY
RATE OF TURN	SIGNIFICANT WAVE HEIGHT	AFT DRAFT
PROPELLER SHAFT TORQUE	WAVE HEIGHT	FORE DRAFT
STERN TUBE BEARING TEMPERATURE	WAVE DIRECTION	CURRENT SPEED
INTERMEDIATE SHAFT BEARING TEMP	WAVE LENGTH	MEAN DRAFT
ME RPM ME AXIAL VIBRATION	SWELL SIGNIFICANT WAVE HEIGHT	WIND TRUE ANGLE
MAIN ENGINE LUB OIL INLET TEMP	SWELL WAVE HEIGHT	WIND TRUE SPEED
THRUST MAIN BEARING TEMP	SWELL WAVE DIRECTION	TRIM
ME CYL 1 PCO OUTLET TEMP	SWELL WAVE LENGTH	ME FUEL CONSUMPTION
ME CYL 2 PCO OUTLET TEMP	CURRENT DIRECTION	FUEL LOAD %
ME CYL 3 PCO OUTLET TEMP	SPEED OVER GROUND	EXH GAS AVER CYL 1
ME CYL 4 PCO OUTLET TEMP	SPEED THROUGH WATER	ME CYL 1 JCW OUTLET TEMP
ME CYL 5 PCO OUTLET TEMP	TURBOCHARGER SPEED	ME CYL 2 JCW OUTLET TEMP
ME CYL 6 PCO OUTLET TEMP	MAIN ENGINE SCAVENGE AIR PRESSURE	ME CYL 3 JCW OUTLET TEMP
FUEL OIL INLET PRESSURE	SHAFT POWER	ME CYL 4 JCW OUTLET TEMP
ME CYL 1 SCAV AIR TEMP	TURBOCHARGER EXH EXH GAS INLET TEMP	ME CYL 5 JCW OUTLET TEMP
ME CYL 2 SCAV AIR TEMP	TURBOCHARGER EXH EXH GAS OUTLET TEMP	ME CYL 6 JCW OUTLET TEMP
ME CYL 3 SCAV AIR TEMP	WIND SPEED	
ME CYL 4 SCAV AIR TEMP	WIND ANGLE	
ME CYL 5 SCAV AIR TEMP	HEADING	
ME CYL 6 SCAV AIR TEMP	PROPELLER SHAFT REVOLUTIONS	

Πιν.2 dataset attributes

Προ επεξεργασία δεδομένων εκπαίδευσης

Στο πλαίσιο του περιορισμού του υπολογιστικού κόστους εκπαίδευσης και με δεδομένο πως κάποια από τα προαναφερθέντα attributes μπορούν να προσεγγιστούν με ιδιαίτερη ακρίβεια από άλλα, παραλήφθηκαν. Αρχικά και με βάση το domain expertise, απαλείφθηκαν τα attributes "DATETIME, LATITUDE, LONGITUDE, WATER DEPTH, PROPELLER SHAFT TORQUE, AIR PRESSURE AT SEA LEVEL, AIR TEMPERATURE AT 10M, SEA TEMPERATURE, SHAFT POWER, PROPELLER SHAFT REVOLUTIONS, ME FUEL CONSUMPTION, FUEL LOAD %".

Λόγω της χαμηλής σχετικά ταχύτητας κίνησης των εμπορικών πλοίων αυτού του τύπου δεν παρουσιάζονται γρήγορες και μεγάλες μεταβολές μεγεθών. Σύμφωνα και με τις επιταγές του ISO-15016 του 2015, προς αποφυγή spikes που μπορεί να προκληθούν από ελαττωματικούς αισθητήρες εκτελέστηκε ομαλοποίηση των δεδομένων με χρήση moving average και χρονικό "παράθυρο" 10 λεπτών.

Τα δεδομένα για την εκπαίδευση, το tuning καθώς και του ελέγχου επιδόσεων των DL μοντέλων χωρίστηκαν σε train, validation και test datasets σε ποσοστά 70%, 10% και 20%. Κατόπιν κανονικοποιήθηκαν με χρήση της συνάρτησης StandardScaler() της βιβλιοθήκης scikit-learn της python (το fit εκτελέστηκε μόνο στα training δεδομένα προς αποφυγή πληροφορίας στα validation και test dataset).

Το αρχικό dataset περιείχε ελλείπουσες τιμές (γεγονός που πιστοποιεί την validity του πειράματος). Τα instances που είχαν ελλείπουσες τιμές απορρίφθηκαν με σκοπό να δημιουργηθεί ένα «ομοιογενές» dataset που θα διευκόλυνε τον έλεγχο των επιδόσεων.

Καθόσον ο μηχανισμός που «προκαλεί» την μη καταγραφή κάποιον τιμών στο dataset προκαλείται από αμιγώς τεχνικά προβλήματα και δεν επηρεάζεται από τα δεδομένα καθαυτά, μπορεί να θεωρηθεί ως missing completely at random (MCAR). Μια συνάρτηση που «αφαιρεί» δεδομένα τυχαία από το dataset χρησιμοποιήθηκε για να προσομοιώσει αυτό τον μηχανισμό. Ορίστηκαν τέσσερα διαφορετικά ποσοστά missing τιμών (10%, 20%, 30% και 50%) στα οποία δοκιμάστηκαν τα μοντέλα.

Μοντέλα / διαδικασία / τεστ

Αρχικά και με σκοπό να δημιουργηθεί μία βάση σύγκρισης των αποτελεσμάτων, χρησιμοποιήθηκε η συνάρτηση mean() με την χρήση της οποίας τοποθετήθηκε σε όλες τις θέσεις ελλειπόντων τιμών κάθε στήλης, η μέση τιμή της στήλης αυτής.

Ως δεύτερη μέθοδος χρησιμοποιήθηκε η KNNImputer() που διατίθεται από την βιβλιοθήκη scikit-learn. Μετά από αρκετούς πειραματισμούς επιλέχθηκε η τιμή πέντε (5) όσο αναφορά την υπερπαράμετρο των nearest neighbors καθώς επέτυχε και τα καλύτερα αποτελέσματα.

Η τρίτη και τελευταία μέθοδος που χρησιμοποιήθηκε (στο ML μέρος του πειράματος) ήταν η (πειραματική) συνάρτηση IterativeImputer () που διατίθεται από την βιβλιοθήκη scikit-learn. Σε αυτή την περίπτωση χρησιμοποιήθηκε ως estimator η LinearRegression με υπόλοιπες υπερπαραμέτρους ως ακολούθως:

- tol=1e-7
- max_iter=20
- initial_strategy='mean'
- imputation_order='descending'

Κατά το "τμήμα" DL του πειράματος χρησιμοποιήθηκαν Autoencoders, Variational Autoencoders καθώς και Generative Adversarial Networks μοντέλα. Σε κάθε περίπτωση τα μοντέλα εκπαιδεύτηκαν στο 70% των δεδομένων, έγινε tuning / validation στο επόμενο 10% και τέλος δοκιμάστηκαν στο εναπομείναν 20% του dataset. Σε αυτό το σημείο είναι ιδιαίτερα σημαντικό να τονιστεί πως τα δεδομένα του dataset αποτελούν ουσιαστικά την σύμπτυξη διαφορετικών ταξιδιών του ιδίου πλοίου με **διαφορετικές καιρικές συνθήκες** και **διαφορετικές καταστάσεις φορτώσεως**, γεγονός που κατά τον γράφοντα ενισχύει το validity των αποτελεσμάτων.

Όλα τα μοντέλα έκαναν χρήση LSTM σε διάφορες διαμορφώσεις (με ένα ή δύο επίπεδα) και σε διαφορετικές τιμές των παρακάτω υπερπαραμέτρων:

- Embedding Dim
- Hidden Dim
- Batch size
- Learning Rate
- Seq Length

Οι διαμορφώσεις των μοντέλων ήταν ως εξής:

1. Autoencoders

   Αρχιτεκτονική:

   Δυο βασικές αρχιτεκτονικές χρησιμοποιήθηκαν με χρήση είτε δύο LSTM layers στον ENCODER καθώς και δύο LSTM layers στον DECODER, είτε ενός LSTM layer στον ENCODER και ενός LSTM layer στον DECODER. Μετά από κάθε LSTM layer εφαρμόστηκε ένα dropout layer του οποίου η τιμή p (0 – 1) αποτέλεσε υπερπαράμετρο. Στην περίπτωση της χρήσης δύο LSTM layers η διάσταση από το ένα layer στο άλλο τόσο στον ENCODER όσο και στον DECODER ορίστηκε ως η διπλάσια του latent space. Ως activation

function χρησιμοποιήθηκε σε όλα τα layers η tanh (default σε pytorch για LSTM).

Training phase:

Το dataset περιέχουν ελλείπουσες τιμές και για να χρησιμοποιηθούν από τον Autoencoder αρχικά έγινε impute στις θέσεις που υπήρχαν κενά τιμές με χρήση της συνάρτησης mean(), (η μέση τιμή της στήλης του training dataset τοποθετήθηκε στις θέσεις που έλειπαν δεδομένα. Οι "θέσεις" των δεδομένων που δεν υπήρχαν σημειώθηκαν με χρήση κατάλληλης συνάρτησης η οποία δημιουργεί ανάλογων διαστάσεων με το αρχικό dataset και στις θέσεις που υπήρχαν δεδομένα θέτει την τιμή 1 ενώ στις θέσεις ελλειπόντων τιμών θέτει τιμή 0 (Mask). Εν συνεχεία τα δεδομένα εκπαίδευσης κανονικοποιήθηκαν με χρήση συναρτήσεως MinMaxScaler() της βιβλιοθήκης scikit-learn της Python.

Το training dataset χωρίστηκε σε sequences οι οποίες είχαν μέγεθος 66 (number of attributes) X 10 (sequence length) και με αυτά τα sequences εφαρμόστηκαν στην είσοδο του autoencoder.(Χρησιμοποιήθηκαν και άλλες τιμές sequence length αλλά η τιμή 10 έδωσε τα καλύτερα αποτελέσματα στους autoencoders). Ως Loss χρησιμοποιήθηκε το Mean Square Error (MSE) και ως optimizer ο Adam.

Κατά τη φάση της εκπαίδευσης, η μάσκα (Mask) χρησιμοποιήθηκε για να λαμβάνονται υπόψη επιλεκτικά μόνο οι παρατηρούμενες (μη ελλείπουσες) τιμές στον υπολογισμό του Loss.

Σε κάθε εποχή υπολογιζόταν η τιμή του validation Loss και η καλύτερη εκ των τιμών διατηρούταν στην μνήμη, αυτό έδωσε την δυνατότητα να χρησιμοποιηθεί patience ως κριτήριο εκπαίδευσης.

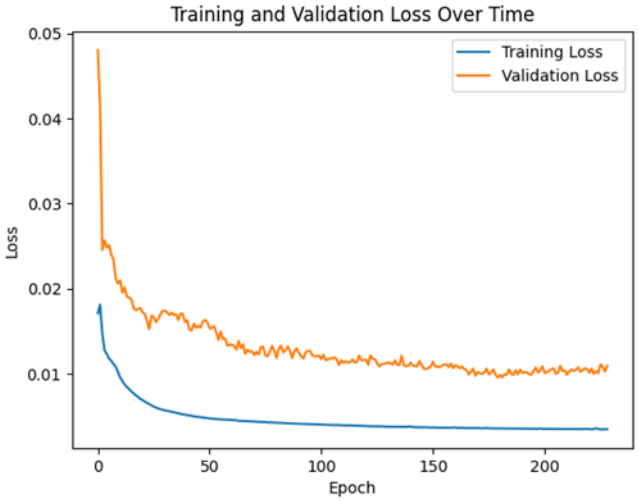
Μόλις επετεύχθη overfitting και πιστοποιήθηκε πως το expressivity του μοντέλου ήταν επαρκές για το πρόβλημα, εφαρμόστηκαν τεχνικές regularization ώστε να αντιμετωπιστεί το φαινόμενο και να συνεχιστεί η εκπαίδευση. Για το σκοπό αυτό χρησιμοποιήθηκαν διάφορες τιμές Dropout Rate καθώς και Weight Decay. Οι καλύτερες διαμορφώσεις φαίνονται σε πίνακα 3α.

Inference phase:

Μετά την ολοκλήρωση της εκπαίδευσης του ο Autoencoder χρησιμοποιήθηκε ώστε να κάνει impute της ελλείπουσες τιμές του test dataset. Το test dataset χωρίστηκε επίσης σε sequences οι οποίες είχαν μέγεθος 66 (number of attributes) X 10 (sequence length) και ο autoencoder έκανε reconstruct τα δεδομένα αυτά. (Original space – Latent space – Back to original space).

Με στόχο την βελτίωση της συνολικής επίδοσης της διαδικασίας εκτελέστηκε το εξής: Κάνοντας εκ νέου χρήση του Mask οι θέσεις δεδομένων που προϋπήρχαν στο dataset αντικαταστάθηκαν στο imputed test dataset με τις αρχικές. Επομένως στο τελικό dataset μόνο οι τιμές που αρχικά έλειπαν έχουν προκύψει από την χρήση του μοντέλου και όλες οι υπόλοιπες είναι ταυτόσημες με τις αρχικές.

Στα παρακάτω διαγράμματα απεικονίζονται τα training / validation losses κατά την διαδικασία εκπαίδευσης.



2. Variational Autoencoders (VAE)

Αρχιτεκτονική:

Σχεδιάστηκαν κατά αντιστοιχία με τους Autoencoders δυο βασικές αρχιτεκτονικές με χρήση είτε δύο LSTM layers στον ENCODER καθώς και δύο LSTM layers στον DECODER, καθώς και ενός LSTM layer στον ENCODER και ενός LSTM layer στον DECODER. Μετά από κάθε LSTM layer εφαρμόστηκε ένα dropout layer του οποίου η τιμή p (0 – 1) αποτέλεσε υπερπαράμετρο. Ως activation function χρησιμοποιήθηκε σε όλα τα layers η tanh (default σε pytorch για LSTM).

Training phase:

Εφαρμόστηκαν επίσης κατά αντιστοιχία η αντικατάσταση ελλειπουσών τιμών από την mean() καθώς και η κανονικοποίηση τους με χρήση συναρτήσεως MinMaxScaler().

Το training dataset χωρίστηκε σε sequences οι οποίες είχαν μέγεθος 66 (number of attributes) X 10 (sequence length) και με αυτά τα sequences εφαρμόστηκαν στην είσοδο του VAE.

Η Loss function του VAE αποτελείται από δύο συνιστώσες, την BinnaryCrossEntropy (BCE) Loss (η οποία κατά αντιστοιχία με τους Autoencoders χρησιμοποιήθηκε η μάσκα ώστε να λαμβάνονται υπόψη επιλεκτικά μόνο οι παρατηρούμενες τιμές στον υπολογισμό) και την KLD που αντιστοιχεί στην Kullback-Leibler.

$$\text{kl\_divergence} = \alpha \times \sum(1 + \log(\sigma^2) - \mu^2 - \sigma^2)$$

Χρησιμοποιήθηκαν διαφορετικές τιμές a σε μία προσπάθεια βελτίωσης της συμπεριφοράς στην εκπαίδευση χωρίς αποτέλεσμα. Ως optimizer ορίστηκε επίσης ο Adam.

Παρόλο τον αρκετό πειραματισμό με διάφορες εναλλαγές υπερπαραμέτρων και διαφορετικές τοπολογίες δεν κατέστη δυνατό να δημιουργηθεί ένα αξιόπιστο μοντέλο. Ενδεικτικά στο παρακάτω διάγραμμα φαίνεται η “συμπεριφορά” του μοντέλου κατά την διάρκεια της εκπαίδευσης.

3. Generative Adversarial Networks (GAN)

Αρχιτεκτονική:

Χρησιμοποιήθηκε ως αρχιτεκτονική η χρήση δύο LSTM layers στον GENERATOR. Στον DISCRIMINATOR χρησιμοποιήθηκαν δύο LSTM layers και εν συνεχεία ένα linear layer. Η διάσταση των hidden layers ορίστηκε στο 256 τόσο για τον GENERATOR όσο και για τον DISCRIMINATOR. Μετά από κάθε LSTM layer εφαρμόστηκε ένα dropout layer του οποίου η τιμή p (0 – 1) αποτέλεσε υπερπαράμετρο. Ως activation function χρησιμοποιήθηκε σε όλα τα hidden layers η tanh (default σε pytorch για LSTM) και στο τελευταίο (linear) layer του DISCRIMINATOR χρησιμοποιήθηκε sigmoid activation function.

Training phase:

Εφαρμόστηκαν επίσης κατά αντιστοιχία η αντικατάσταση ελλειπουσών τιμών από την mean(), η κανονικοποίηση τους με χρήση συναρτήσεως MinMaxScaler().

Το training dataset χωρίστηκε σε sequences οι οποίες είχαν μέγεθος 66 (number of attributes) X 10 (sequence length) και με αυτά τα sequences εφαρμόστηκαν στην είσοδο του autoencoder.

Εφαρμόστηκε διαφορετικός λόγος εκπαίδευσης 1 προς 5 για το Generator / Discriminator (γίνεται update των βαρών του DISCRIMINATOR πέντε φορές πιο συχνά σε σχέση με τον GENERATOR).

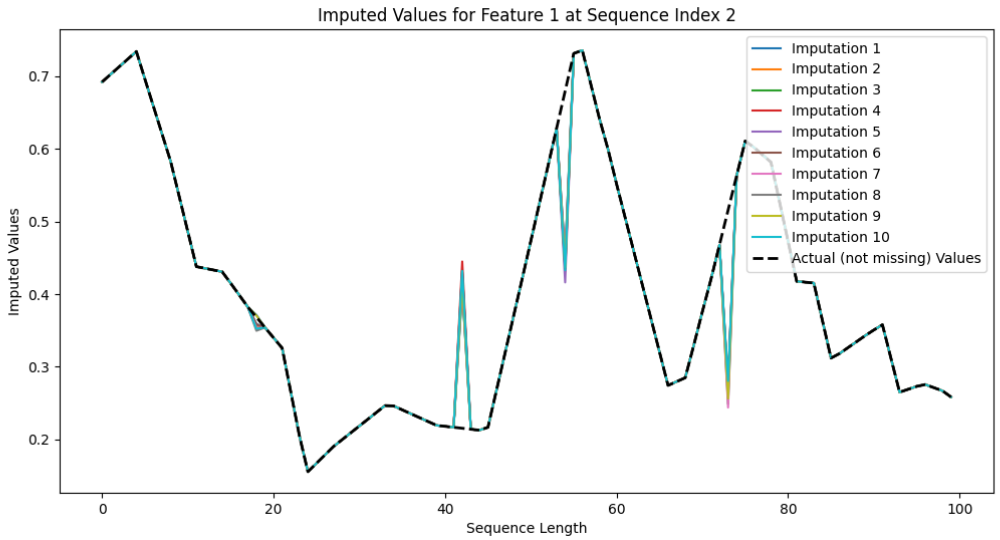
Ως Loss function χρησιμοποιήθηκε η BCE Loss, ενώ κατά το validation phase χρησιμοποιήθηκε η MSE. Ως optimizer ορίστηκε ο Adam τόσο στον GENERATOR όσο και για τον DISCRIMINATOR.

Inference phase:

Μετά την ολοκλήρωση της εκπαίδευσης του ο GENERATOR αποθηκεύτηκε σε μορφή pth και χρησιμοποιήθηκε ώστε να κάνει impute της ελλείπουσες τιμές του test dataset. Το test dataset χωρίστηκε επίσης σε sequences οι οποίες είχαν μέγεθος 66 (number of attributes) X 10 (sequence

length) και μαζί με θόρυβο (ίδιου structure) τροφοδοτήθηκε στον trained GENERATOR. Με στόχο την βελτίωση της συνολικής επίδοσης της διαδικασίας εκτελέστηκε το εξής: Αντίστοιχα με τον Autoencoder χρησιμοποιήθηκε η Mask ώστε οι θέσεις δεδομένων που προϋπήρχαν στο dataset να αντικατασταθούν στο imputed test dataset από τις αρχικές.

Παρόλο που χρειάστηκαν, με μεγάλη διαφορά τον μεγαλύτερο χρόνο εκπαίδευσης, σημάδι της αυξημένης υπολογιστικής ισχύος που απαιτούν, έδωσαν τα καλύτερα αποτελέσματα από όλα τα μοντέλα DL με όρους RMSE στον test dataset. Στο παρακάτω διάγραμμα φαίνονται τα αποτελέσματα του imputation για ένα τυχαίο feature με χρήση του εκπαιδευμένου GAN ώστε να αποτυπωθεί η διακύμανση των αποτελεσμάτων λόγω της στοχαστικής λειτουργίας τους.



Για αύξηση της ρεαλιστικότητας θεωρήθηκε πως τα datasets με τα διάφορα ποσοστά missing data ήταν η μοναδική πηγή δεδομένων κάθε φορά (η τεχνική masking χρησιμοποιήθηκε σε αυτά τα δεδομένα). Ως "οδηγός" κάθε κατηγορίας μοντέλων υπήρξαν οι επιδώσεις τους στο 10% missing dataset και εφόσον μία συγκεκριμένη διάταξη παρουσίαζε υποσχόμενα αποτελέσματα δοκιμάζονταν και στα διαφορετικά ποσοστά. (Είναι αντιληπτό από τον γράφων το "no free launch theorem" αλλά λόγω ελλείψεως χρόνου και πόρων επιλέχθηκε αυτή η στρατηγική).

Στους παρακάτω πίνακες 3α καθώς και 3β φαίνονται οι τιμές των υπερπαραμέτρων των μοντέλων που επικράτησαν. Δοκιμάστηκαν πού περισσότεροι συνδυασμοί άλλα για οικονομία χώρου παρουσιάζονται οι επικρατέστεροι.

Model	AE	AE	AE	AE	AE	AE	AE	AE
Missing percentage	10%	10%	10%	10%	10%	20%	30%	50%
Type	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM
Num of Layers	2	2	1	1	2	1	1	1
Num Epochs	200	200	600	600	600	600	600	600
Dropout Rate	0	0.2	0.1	0.2	0.2	0.1	0.1	0.1
Patience	10	10	10	10	10	10	10	10

Embedding Dim	24	24	40	40	20	40	40	40
Batch size	128	128	32	32	128	32	32	32
Learning Rate	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001
Weight Decay	0	0	0	0	0	0	0	0
Seq Length	100	100	10	10	100	10	10	10
Validation Loss	0.0152	0.0162	0.0109	0.0238	0.1552	0.0117	0.0105	0.0145

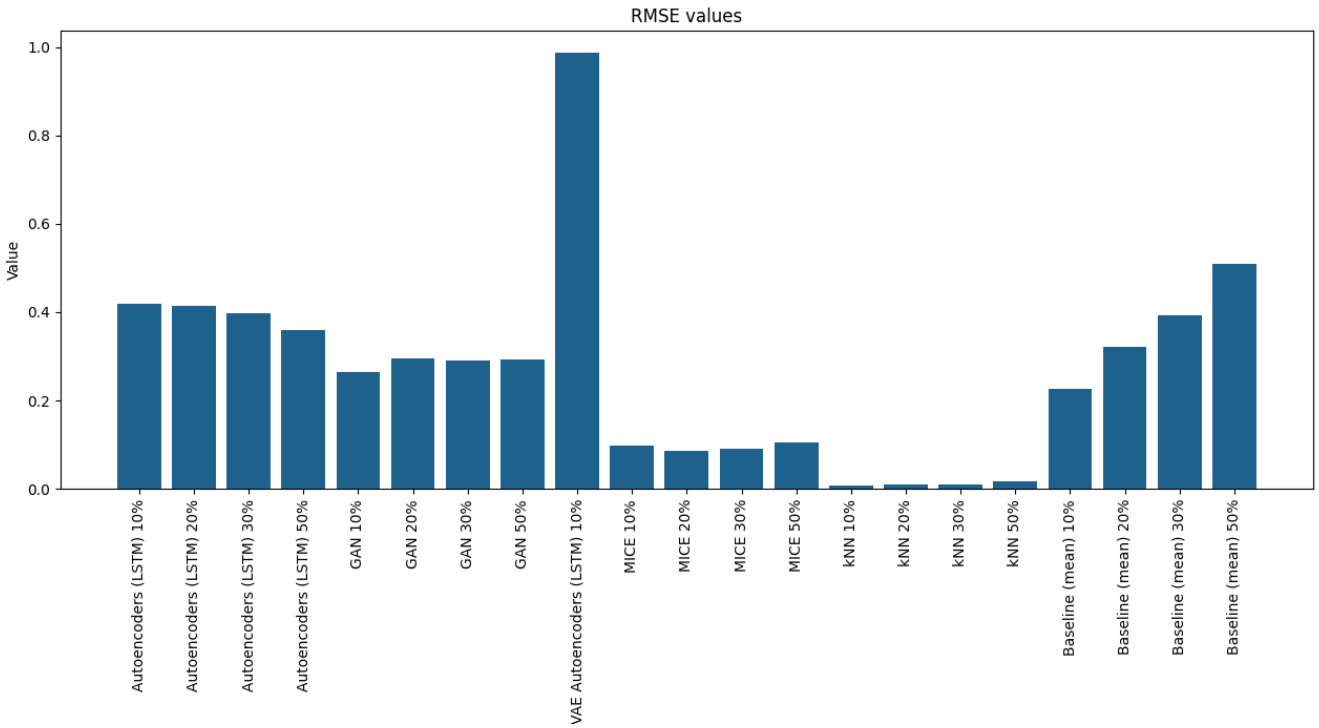
Πιν.3α Εύρος των Hyperparameters

Model	VAE	GAN	GAN	GAN	GAN
Missing percentage	10%	10%	20%	30%	50%
Type	LSTM	LSTM	LSTM	LSTM	LSTM
Num of Layers	2	2	2	2	2
Num Epochs	400	200	200	200	200
Dropout Rate	0.1	0.1	0.1	0.1	0.1
Patience	10	10	10	10	10
Embedding Dim/hidden Dim (GAN)	24	128	128	128	128
Batch size	32	128	128	128	128
Learning Rate	0.001	0.0001	0.0001	0.0001	0.0001
Weight Decay	0	0	0	0	0
Seq Length	100	10	100	100	100
Validation Loss	0.043	0.0399	0.0412	0.0395	0.0388

Πιν.3β Εύρος των Hyperparameters

Αποτελέσματα / Σχολιασμός

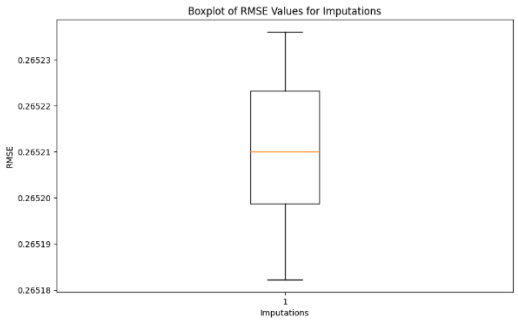
Στο παρακάτω διάγραμμα No1 φαίνονται συγκεντρωτικά τα αποτελέσματα των επιδόσεων όλων των μοντέλων σε όρους RMSE σε σχέση με το αρχικό dataset που δεν περιείχε καθόλου ελλείπουσες τιμές. (Στην περίπτωση των DL μοντέλων αναφερόμαστε προφανώς στο test dataset).



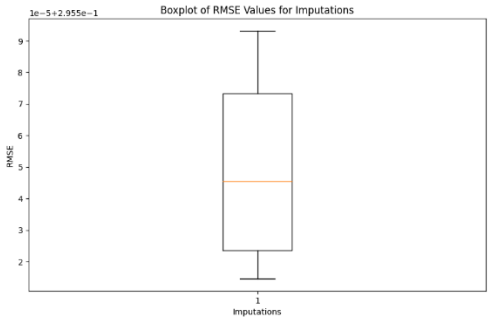
Διαγρ.1 Αποτελέσματα με όρους RMSE

Από την ανάλυση των αποτελεσμάτων προκύπτουν τα εξής συμπεράσματα:

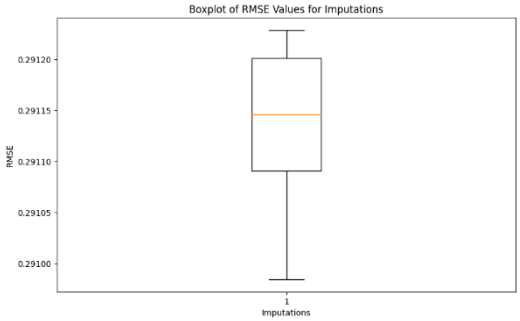
- Η καλύτερη μέθοδος data imputation σε αυτό το dataset είναι η kNN με μεγάλη διαφορά από τις υπόλοιπες μεθόδους. Το αρνητικό αυτής της μεθόδου με τον τρόπο που εφαρμόστηκε είναι ο ιδιαίτερα μεγάλος χρόνος εκτέλεσης (πολύ περισσότερος ακόμα και από την υπολογιστικά ακριβότερη μέθοδο DL που εφαρμόστηκε), διότι δεν είναι δυνατός ο παραλληλισμός των υπολογισμών.
- Η μέθοδος IterativeImputer έδωσε τα δεύτερα καλύτερα αποτελέσματα με σαφώς χαμηλότερους χρόνους εκτέλεσης (συγκρίσιμους των DL μεθόδων)
- Από τα μοντέλα DL, η χρήση GAN είχε τα καλύτερα αποτελέσματα, τα οποία από το ποσοστό 20% έως 50% έμειναν πρακτικά τα ίδια (η αύξηση των missing data δεν επέφερε μείωση της ποιότητας των αποτελεσμάτων). Τα αποτελέσματα των GAN αποτελούν την μέση τιμή των αποτελεσμάτων χρήσης τους 10 φορές. Στα διαγράμματα 2 έως 5 προβάλλονται τα αποτελέσματα του inference phase των GAN για τα διάφορα ποσοστά missing data.



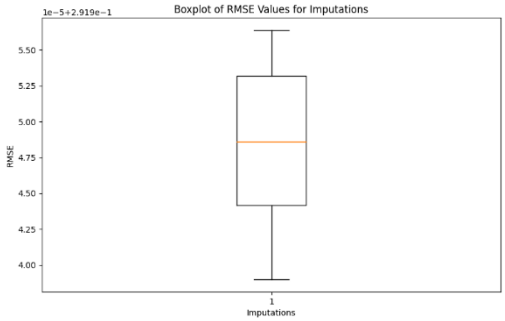
Διαγρ.2 10% missing data



Διαγρ.3 20% missing data



Διαγρ.4 30% missing data



Διαγρ.5 50% missing data

Συμπερασματικά, καθόσον οι μέθοδοι kNN και IterativeImputer για να εφαρμοστούν απαιτούν να προϋπάρχει το πλήρες dataset εφαρμογής, δεν είναι δυνατόν να χρησιμοποιηθούν σε ένα streaming σύστημα το οποίο θα είναι σε θέση, σε πραγματικό ή σχεδόν πραγματικό χρόνο, να εκτελεί missing data imputation. Σε περιπτώσεις που χρειάζεται τα δεδομένα να επεξεργαστούν σε πολλαπλά συστήματα σε σειρά (π.χ. περαιτέρω επεξεργασία από ένα φυσικό

μοντέλο στα πλαίσια δημιουργίας ενός ψηφιακού διδύμου) τα δίκτυα GAN αποτελούν αξιόπιστη λύση καθώς επιτυγχάνουν επιδόσεις καλύτερες του baseline και επιτρέπουν την δημιουργία ενός pretrained μοντέλου που μπορεί να εφαρμοστεί απευθείας στο pipeline και με μικρούς χρόνους απόκρισης.