

UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

**VULNÉRABILITÉ ET ROBUSTESSE DES MODÈLES
D'APPRENTISSAGE POUR LA RECONNAISSANCE DE GENRE EN
RÉALITÉ VIRTUELLE**

MÉMOIRE
PRÉSENTÉ
COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN INFORMATIQUE

PAR
EMNA KRAIEM

AOÛT 2026

UNIVERSITÉ DU QUÉBEC EN OUTAOUAIS

Département d'informatique et d'ingénierie

Ce mémoire intitulé :

**VULNÉRABILITÉ ET ROBUSTESSE DES MODÈLES
D'APPRENTISSAGE POUR LA RECONNAISSANCE DE GENRE EN
RÉALITÉ VIRTUELLE**

présenté par
EMNA KRAIEM

pour l'obtention du grade de Maîtrise ès sciences (M.Sc.)

a été évalué par un jury composé des personnes suivantes :

Dr. Mohand Saïd Allili Président du jury
Dr. Tiago Falk Membre du jury
Dr. Alan Davoust Directeur de recherche

Mémoire accepté en : AOÛT 2026

Dédicace

Ce travail est dédié à ma famille

Table des matières

Liste des figures	viii
Liste des tableaux	xii
Liste des abréviations, sigles et acronymes	xiv
Résumé	xv
1 Introduction	1
1.1 Motivation	1
1.2 Problématique	2
1.3 Contributions	3
1.4 Organisation du mémoire	4
2 Fondements théoriques et méthodologiques	6
2.1 Fondements de la réalité virtuelle et des signaux de mouvement	7
2.2 Caractéristiques et défis des données RV	7
2.3 Fondements de l'apprentissage automatique pour les signaux de réalité virtuelle	8
2.3.1 Modèles d'apprentissage automatique classiques	9
2.3.2 Modèles neuronaux profonds	10
2.3.3 Modèles temporels	11
2.4 Attaques adversariales : principes généraux	12
2.5 Fondements des mécanismes de défense	13
2.5.1 Catégorie 1 : Transformation des données	14
2.5.1.1 Winsorization	14
2.5.1.2 Z-score clipping	14

2.5.1.3	Log transform	15
2.5.1.4	Feature normalization	15
2.5.1.5	Feature squeezing	16
2.5.1.6	PCA reconstruction	16
2.5.1.7	Median filtering	17
2.5.2	Catégorie 2 : Détection et rejet	17
2.5.2.1	Local outlier factor (LOF)	18
2.5.2.2	One-Class SVM	18
2.5.2.3	Distance de mahalanobis	19
2.5.2.4	Test statistique de Kolmogorov-Smirnov	19
2.5.3	Catégorie 3 : Méthodes basées sur le vote	19
2.5.3.1	Ensemble de modèles	20
2.5.3.2	Bootstrap ensemble	20
2.5.3.3	Bagging defense	20
2.5.3.4	Feature subspace voting	21
2.5.3.5	Gaussian smoothing	21
2.5.4	Catégorie 4 : Entraînement robuste	21
2.5.4.1	Adversarial training	22
2.5.4.2	Label smoothing	22
2.5.4.3	Mixup training	22
3	État de l’art	24
3.1	Prédiction du genre à partir de signaux de réalité virtuelle	24
3.2	Perturbations comportementales et limitations des signaux en RV	25
3.2.1	Attaques adversariales en biométrie comportementale	26
3.3	Stratégies de défense appliquées aux données de RV	27
3.4	Synthèse critique et positionnement du mémoire	27
4	Analyse et prétraitement des données	29
4.1	Présentation des jeux de données	29
4.1.1	Présentation générale	29
4.1.2	Jeu de données FAST	30
4.1.3	Organisation et annotation des labels dans le jeu de données FAST	31
4.1.4	Jeu de données Alyx	32
4.1.5	Organisation spécifique du jeu de données <i>Who Is Alyx</i>	33

4.2	Analyse descriptive des jeux de données	35
4.2.1	Statistiques descriptives	35
4.2.2	Distribution des classes	36
4.2.3	Visualisations exploratoires	36
4.2.3.1	Dataset Who is Alyx	36
4.2.3.2	Dataset FAST	39
4.3	Préparation et nettoyage des données	41
4.3.1	Nettoyage des valeurs manquantes et aberrantes	41
4.3.2	Harmonisation temporelle	42
4.3.3	Nettoyage structurel	42
4.3.4	Ajout de caractéristiques dérivées	43
4.3.4.1	Caractéristiques morphologiques	43
4.3.4.2	Caractéristiques d'asymétrie	43
4.3.5	Normalisation des données	44
4.4	Synthèse du pipeline expérimental	44
5	Reconnaissance du genre à partir du mouvement en RV	45
5.1	Expérimentations Cross-Dataset : Entraînement sur Alyx, Test sur FAST	46
5.1.1	Préparation des données	46
5.1.2	Approche : Agrégation Temporelle par Seconde sur 49 Combinaisons	48
5.1.2.1	Construction des Features	48
5.1.2.2	Combinaisons de Trackers et Représentations Spatiales .	49
5.1.2.3	Résultats	49
5.1.2.4	Visualisation par Heatmap	50
5.2	Représentation tabulaire des signaux de mouvement	52
5.2.1	Extraction statistique	52
5.2.2	Caractéristiques morphologiques et dynamiques dérivées	54
5.2.3	Agrégation par session	56
5.2.4	Sélection d'attributs	56
5.2.5	Validation croisée imbriquée et prévention du surapprentissage . .	57
5.2.6	Modèles classiques et résultats	58
5.2.7	Expérimentation : split aléatoire	62
5.3	Représentation temporelle des signaux de mouvement	65
5.3.1	Caractéristiques dynamiques dérivées	65

5.3.2	Génération des séquences	69
5.3.3	Normalisation des séquences	69
5.3.4	Résultats des modèles temporels	69
5.4	Comparaison des modèles	72
5.5	Analyse des variables les plus discriminantes	75
5.6	Conclusion du chapitre	76
6	Menaces et perturbations sur la reconnaissance du genre en RV	78
6.1	Calibration des paramètres des perturbations	79
6.1.1	Identification des familles de perturbations	79
6.1.2	Choix des distributions probabilistes	80
6.1.3	Calibration des amplitudes	80
6.2	Scénarios de perturbations comportementales	82
6.2.1	Regroupement des perturbations en familles réalistes	83
6.2.2	Famille 1 - Variations comportementales naturelles (non intentionnelles)	84
6.2.3	Famille 2 - Modifications volontaires du comportement utilisateur	85
6.2.4	Famille 3 - Perturbations matérielles, système et environnementales	86
6.3	Évaluation de la performance et de la robustesse	87
6.4	Impact des perturbations comportementales sur les modèles classiques . .	89
6.4.1	MLP	89
6.4.2	SVM à noyau RBF	90
6.4.3	Gradient Boosting	91
6.4.4	LightGBM	93
6.4.5	Random Forest	94
6.4.6	XGBoost	95
6.4.7	Résultats globaux :	96
6.5	Impact des perturbations comportementales sur les modèles temporels . .	101
6.6	Conclusion du chapitre	104
7	Stratégies de Défense	106
7.1	Performances de référence	107
7.1.1	Impact des attaques sans défense	107
7.2	Évaluation des défenses	108
7.2.1	Vue d'ensemble : gain moyen par défense	109

7.2.2	Analyse par type d'attaque et par modèle	110
7.2.3	Défenses empilées : combinaison séquentielle	113
7.2.3.1	Gain moyen global	114
7.2.3.2	Gain par modèle	114
7.2.3.3	Gain par attaque	115
7.2.3.4	Comparaison par niveau d'empilement	116
7.3	Système Adaptatif de Détection et Défense	120
7.3.1	Architecture du Pipeline	121
7.3.2	Évaluation	124
7.3.2.1	Pipeline entraînement/test	124
7.3.2.2	Taux de détection	125
7.3.2.3	Gain de robustesse	126
7.3.3	Limitations :	127
7.4	Bilan Comparatif : Système Adaptatif vs Défenses Individuelles	127
7.5	Analyse de la confiance de prédiction	128
7.6	Synthèse et Choix Final	130
7.7	Estimation du risque de tromperie	132
7.8	Conclusion	133
8	Conclusion	135
8.1	Résumé	135
8.2	Réponses aux questions de recherche	137
8.3	Recommandations pratiques	138
8.4	Limitations et travaux futurs	139
8.5	Publication dérivée de ce travail	141
	Bibliographie	142
A	Famille 4 - Attaques guidées par le modèle (semi-informées, coût computationnel élevé)	154
B	Analyse détaillée par défense	156
B.1	Adversarial training	156
B.2	LOF + Rejet	157
B.3	PCA reconstruction	158

B.4	Squeezing défense	159
B.5	Smoothing	160
B.6	Ensemble	161
B.7	Median filter	162
B.8	Winsorization	163
B.9	Z-score	164
B.10	Log transform	165
B.11	Feature normalization	166
B.12	OC-SVM (One-Class SVM, SVM à une classe)	166
B.13	Mahalanobis	167
B.14	Statistical test	168
B.15	Bagging	169
B.16	Subspace	170
B.17	Bootstrap	171
B.18	Label smoothing	172
B.19	Mixup	173
C	Autres versions du système adaptatif de défense	175
C.1	Version 2 - Détection par Centroïdes	175
C.1.1	Architecture du Pipeline V2	175
C.1.1.1	Étape 1 : Détection - Modifications :	176
C.1.1.2	Étape 2 : Sélection de la défense - Dictionnaire mis à jour :176	
C.1.1.3	Étape 3 : Application de la défense :	176
C.1.2	Taux de détection :	177
C.1.3	Gain de robustesse :	177
C.1.4	Limitations :	178
C.2	Version 3 - Défense par Modèle	178
C.2.1	Architecture du Pipeline V3	179
C.2.1.1	Étape 1 : Détection - Modifications par rapport à V2 :	179
C.2.1.2	Étape 2 : Sélection de la défense - Dictionnaire par modèle :181	
C.2.1.3	Étape 3 : Application de la défense :	181
C.2.2	Taux de détection :	181
C.2.3	Gain de robustesse :	182
C.2.4	Limitations :	183

C.3	Version 4 - Correction de Behavioral Drift	183
C.3.1	Architecture du Pipeline V4	183
C.3.1.1	Étape 1 : Détection - Unique modification par rapport à V3 :	183
C.3.1.2	Étape 2 : Sélection de la défense - Dictionnaire inchangé :	184
C.3.1.3	Étape 3 : Application de la défense :	184
C.3.2	Taux de détection :	185
C.3.3	Gain de robustesse :	185
C.3.4	Limitations :	185
C.4	Version 5 - Système Hybride Optimisé	186
C.4.1	Axe 1 : Détecteur hybride V2+V3 :	186
C.4.2	Axe 2 : Dictionnaire de défenses optimisé par modèle :	187
C.4.3	Architecture du Pipeline V5	188
C.4.3.1	Étape 1 : Détection - Détecteur hybride :	188
C.4.3.2	Étape 2 : Sélection de la défense - Dictionnaire V5 :	188
C.4.3.3	Étape 3 : Application de la défense :	188
C.4.4	Taux de détection :	189
C.4.5	Gain de robustesse :	189
C.4.6	Limitation persistante :	190

Liste des figures

4.1	Structure du dataset FAST : données de tracking capturées à chaque timestamp (tête, main gauche, main droite) et métadonnées de session (genre et identifiant utilisateur) ajoutées une fois par session.	31
4.2	Structure du dataset Who is Alyx : données de tracking capturées à chaque timestamp (tête, main gauche, main droite, eye tracking) et métadonnées de session (identifiant joueur, session et genre).	34
4.3	Distribution de la variable <code>hmd_pos_y</code> - dataset Who is Alyx.	37
4.4	Évolution temporelle de la hauteur de tête (extrait de session) - dataset Who is Alyx.	37
4.5	Matrice de corrélation des variables brutes et dérivées - Who is Alyx. . .	38
4.6	Distribution de la variable <code>Head_position_y</code> - dataset FAST.	39
4.7	Évolution temporelle de <code>Head_position_y</code> - participant 10068 (FAST). .	39
4.8	Matrice de corrélation des variables de position et d'orientation - dataset FAST.	40
5.1	Balanced Accuracy moyenne (Alyx→FAST et FAST→Alyx) par combinaison de trackers et représentation spatiale, évaluée avec un SVM optimisé ($C = 10$, $\gamma = 0.001$). Les cellules vertes indiquent les meilleures performances, les cellules rouges les plus faibles, et N/A les combinaisons inexistantes.	51
5.2	Comparaison entre la BAcc moyenne en validation croisée imbriquée (CV BalancedAcc, avec écart-type) et la BAcc sur le jeu de test final (Test BalancedAcc), pour les six modèles évalués.	61
5.3	Accuracy des différents modèles temporels sur le jeu de test.	71
5.4	Top 20 features les plus importantes en moyenne sur tous les modèles. . .	75

6.1	BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - MLP.	89
6.2	BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - SVM (RBF).	90
6.3	BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - Gradient Boosting.	92
6.4	BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - LightGBM.	93
6.5	BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - Random Forest.	94
6.6	BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - XGBoost.	95
6.7	Chute de <i>balanced accuracy</i> (Δ BAcc) par modèle et par type d'attaque. Le rouge indique une attaque très efficace (forte dégradation des performances), tandis que le vert indique une attaque peu efficace (valeurs négatives).	97
6.8	Taux de basculement (<i>FlipRate</i>) par modèle et par type d'attaque, mesurant la proportion de prédictions dont la classe change sous l'effet de la perturbation. Le rouge indique une attaque très déstabilisante.	97
6.9	Confiance moyenne de prédiction sous attaque par modèle et par type d'attaque. Le vert indique une confiance élevée, le rouge une confiance faible.	99
6.10	Écart absolu entre confiance et <i>balanced accuracy</i> sous attaque par modèle et par type d'attaque. Le rouge indique une mauvaise calibration (confiance éloignée de la performance réelle, dans un sens ou dans l'autre), le vert une confiance cohérente avec la performance.	100
6.11	Impact des perturbations comportementales sur les modèles temporels. .	101
6.12	Détails de l'impact des sept attaques comportementales les plus représentatives sur les modèles temporels, mesuré en <i>balanced accuracy</i> après attaque (BAcc_{adv} , bleu foncé) et en confiance de prédiction sur les exemples mal classés ($\text{Confiance}_{\text{adv}}$, orange).	102

7.1	BAcc sous perturbation sans défense par modèle et par attaque. Les cellules vertes indiquent une BAcc élevée (faible impact de l'attaque) et les cellules rouges une BAcc faible (fort impact).	108
7.2	Gain moyen en BAcc par défense.	109
7.3	Gain moyen en BAcc par défense (lot V2).	110
7.4	Gain moyen en BAcc par défense et par type d'attaque (vert = défense efficace, rouge = défense nuisible).	111
7.5	Gain moyen en BAcc par défense et par modèle (vert = défense efficace, rouge = défense nuisible).	111
7.6	Gain moyen en BAcc par défense et par type d'attaque (lot V2).	112
7.7	Gain moyen en BAcc par défense et par modèle (lot V2).	112
7.8	Gain moyen en BAcc par empilement de défenses, toutes attaques et modèles confondus. Les empilements sont triés par gain décroissant.	114
7.9	Gain moyen par empilement et par modèle (vert = efficace, rouge = nuisible).	115
7.10	Gain moyen par empilement et par type d'attaque (vert = efficace, rouge = nuisible).	116
7.11	Comparaison des gains par niveau d'empilement. Chaque groupe représente un niveau : Niveau 2 (une seule transformation avant la prédiction), Niveau 3 (deux transformations en séquence), Niveau 4 (trois transformations en séquence).	117
7.12	BAcc avant attaque (Clean), après attaque sans défense, et avec le meilleur empilement, par modèle et par attaque.	118
7.13	BAcc avant attaque (Clean), après attaque sans défense, et avec le meilleur empilement, par type d'attaque et par modèle.	119
7.14	BAcc moyenne sur toutes les attaques, pour les trois situations (Clean, Sans défense, Meilleur empilement), par modèle.	119
7.15	Association entre chaque type d'attaque détecté et la défense sélectionnée dans le dictionnaire δ^* . La valeur 1 (case verte) indique que la défense correspondante est celle choisie pour cette attaque, 0 sinon.	123
7.16	Pipeline du système adaptatif de détection et défense.	125
B.1	Impact de l'Adversarial Training.	157
B.2	Impact du LOF + Rejet.	158

B.3	Impact de la PCA Reconstruction.	159
B.4	Impact du Squeezing Défense.	160
B.5	Impact du Smoothing.	161
B.6	Impact de la défense Ensemble.	162
B.7	Impact du Median Filter.	163
B.8	Impact de la Winsorization.	164
B.9	Impact du Z-score.	164
B.10	Impact du Log Transform.	165
B.11	Impact de la Feature Normalization.	166
B.12	Impact de l'OC-SVM.	167
B.13	Impact de la distance de Mahalanobis.	168
B.14	Impact du Statistical Test.	169
B.15	Impact du Bagging.	170
B.16	Impact du Subspace.	171
B.17	Impact du Bootstrap.	172
B.18	Impact du Label Smoothing.	173
B.19	Impact du Mixup.	174

Liste des tableaux

5.1	Correspondance des colonnes entre le dataset FAST et le dataset Alyx . .	48
5.2	Résultats cross-dataset par combinaison de trackers et représentation spatiale (SVM optimisé, triés par moyenne décroissante)	50
5.3	Résumé des caractéristiques extraites par catégorie après nettoyage . . .	55
5.4	Performances des modèles classiques après validation croisée imbriquée (sans attaques).	60
5.5	Performances des modèles avec split aléatoire stratifié (StratifiedKFold , $k = 5$). k désigne le nombre de features sélectionnées par SelectKBest . .	62
5.6	Comparaison des performances (BAcc) selon le protocole de split.	63
5.7	Performances comparées des modèles classiques, neuronaux et temporels.	73
6.1	Famille 1 - Variations naturelles (non intentionnelles).	84
6.2	Modifications volontaires du comportement utilisateur.	85
6.3	Perturbations matérielles, système et environnementales.	86
7.1	Performance de référence des six modèles sur données non perturbées . .	107
7.2	Taux de détection du système adaptatif par type d'attaque	126
7.3	Gain moyen du système adaptatif par modèle (moyenne sur les 7 attaques)	126
7.4	Confiance moyenne selon la justesse de la prédiction (Clean vs Sans défense vs système adaptatif)	128
7.5	Probabilité de tromperie par modèle, sous l'hypothèse d'une attaque choisie au hasard parmi les sept types testés	132
A.1	Famille 6 - Attaques semi-informées (plus coûteuses).	155
C.1	Dictionnaire de défenses du système adaptatif V2 - modifications par rapport à V1	177

C.2	Comparaison des taux de détection V1 vs V2 par type d'attaque	177
C.3	Gain moyen du système adaptatif V2 par modèle (moyenne sur les 7 attaques)	178
C.4	Dictionnaire complet de défenses du système adaptatif V3 par modèle et par attaque. Les cases en gras indiquent les modifications par rapport à V2.	181
C.5	Comparaison des taux de détection V1 vs V2 vs V3 par type d'attaque .	182
C.6	Gain moyen du système adaptatif V3 par modèle (moyenne sur les 7 attaques)	182
C.7	Dictionnaire de défenses du système adaptatif V4.	184
C.8	Comparaison des taux de détection V1 vs V2 vs V3 vs V4 par type d'attaque	185
C.9	Comparaison des gains moyens V1 vs V2 vs V3 vs V4 par modèle	186
C.10	Dictionnaire complet des défenses du système adaptatif V5 par modèle et par attaque	188
C.11	Taux de détection du système adaptatif V5 par type d'attaque, comparé aux versions précédentes	189
C.12	Gain moyen du système adaptatif V5 par modèle, comparé aux versions précédentes (moyenne sur les 7 attaques)	190

Liste des abréviations, sigles et acronymes

RV Réalité Virtuelle

VR Virtual Reality

HMD Head Mounted Display

CNN Convolutional Neural Network

LSTM Long short-term memory

FGSM Fast Gradient Sign Method

BIM Basic Iterative Method

PGD Projected Gradient Descent

JSMA Jacobian-based Saliency Map Attack

CW-L2 Carlini & Wagner L2 Attack

MLP Multi-Layer Perceptron

CNN-LSTM Convolutional Neural Network with Long Short-Term Memory

ResNet1D One-Dimensional Residual Neural Network

TimesNet Temporal 2D-Variation Network for Time Series

RF Random Forest

XGBoost Extreme Gradient Boosting

LightGBM Light Gradient Boosting Machine

GB Gradient Boosting

Résumé

La réalité virtuelle (RV) offre de nouvelles opportunités pour l'analyse comportementale et la reconnaissance d'identité. Les équipements RV capturent en continu les mouvements corporels des utilisateurs, ce qui permet non seulement d'analyser leur comportement, mais aussi de les authentifier de manière continue au sein d'un espace immersif. Ces données peuvent en effet révéler des attributs personnels tels que le genre, ce qui soulève des enjeux importants en matière de vie privée : une personne évoluant dans un environnement virtuel pourrait voir son genre inféré à son insu à partir de sa simple façon de bouger.

Il a été démontré que des modèles d'apprentissage automatique sont capables de déterminer le genre d'une personne à partir des données de mouvement captées par des équipements de RV. Dans ce contexte, on peut faire l'hypothèse qu'une personne cherche à tromper un tel modèle en modifiant volontairement sa façon de bouger. Ce mémoire étudie dans quelle mesure ces modèles restent fiables face à ce type de manipulation, en testant plusieurs familles de modèles : des approches classiques (Random Forest, XGBoost, LightGBM, Gradient Boosting, MLP) et des modèles qui tiennent compte de la structure temporelle des signaux de mouvement (LSTM, CNN-LSTM, ResNet1D, TimesNet). Plus d'une centaine de scénarios ont été testés, incluant des modifications volontaires de posture, de synchronisation des mains ou de dynamique gestuelle, ainsi que des perturbations numériques simulant des erreurs de calibration du matériel.

Les résultats montrent que tous les modèles subissent une dégradation importante de leurs performances lorsque les mouvements sont altérés. Aucun modèle n'est totalement robuste, et les performances varient beaucoup selon le type de modification et l'architecture du modèle.

Pour faire face à ces vulnérabilités, plusieurs mécanismes de défense ont été développés et évalués. Dans un premier temps, dix-neuf défenses appliquées séparément ont été testées : certaines nettoient les données avant la prédiction, d'autres détectent et re-

jettent les exemples suspects, d'autres encore font voter plusieurs modèles ou renforcent le modèle dès l'entraînement. Dans un second temps, ces défenses ont été combinées en les appliquant l'une après l'autre sur le même exemple, ce qui a permis d'améliorer encore davantage la robustesse. Enfin, un système adaptatif a été conçu et amélioré à travers cinq versions successives. Ce système détecte automatiquement le type de manipulation et choisit la défense la plus adaptée selon le modèle utilisé, offrant le meilleur résultat parmi toutes les approches testées.

Ce mémoire montre qu'il est possible de renforcer significativement la robustesse des modèles de classification du genre en RV grâce à des stratégies de défense bien choisies. Il propose un cadre d'évaluation complet et des solutions concrètes pour mieux protéger les systèmes de reconnaissance d'identité dans les environnements immersifs.

Abstract

Virtual reality (VR) offers new opportunities for behavioral analysis and identity recognition. VR devices continuously capture users' body movements, making it possible not only to analyze their behavior but also to authenticate them on an ongoing basis within an immersive space. This movement data can reveal personal attributes such as gender, raising important privacy concerns : a person moving through a virtual environment could have their gender inferred without their knowledge, simply from the way they move.

It has been shown that machine learning models can determine a person's gender from movement data captured by VR devices. In this context, one can assume that a person might try to fool such a model by deliberately changing the way they move. This thesis examines how reliable these models remain when faced with this type of manipulation, by testing several families of models : standard approaches (Random Forest, XGBoost, LightGBM, Gradient Boosting, MLP) and models that take into account the time-based structure of movement signals (LSTM, CNN-LSTM, ResNet1D, TimesNet). Over a hundred scenarios were tested, including deliberate changes in posture, hand timing, and gesture dynamics, as well as numerical perturbations simulating hardware calibration errors.

The results show that all models experience a significant drop in performance when movements are altered. No model is fully robust, and performance varies considerably depending on the type of modification and the model architecture.

To address these weaknesses, several defense mechanisms were developed and evaluated. First, nineteen defenses applied individually were tested : some clean the input data before making a prediction, others detect and reject suspicious examples, some combine votes from multiple models, and others make the model more resistant during training. Second, these defenses were combined by applying them one after another on the same example, which further improved robustness. Finally, an adaptive system was

designed and refined through five successive versions. It automatically detects the type of manipulation and selects the most appropriate defense for the model in use, giving the best results among all approaches tested.

This work shows that the robustness of gender classification models in VR can be significantly improved with well-chosen defense strategies. It provides a complete evaluation framework and practical solutions to better protect identity recognition systems in immersive environments.

Chapitre 1

Introduction

1.1 Motivation

L'essor rapide des technologies immersives a profondément transformé la manière dont les individus interagissent dans des environnements virtuels. Les casques de réalité virtuelle (Head Mounted Device, HMD) et les contrôleurs manuels capturent en continu des signaux de mouvement détaillés, tels que les positions, rotations, vitesses et dynamiques corporelles, initialement conçus pour améliorer l'immersion et la synchronisation des avatars.

Cependant, plusieurs travaux ont montré que ces données constituent également une forme de biométrie comportementale hautement distinctive. Les signaux de tracking en RV permettent d'identifier de manière fiable un utilisateur parmi une large population, même sur de longues périodes temporelles [1, 2, 3]. D'autres études ont démontré qu'il est possible de déterminer le genre d'une personne uniquement à partir de sa façon de bouger dans un environnement virtuel [4, 5, 6].

Dans certains environnements immersifs, ces capacités peuvent être intégrées à des mécanismes d'authentification continue. Par exemple, un système pourrait analyser les mouvements d'un utilisateur en temps réel afin de vérifier automatiquement son genre et de contrôler l'accès à certaines fonctionnalités sociales ou à des espaces restreints. Dans ce contexte, une question intuitive se pose : *en supposant qu'un système d'authentification en RV cherche à vérifier le genre d'un utilisateur de manière continue, est-ce qu'un utilisateur averti peut tromper le système en modifiant volontairement sa façon de bouger ?*

Cette question se distingue des attaques adversariales classiques étudiées en vision par ordinateur [7, 8, 9], où les perturbations sont numériques et invisibles à l’œil nu. Ici, les modifications sont comportementales et physiquement réalisables par n’importe quel utilisateur : changer sa posture, ralentir ses gestes, modifier la synchronisation de ses mains. Bien que quelques travaux aient commencé à explorer cette problématique [10, 11, 12], la robustesse des modèles de classification du genre face à ce type de manipulation reste peu étudiée.

Ce mémoire s’attaque précisément à cette question en évaluant dans quelle mesure les modèles de classification du genre entraînés sur des données de mouvement RV peuvent être trompés par un utilisateur qui modifie volontairement ses mouvements, et en développant des mécanismes de défense adaptés pour contrer ces manipulations.

1.2 Problématique

L’étude de l’inférence d’attributs sensibles à partir de données de mouvement en réalité virtuelle demeure encore limitée, en particulier en ce qui concerne la robustesse des modèles face à des perturbations numériques, comportementales ou physiques affectant les signaux de tracking. Les données de suivi de mouvement en RV présentent en effet des propriétés structurelles spécifiques : une forte dépendance temporelle, une variabilité inter-utilisateur marquée et une sensibilité aux modifications physiques ou comportementales [13, 14]. Ces caractéristiques distinguent fondamentalement les signaux RV des données visuelles statiques traditionnellement étudiées en apprentissage automatique.

Ces données sont également soumises à des sources de perturbations spécifiques liées au tracking et à l’environnement immersif, telles que le bruit des capteurs, les pertes temporaires de suivi, les erreurs de calibration ou encore les variations de latence dans le système de capture [15, 16]. Les stratégies de défense proposées dans la littérature, normalisation robuste et réduction de dimension [17, 18], randomized smoothing [19], détection d’anomalies [20] et apprentissage adversarial [9, 21] ont principalement été développées pour des données d’images ou tabulaires classiques, et leur adaptation aux signaux de mouvement en RV reste peu explorée.

Dans ce contexte, il devient nécessaire d’adopter une démarche expérimentale rigoureuse visant à caractériser la vulnérabilité des modèles de reconnaissance du genre en réalité virtuelle face à des perturbations comportementales réalistes, et à évaluer dans

quelle mesure des mécanismes de défense adaptés permettent de restaurer leur robustesse. Ce travail s’articule autour des trois questions de recherche suivantes :

- **RQ1** : Dans quelle mesure les données de mouvement issues de la réalité virtuelle permettent-elles de classifier le genre des utilisateurs à l’aide de modèles d’apprentissage automatique, en l’absence de perturbations comportementales ?
- **RQ2** : Quel est l’impact de perturbations comportementales réalistes sur les performances et la fiabilité des modèles de reconnaissance du genre à partir de données de mouvement en réalité virtuelle ?
- **RQ3** : Dans quelle mesure l’intégration de stratégies de défense permet-elle d’améliorer la robustesse des modèles de classification du genre face aux perturbations comportementales dans un contexte de réalité virtuelle ?

1.3 Contributions

Ce mémoire propose une analyse systématique de la robustesse des modèles de classification du genre à partir de données de mouvement en réalité virtuelle.

Premièrement, nous confirmons que les données biomécaniques issues du suivi de mouvement en RV permettent une classification du genre avec de bonnes performances en conditions normales, en cohérence avec les travaux antérieurs sur l’identifiabilité en RV [1, 2, 3] et l’inférence démographique [4, 5, 6]. Les modèles testés atteignent des BAcc comprises entre 0.770 et 0.933 sur données propres, GradientBoosting et SVM étant les plus performants avec une BAcc de 0.933.

Deuxièmement, ce travail mesure l’impact de différents types de perturbations comportementales sur les performances des modèles. Sans aucune défense, la BAcc peut chuter jusqu’à 0.477 pour GradientBoosting face aux attaques les plus sévères, contre 0.933 sur données propres. Les perturbations qui simulent des variations réalistes de posture, de synchronisation des mains ou de dynamique gestuelle affectent fortement les prédictions, révélant une vulnérabilité claire des modèles face à des modifications réalistes du signal.

Troisièmement, dix-neuf défenses fixes ont été évaluées sur deux lots successifs. Le premier lot montre que l’Adversarial Training est la meilleure défense avec un gain moyen de +0.066, mais son effet varie fortement selon le modèle. Le second lot révèle que le Bootstrap (+0.043) et le Bagging (+0.037) sont les plus efficaces, le Bootstrap étant le seul à améliorer tous les modèles en même temps. À l’opposé, les méthodes de détection

et rejet comme le LOF, l'OC-SVM ou la distance de Mahalanobis rejettent presque tous les exemples et sont inutilisables en pratique. Les transformations comme le Log Transform et le Median Filter dégradent systématiquement les performances (-0.196 en moyenne). Ces résultats montrent qu'aucune défense unique ne fonctionne bien dans tous les contextes.

Quatrièmement, des combinaisons de défenses empilées en séquence ont été évaluées afin de dépasser les limites des défenses appliquées séparément. Le meilleur empilement **Wins+Zscore+PCA+Ens** qui applique successivement la Winsorization, le Z-score et la reconstruction PCA avant de voter entre tous les modèles atteint un gain moyen de $+0.057$, supérieur au Bootstrap seul ($+0.043$). En moyenne sur toutes les attaques, le SVM passe de 0.682 à 0.854 et XGBoost de 0.714 à 0.860 avec cet empilement. Ces résultats montrent que combiner plusieurs défenses en séquence améliore la robustesse, mais aucun modèle ne retrouve entièrement sa BAcc d'origine, ce qui confirme la nécessité d'une approche adaptative.

Cinquièmement, ce constat a conduit au développement d'un système adaptatif capable de détecter automatiquement le type d'attaque et de choisir la défense la plus adaptée selon le modèle ciblé. Il obtient le meilleur gain moyen ($+0.081$) et détecte 76.3% des attaques ; il est donc retenu comme solution finale.

Enfin, au-delà des résultats quantitatifs, ce mémoire propose un cadre expérimental unifié qui distingue les attaques numériques classiques et les perturbations comportementales réalistes adaptées aux données biomécaniques en réalité virtuelle. Ce cadre constitue une base reproductible pour de futurs travaux sur la robustesse des systèmes d'analyse du mouvement en environnement immersif.

1.4 Organisation du mémoire

Ce mémoire est structuré en huit chapitres.

Le chapitre 1 introduit le contexte général du travail, la problématique étudiée, les questions de recherche ainsi que l'approche proposée.

Le chapitre 2 expose les fondements théoriques et méthodologiques nécessaires à la compréhension du travail, notamment les notions liées à l'apprentissage automatique, à la modélisation temporelle et aux mécanismes de robustesse.

Le chapitre 3 présente l'état de l'art relatif à la biométrie comportementale en réalité virtuelle, aux méthodes de classification du genre et aux travaux existants sur les menaces et la robustesse des modèles d'apprentissage automatique.

Le chapitre 4 décrit les jeux de données utilisés, en particulier les bases FAST et Alyx, ainsi que la méthodologie expérimentale adoptée pour la préparation, le prétraitement et l'évaluation des données.

Le chapitre 5 est consacré à la reconnaissance du genre à partir des données de mouvement en réalité virtuelle. Il présente les modèles étudiés et analyse leurs performances en conditions nominales.

Le chapitre 6 examine les menaces et perturbations affectant la reconnaissance du genre en réalité virtuelle. Il analyse l'impact des différentes perturbations considérées sur la stabilité et la fiabilité des modèles.

Le chapitre 7 présente les mécanismes de défense évalués face aux attaques comportementales identifiées, en analysant leur efficacité à préserver la robustesse des modèles de reconnaissance de genre en contexte RV.

Le chapitre 8 conclut ce mémoire en synthétisant les principaux enseignements du travail, puis présente les perspectives de recherche et les travaux à venir.

Chapitre 2

Fondements théoriques et méthodologiques

Ce chapitre présente les fondements théoriques et méthodologiques nécessaires à l'étude des modèles d'apprentissage automatique et à l'analyse de leur robustesse dans un contexte perturbé. Il vise à établir un cadre conceptuel commun permettant de comprendre les principes de modélisation, les hypothèses sous-jacentes et les outils généralement mobilisés dans les travaux en apprentissage automatique.

Les grandes familles de modèles abordées dans ce chapitre couvrent les approches fondées sur l'extraction de caractéristiques et les modèles tabulaires, les modèles neuronaux profonds, ainsi que les modèles temporels conçus pour exploiter la dynamique des données. Les principes généraux de l'évaluation des modèles et de l'analyse de leurs performances sont également introduits.

Enfin, ce chapitre présente les concepts fondamentaux liés aux perturbations comportementales et à l'étude de la robustesse des modèles d'apprentissage automatique. Ces notions sont exposées de manière générale, indépendamment de tout contexte applicatif spécifique, et constituent la base théorique des analyses développées dans les chapitres suivants.

2.1 Fondements de la réalité virtuelle et des signaux de mouvement

La réalité virtuelle repose sur des dispositifs immersifs permettant de suivre avec précision les mouvements de l'utilisateur dans un environnement numérique. Les systèmes actuels combinent généralement des casques de réalité virtuelle (Head-Mounted Displays, HMD), des contrôleurs manuels et divers capteurs de position et d'orientation, tels que des unités de mesure inertielle (Inertial Measurement Units, IMU), des capteurs optiques ou des systèmes de suivi hybrides [22, 23]. Ces dispositifs permettent de capturer en continu les mouvements de la tête, des mains et, dans certains cas, du corps entier, à des fréquences élevées, allant généralement de 30 à plus de 120 Hz.

Les signaux collectés dans les environnements de RV sont principalement de nature spatio-temporelle. Ils incluent des positions tridimensionnelles, des vitesses et des accélérations linéaires, ainsi que des rotations et orientations, souvent représentées sous forme d'angles d'Euler ou de quaternions. Combinées dans le temps, ces mesures permettent de reconstruire des gestes, des trajectoires et des schémas de mouvement complexes, reflétant le comportement moteur de l'utilisateur dans l'environnement immersif [23, 24].

Malgré leur richesse informationnelle, les signaux de mouvement en réalité virtuelle présentent plusieurs limitations inhérentes aux dispositifs de suivi. Ils peuvent notamment être affectés par un bruit de capteur dû aux imprécisions matérielles, aux occlusions, ou aux erreurs de synchronisation entre capteurs, pouvant introduire des variations artificielles dans les mesures [15, 16]. Des dérives progressives peuvent également apparaître au cours du temps, altérant la précision des trajectoires reconstruites. Enfin, les contraintes matérielles liées au coût, à l'autonomie et à l'ergonomie des dispositifs limitent parfois la qualité ou la stabilité des signaux collectés, ce qui complexifie leur exploitation dans des systèmes d'apprentissage automatique.

2.2 Caractéristiques et défis des données RV

Les données de mouvement issues de la réalité virtuelle se distinguent par de fortes dépendances temporelles. Les mouvements humains étant continus par nature, les observations successives ne peuvent être considérées comme indépendantes, et leur modélisation nécessite de prendre en compte la dynamique temporelle des signaux [25].

Cette propriété rend inadéquates de nombreuses approches classiques qui supposent des échantillons indépendants et identiquement distribués.

Un autre défi majeur réside dans la variabilité inter-utilisateur. Chaque individu développe des schémas moteurs spécifiques, influencés par des facteurs physiologiques, cognitifs et culturels, ce qui conduit à une forte hétérogénéité des comportements observés [26]. Si cette variabilité constitue un atout dans un contexte biométrique, elle peut toutefois compliquer la généralisation des modèles visant à inférer des attributs sensibles, tels que le genre ou l'âge, notamment lorsque le nombre d'utilisateurs disponibles pour l'apprentissage est limité.

Les données de RV présentent également une hétérogénéité inter-sessions significative. Les comportements d'un même utilisateur peuvent varier selon le contexte expérimental, la durée d'utilisation, la fatigue ou la familiarité avec le système, entraînant des changements de distribution entre les sessions d'enregistrement [27]. Cette variabilité temporelle à long terme peut dégrader les performances des modèles entraînés dans des conditions spécifiques.

Par ailleurs, la combinaison de multiples capteurs, de degrés de liberté et de fréquences d'échantillonnage conduit à une forte dimensionnalité des données. Cette propriété augmente le risque de surapprentissage et impose des contraintes importantes en termes de complexité computationnelle et de sélection de caractéristiques [28]. Enfin, l'ensemble de ces facteurs contribue à des problèmes de généralisation, les modèles pouvant présenter des performances élevées sur des données connues tout en échouant à s'adapter à de nouveaux utilisateurs, de nouvelles sessions ou des conditions légèrement différentes.

Ces défis soulignent la complexité intrinsèque des signaux de mouvement en réalité virtuelle et motivent l'étude approfondie des modèles d'apprentissage automatique ainsi que de leur robustesse face aux variations et perturbations des données.

2.3 Fondements de l'apprentissage automatique pour les signaux de réalité virtuelle

Les signaux de mouvement issus des environnements de réalité virtuelle constituent des données spatio-temporelles complexes dont l'analyse requiert des approches d'apprentissage automatique adaptées. Les contraintes inhérentes à ces signaux, no-

tamment le bruit de mesure, la dépendance temporelle des observations, la variabilité inter-individuelle et la forte dimensionnalité des données, ont été largement documentées dans la littérature et motivent le recours à des stratégies de modélisation spécifiques [15, 16, 26].

Dans ce contexte, le traitement des signaux RV en apprentissage automatique s'est structuré autour de trois grandes familles d'approches. Les modèles tabulaires reposent sur l'extraction explicite de caractéristiques à partir des signaux bruts. Les modèles neuronaux profonds non séquentiels, comme les MLP, sont appliqués aux mêmes représentations vectorielles que les modèles tabulaires, mais sont capables d'apprendre des frontières de décision non linéaires plus complexes à partir de ces caractéristiques. Enfin, les modèles temporels visent à capturer explicitement les dépendances dynamiques des signaux, en particulier dans des contextes où la continuité du mouvement joue un rôle central [25].

L'évaluation et la comparaison de ces approches s'appuient largement sur des jeux de données publics de référence en réalité virtuelle, parmi lesquels les ensembles **FAST** [29, 30] et **Who Is Alyx?** [1, 31] occupent une place centrale. Ces jeux de données ont notamment permis de conduire des analyses d'identifiabilité à grande échelle, mettant en évidence le caractère distinctif des signaux de mouvement en réalité virtuelle [2], et ont été utilisés dans quelques travaux récents consacrés à la classification du genre en environnement immersif [5, 32].

2.3.1 Modèles d'apprentissage automatique classiques

Les modèles tabulaires reposent sur une étape préalable de transformation des signaux bruts en vecteurs de caractéristiques statiques, généralement obtenus par des descripteurs statistiques (moyenne, variance, maximum, minimum, corrélations, mesures fréquentielles). Cette représentation permet d'appliquer des algorithmes d'apprentissage supervisé classiques, parmi lesquels les modèles d'ensemble occupent une place centrale.

Les *Random Forests* [33] exploitent un ensemble d'arbres de décision entraînés sur des sous-échantillons aléatoires des données et des variables, ce qui leur confère une bonne robustesse au bruit et une capacité de généralisation satisfaisante. Les méthodes de *Gradient Boosting* [34], ainsi que leurs implémentations optimisées telles que XGBoost [35] et LightGBM [36], construisent des ensembles séquentiels de modèles faibles afin de mi-

nimiser progressivement l'erreur de prédiction, offrant d'excellentes performances sur des données structurées de grande dimension.

En complément, les machines à vecteurs de support (*Support Vector Machines*, SVM) à noyau radial (*Radial Basis Function*, RBF) sont considérées afin de modéliser des frontières de décision non linéaires complexes. Le noyau RBF mesure la similarité entre les échantillons en fonction de leur distance dans l'espace des caractéristiques, ce qui permet de capturer des relations locales et non linéaires entre les variables d'entrée, particulièrement pertinentes pour l'analyse de signaux biomécaniques et comportementaux bruités ¹. Dans le contexte de la réalité virtuelle, ces modèles ont été largement utilisés pour modéliser des comportements utilisateurs à partir de signaux de mouvement, notamment pour des tâches d'identification ou de classification du genre [5, 37].

Ainsi, les modèles tabulaires se distinguent notamment par leur capacité à traiter efficacement des représentations de signaux et par leur rapidité d'entraînement. En revanche, ils ne capturent pas explicitement la dynamique temporelle des signaux, celle-ci étant indirectement encodée par les caractéristiques extraites, ce qui peut limiter leur expressivité face à des comportements complexes ou évolutifs.

Cette limitation est particulièrement marquée dans les environnements de réalité virtuelle, où la dynamique temporelle fine des mouvements joue un rôle central dans la discrimination des comportements utilisateurs.

2.3.2 Modèles neuronaux profonds

En complément des modèles d'apprentissage automatique classiques, des modèles neuronaux profonds non séquentiels, en particulier les perceptrons multicouches (MLP), peuvent également être appliqués aux vecteurs de caractéristiques tabulaires décrits dans la section précédente. Comme pour les modèles classiques, les signaux bruts sont d'abord transformés en représentations vectorielles statiques à l'aide de descripteurs statistiques.

Les MLP sont capables d'apprendre des frontières de décision non linéaires complexes à partir de ces représentations vectorielles d'entrée, sans nécessiter une extraction manuelle supplémentaire des caractéristiques [28].

Appliqués aux données de mouvement en RV, ces modèles permettent de capturer des relations complexes entre les variables biomécaniques et les classes cibles.

1. <https://github.com/scikit-learn/scikit-learn/tree/main/examples/svm>

2.3.3 Modèles temporels

Afin de tirer parti de la structure séquentielle des signaux RV, de nombreuses approches reposent sur des modèles temporels capables d'exploiter explicitement les dépendances entre observations successives. Les réseaux de neurones récurrents, et en particulier les architectures LSTM et GRU [25, 38], ont été largement adoptés pour modéliser l'évolution temporelle continue des trajectoires et des gestes utilisateurs. Ces modèles sont capables de capturer des dépendances à long terme, mais leur entraînement peut être coûteux et sensible au bruit.

En parallèle, les réseaux convolutionnels 1D ont été introduits pour extraire des motifs locaux dans le temps, offrant une alternative plus stable et parallélisable [39, 40]. Ces architectures permettent de détecter des patterns temporels discriminants tout en réduisant la complexité computationnelle par rapport aux réseaux récurrents.

Plus récemment, des architectures basées sur les Transformers ont été proposées pour l'analyse des séries temporelles, avec pour objectif de mieux modéliser les relations à long terme et les variations multi-échelles. Des modèles tels que Informer [41] ou TimesNet [42, 43] introduisent des mécanismes spécifiques pour capturer les structures temporelles complexes, et ont été évalués dans des contextes récents d'identification et de reconnaissance comportementale en réalité virtuelle [27, 32]. Cependant, ces architectures requièrent généralement de grandes quantités de données d'entraînement pour atteindre leur plein potentiel, ce qui peut constituer une limitation dans des contextes comme la réalité virtuelle où les jeux de données disponibles restent de taille modeste.

Dans le contexte de la réalité virtuelle, ces architectures permettent de modéliser plus fidèlement la continuité et la structure séquentielle des mouvements utilisateurs, ce qui explique leur adoption croissante pour des tâches d'identification et de classification du genre.

Malgré leurs performances prometteuses, ces modèles temporels présentent une sensibilité accrue aux perturbations des signaux d'entrée, ce qui soulève des questions essentielles quant à leur robustesse face aux perturbations. Cette observation constitue une motivation centrale pour l'analyse approfondie menée dans ce mémoire.

2.4 Attaques adversariales : principes généraux

Les attaques adversariales désignent un ensemble de techniques visant à tromper un modèle d'apprentissage automatique en introduisant des perturbations contrôlées dans les données d'entrée afin de provoquer une erreur de prédiction, tout en maintenant une confiance élevée du modèle dans cette prédiction erronée [7]. Cette dernière caractéristique est ce qui distingue fondamentalement une attaque adversariale d'une simple donnée hors distribution (OOD) : alors que n'importe quel type de bruit peut provoquer un décalage entre la distribution des données observées et celle de l'ensemble d'apprentissage, entraînant une erreur de classification, une attaque adversariale se caractérise par le fait que la perturbation est spécifiquement conçue pour exploiter les faiblesses du modèle, de sorte que celui-ci demeure confiant dans sa prédiction erronée plutôt que de manifester une incertitude qui trahirait la présence d'une anomalie. Ce phénomène illustre la vulnérabilité structurelle de nombreux modèles d'apprentissage profond, même lorsque ceux-ci présentent des performances élevées dans des conditions d'utilisation normales.

Les premières études sur les attaques adversariales ont été réalisées dans le domaine de la vision par ordinateur, où il a été démontré que de légères modifications des pixels d'une image peuvent entraîner des erreurs de classification importantes, l'image restant pourtant visuellement identique à l'œil nu [8, 44]. Ces travaux ont révélé que les réseaux de neurones profonds peuvent apprendre des frontières de décision sensibles à de petites variations dans l'espace des caractéristiques. En conséquence, un attaquant capable de modifier légèrement les données d'entrée peut influencer le comportement du modèle et provoquer des prédictions incorrectes.

Au-delà du domaine de la vision, la recherche s'est progressivement étendue à d'autres types de données, notamment les séries temporelles et les signaux sensoriels. Contrairement aux images, les données temporelles présentent des dépendances séquentielles fortes, ce qui impose des contraintes supplémentaires sur la construction des perturbations comportementales. Les attaques doivent en effet préserver une certaine cohérence temporelle afin de produire des signaux cohérents. Plusieurs travaux ont montré que les modèles de classification de séries temporelles restent vulnérables à des perturbations modifiant légèrement les motifs dynamiques ou les tendances locales du signal [45]. Dans les systèmes reposant sur des capteurs corporels ou des dispositifs immersifs, les perturbations comportementales peuvent également prendre la forme de modifications

ciblées des signaux mesurés par les capteurs. Des études ont montré que la perturbation partielle de certains capteurs ou la modification de leurs mesures peut suffire à induire des erreurs de prédiction dans des modèles de reconnaissance d'activité humaine ou d'analyse comportementale [46]. Ces attaques peuvent correspondre, par exemple, à l'injection de bruit dans les mesures, à des altérations de l'amplitude du signal ou à des perturbations affectant seulement une partie des capteurs utilisés par le système.

Certaines attaques adversariales exploitent directement les propriétés mathématiques des modèles d'apprentissage afin de calculer automatiquement les perturbations à appliquer aux données d'entrée. En particulier, les méthodes basées sur le gradient utilisent la dérivée de la fonction de perte du modèle pour déterminer dans quelle direction modifier les caractéristiques d'entrée afin de provoquer une erreur de classification. Ces techniques sont principalement appliquées aux modèles différentiables, tels que les réseaux neuronaux profonds, et constituent une approche couramment utilisée pour analyser la robustesse des systèmes d'apprentissage automatique face aux perturbations comportementales. Toutes les attaques adversariales ne reposent cependant pas sur l'utilisation de gradients, dans certains contextes, notamment celui des données comportementales issues de la réalité virtuelle, un attaquant peut chercher à modifier directement certaines caractéristiques du comportement moteur posture, dynamique gestuelle, mouvements oculaires utilisées par un système de classification afin d'échapper à la détection. Par exemple, dans les systèmes de filtrage anti-spam, un expéditeur peut modifier légèrement la forme de certains messages ou caractéristiques textuelles afin d'éviter leur détection par les règles de classification. Ces stratégies correspondent à des modifications ciblées des caractéristiques d'entrée et s'inscrivent dans le cadre de la classification stratégique (*strategic classification*), où les individus adaptent leurs caractéristiques afin d'influencer la décision d'un système d'apprentissage automatique [47, 48].

2.5 Fondements des mécanismes de défense

Face aux perturbations comportementales susceptibles de tromper les modèles de classification du genre en réalité virtuelle, plusieurs familles de mécanismes de défense ont été proposées dans la littérature. Ces approches visent à améliorer la robustesse des modèles en agissant soit sur les données d'entrée avant classification, soit sur le processus d'apprentissage, soit sur la structure du classifieur lui-même.

Les 19 défenses évaluées dans ce travail sont organisées en quatre catégories selon leur principe d'action : **(i)** les transformations des données, qui filtrent ou normalisent les exemples adversariaux avant de les soumettre au classifieur ; **(ii)** les mécanismes de détection et rejet, qui identifient et écartent les exemples suspects ; **(iii)** les méthodes basées sur le vote, qui agrègent les prédictions de plusieurs modèles pour corriger les erreurs induites par les attaques ; et **(iv)** les approches d'entraînement robuste, qui modifient le processus d'optimisation afin de rendre les modèles intrinsèquement plus résistants aux perturbations.

2.5.1 Catégorie 1 : Transformation des données

Les défenses par transformation modifient l'exemple $\mathbf{x}_{adv}$ avant de le soumettre au classifieur f , dans le but d'éliminer ou d'atténuer les perturbations comportementales [49, 50]. Soit $\mathbf{x}_{adv} \in \mathbb{R}^d$ l'exemple perturbé, l'objectif est de construire $\tilde{\mathbf{x}}$ tel que $f(\tilde{\mathbf{x}}) \approx f(\mathbf{x}_{clean})$.

L'idée fondamentale est de supprimer ou d'atténuer les perturbations présentes dans les features en appliquant des opérations de prétraitement robustes.

2.5.1.1 Winsorization

La Winsorization consiste à écrêter les valeurs extrêmes aux percentiles p et $(1 - p)$ calculés sur les données d'entraînement $\mathcal{D}_{train}$ [51, 52, 53]. Cette technique vise principalement à neutraliser les attaques qui produisent des valeurs aberrantes, comme *unit mismatch* qui multiplie les features de position par 10^3 . La transformation s'écrit :

$$\tilde{x}_j = \text{clip} \left(x_j, q_p^{(j)}, q_{1-p}^{(j)} \right) \quad (2.1)$$

où $q_\alpha^{(j)}$ désigne le quantile d'ordre α de la j -ième feature sur $\mathcal{D}_{train}$, et $\text{clip}(a, b, c) = \max(b, \min(a, c))$. Dans nos expériences, nous fixons $p = 0.05$, ce qui correspond aux quantiles 5% et 95%.

2.5.1.2 Z-score clipping

Cette approche normalise chaque feature par rapport à sa distribution dans les données d'entraînement, écrête les valeurs dont le z-score dépasse $\pm\tau$ écarts-types, puis

dénormalise [54, 55, 56]. Les perturbations comportementales fortes produisent des z-scores très élevés (de l'ordre de 10^4 pour *unit mismatch*), qui sont ainsi ramenés dans la plage normale. La transformation se décompose en trois étapes :

$$z_j = \frac{x_j - \mu_j}{\sigma_j + \epsilon} \quad (2.2)$$

$$\tilde{z}_j = \text{clip}(z_j, -\tau, +\tau) \quad (2.3)$$

$$\tilde{x}_j = \tilde{z}_j \cdot \sigma_j + \mu_j \quad (2.4)$$

où μ_j et σ_j sont la moyenne et l'écart-type de la j -ième feature sur $\mathcal{D}_{train}$, $\epsilon = 10^{-9}$ évite la division par zéro, et $\tau = 3$ dans nos expériences.

2.5.1.3 Log transform

Cette défense applique une transformation logarithmique sur l'ensemble des features en préservant le signe original, comprimant les grandes valeurs produites par certaines attaques [57, 58] :

$$\tilde{x}_j = \text{sign}(x_j) \cdot \log(1 + |x_j|) \quad (2.5)$$

Pour une valeur $x = 1000$, on obtient $\log(1001) \approx 6.9$. Cependant, cette transformation modifie également les features non perturbées, perturbant les distributions apprises lors de l'entraînement.

2.5.1.4 Feature normalization

La normalisation des features ramène chaque feature dans la plage $[\min_j, \max_j]$ observée dans les données d'entraînement [59, 60]. Ces bornes sont calculées sur $\mathcal{D}_{train}$ afin de représenter la plage de variation naturelle de chaque feature pour des comportements VR normaux, sans biais d'évaluation. La transformation se décompose en trois étapes :

Étape 1 - Écrêtage : Les valeurs hors de la plage d'entraînement sont ramenées aux bornes observées. Sans cette étape, une valeur comme $x_j = 1000$ produirait après normalisation une valeur hors de $[0, 1]$, rendant la dénormalisation inefficace :

$$x_j^{clip} = \text{clip}\left(x_j, \min_j, \max_j\right) \quad (2.6)$$

Étape 2 - Normalisation : Les valeurs écrêtées sont projetées dans $[0, 1]$. La normalisation min-max est préférée au z-score car elle garantit que les valeurs restent strictement dans la plage du train après dénormalisation :

$$x_j^{norm} = \frac{x_j^{clip} - \min_j}{\max_j - \min_j + \epsilon} \quad (2.7)$$

où $\epsilon = 10^{-9}$ évite la division par zéro pour les features constantes ($\max_j = \min_j$).

Étape 3 - Dénormalisation : Les valeurs sont ramenées dans l'espace original :

$$\tilde{x}_j = x_j^{norm} \cdot (\max_j - \min_j) + \min_j \quad (2.8)$$

Pour $\epsilon \rightarrow 0$, la transformation complète se simplifie à $\tilde{x}_j \approx \text{clip}(x_j, \min_j, \max_j)$, montrant que cette défense agit essentiellement comme un écrêtage dans la plage naturelle des données d'entraînement.

2.5.1.5 Feature squeezing

Inspirée des travaux sur les défenses adversariales pour les réseaux de neurones [61, 62], cette technique réduit la précision de représentation des features par quantification [63, 64], éliminant les micro-perturbations de faible amplitude. Chaque feature est arrondie au multiple le plus proche d'un pas de quantification δ :

$$\tilde{x}_j = \delta \cdot \text{round}\left(\frac{x_j}{\delta}\right), \quad \delta = \frac{1}{2^b} \quad (2.9)$$

Dans nos expériences, $b = 8$, soit $\delta = \frac{1}{256} \approx 0.004$. Ce pas de quantification est adapté aux features normalisées dans $[0, 1]$, produisant $2^b = 256$ niveaux de précision distincts.

2.5.1.6 PCA reconstruction

Cette défense projette l'exemple sur le sous-espace des k composantes principales extraites des données d'entraînement, [65] puis reconstruit dans l'espace original. L'hypothèse sous-jacente est que les données normales résident dans ce sous-espace de faible dimension, tandis que les perturbations comportementales résident dans les directions résiduelles. Soit $\mathbf{V}_k \in \mathbb{R}^{d \times k}$ la matrice des k vecteurs propres principaux :

$$\tilde{\mathbf{x}} = \mathbf{V}_k \mathbf{V}_k^\top (\mathbf{x}_{adv} - \boldsymbol{\mu}) + \boldsymbol{\mu} \quad (2.10)$$

où μ est la moyenne des données d'entraînement. Dans nos expériences, $k = 50$.

2.5.1.7 Median filtering

Cette approche remplace chaque feature par la médiane des valeurs observées chez les k plus proches voisins dans les données d'entraînement [66, 67, 68, 69]. Soit $\mathcal{N}_k(\mathbf{x}_{adv})$ l'ensemble des k plus proches voisins de $\mathbf{x}_{adv}$ dans $\mathcal{D}_{train}$ selon la distance euclidienne :

$$\mathcal{N}_k(\mathbf{x}_{adv}) = \arg \min_{\mathcal{S} \subseteq \mathcal{D}_{train}, |\mathcal{S}|=k} \sum_{\mathbf{x}^{(i)} \in \mathcal{S}} \|\mathbf{x}^{(i)} - \mathbf{x}_{adv}\|_2 \quad (2.11)$$

La feature reconstruite est alors :

$$\tilde{x}_j = \text{median} \left(\left\{ x_j^{(i)} : \mathbf{x}^{(i)} \in \mathcal{N}_k(\mathbf{x}_{adv}) \right\} \right) \quad (2.12)$$

Dans nos expériences, $k = 5$.

2.5.2 Catégorie 2 : Détection et rejet

Les défenses par détection cherchent à identifier les exemples adversariaux avant classification [70, 71, 72]. Lorsqu'une anomalie est détectée, le système s'abstient de prédire plutôt que de risquer une prédiction erronée. Formellement, soit $D : \mathbb{R}^d \rightarrow \{0, 1\}$ un détecteur binaire :

$$\hat{y} = \begin{cases} f(\mathbf{x}) & \text{si } D(\mathbf{x}) = 0 \quad (\text{normal}) \\ \perp & \text{si } D(\mathbf{x}) = 1 \quad (\text{anomalie}) \end{cases} \quad (2.13)$$

où $\perp$ désigne l'abstention.

Le principe fondamental de cette famille est qu'une défense universelle ne peut pas être optimale pour toutes les classes de perturbations : une perturbation morphologique nécessite une défense différente d'une perturbation dynamique ou temporelle. Ces approches s'inspirent des systèmes de détection d'intrusion et des méthodes de classification robuste multi-modèles [73, 74].

2.5.2.1 Local outlier factor (LOF)

Le LOF compare la densité locale de chaque point avec celle de ses k plus proches voisins [75, 76, 77, 78]. Un point dont la densité locale est significativement inférieure à celle de ses voisins est considéré comme une anomalie. La distance d'accessibilité entre deux points $\mathbf{x}$ et $\mathbf{o}$ est définie par :

$$\text{reach-dist}_k(\mathbf{x}, \mathbf{o}) = \max(d_k(\mathbf{o}), d(\mathbf{x}, \mathbf{o})) \quad (2.14)$$

où $d(\mathbf{x}, \mathbf{o}) = \|\mathbf{x} - \mathbf{o}\|_2$ est la distance euclidienne et $d_k(\mathbf{o})$ est la distance de $\mathbf{o}$ à son k -ième plus proche voisin. La densité locale d'accessibilité est alors :

$$\text{lrd}_k(\mathbf{x}) = \frac{k}{\sum_{\mathbf{o} \in \mathcal{N}_k(\mathbf{x})} \text{reach-dist}_k(\mathbf{x}, \mathbf{o})} \quad (2.15)$$

Le score LOF final est le rapport entre la densité moyenne des voisins et la densité locale de $\mathbf{x}$:

$$\text{LOF}_k(\mathbf{x}) = \frac{1}{k} \sum_{\mathbf{o} \in \mathcal{N}_k(\mathbf{x})} \frac{\text{lrd}_k(\mathbf{o})}{\text{lrd}_k(\mathbf{x})} \quad (2.16)$$

Un score $\text{LOF}_k(\mathbf{x}) \gg 1$ indique que $\mathbf{x}$ est dans une zone moins dense que ses voisins, signalant une anomalie potentielle. Dans nos expériences, $k = 20$ et le taux de contamination est fixé à 10%.

2.5.2.2 One-Class SVM

Le One-Class SVM apprend un hyperplan séparant les données d'entraînement de l'origine dans un espace de features induit par un noyau RBF [79, 80, 81]. Le problème d'optimisation s'écrit :

$$\min_{\mathbf{w}, \rho, \xi} \frac{1}{2} \|\mathbf{w}\|^2 - \rho + \frac{1}{\nu n} \sum_{i=1}^n \xi_i \quad (2.17)$$

$$\text{s.t. } \mathbf{w} \cdot \phi(\mathbf{x}_i) \geq \rho - \xi_i, \quad \xi_i \geq 0 \quad \forall i \quad (2.18)$$

où $\mathbf{w}$ est le vecteur normal à l'hyperplan, ρ est le biais, $\nu \in (0, 1]$ contrôle le taux d'anomalies attendu, et ϕ est la fonction de transformation induite par le noyau RBF. Un exemple $\mathbf{x}$ est classé comme normal si $\mathbf{w} \cdot \phi(\mathbf{x}) \geq \rho$. Dans nos expériences, $\nu = 0.1$.

2.5.2.3 Distance de mahalanobis

Cette méthode mesure la distance entre un exemple et la distribution des données d'entraînement en tenant compte des corrélations entre features grâce à la matrice de covariance inverse [82, 83, 84] :

$$d_M(\mathbf{x}) = \sqrt{(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})} \quad (2.19)$$

où $\boldsymbol{\mu}$ et $\boldsymbol{\Sigma}$ sont la moyenne et la matrice de covariance des données d'entraînement. Le seuil de détection τ_M est fixé au 95^e percentile des distances calculées sur $\mathcal{D}_{train}$.

2.5.2.4 Test statistique de Kolmogorov-Smirnov

Ce test compare chaque feature de l'exemple entrant avec la distribution gaussienne ajustée sur les données d'entraînement [85, 86]. Pour la j -ième feature, on ajuste une loi normale $\mathcal{N}(\mu_j, \sigma_j^2)$ sur $\mathcal{D}_{train}$, puis on calcule la statistique KS entre la valeur observée x_j et cette distribution de référence :

$$D_{KS}^{(j)} = \sup_t \mathbf{1}[x_j \leq t] - F_0^{(j)}(t) \quad (2.20)$$

où $F_0^{(j)}(t) = \Phi\left(\frac{t - \mu_j}{\sigma_j}\right)$ est la fonction de répartition gaussienne ajustée sur $\mathcal{D}_{train}$, et Φ est la fonction de répartition de la loi normale standard. L'abstention est déclenchée si plus de 20% des features présentent une p-valeur inférieure à 0.01 :

$$D(\mathbf{x}) = \mathbf{1} \left[\frac{1}{d} \sum_{j=1}^d \mathbf{1}[p_j < 0.01] > 0.20 \right] \quad (2.21)$$

où p_j est la p-valeur associée à $D_{KS}^{(j)}$.

2.5.3 Catégorie 3 : Méthodes basées sur le vote

Les défenses par vote combinent les prédictions de plusieurs modèles ou versions d'un même modèle afin de corriger les erreurs induites par les attaques [87, 88]. Soit

$\{f_1, \dots, f_m\}$ un ensemble de m classifieurs, la prédiction finale par vote majoritaire est :

$$\hat{y} = \arg \max_{c \in \mathcal{Y}} \sum_{i=1}^m \mathbf{1}[f_i(\mathbf{x}) = c] \quad (2.22)$$

Cette famille exploite les propriétés de robustesse naturelle des méthodes ensemblistes. Le *bagging* (*Bootstrap Aggregating*) entraîne plusieurs classifieurs sur des sous-ensembles bootstrap du jeu d'entraînement et agrège leurs prédictions, réduisant ainsi la variance et la sensibilité aux perturbations locales [89]. Des travaux récents ont montré que les méthodes ensemblistes offrent une résistance naturelle à certaines classes de perturbations, notamment celles qui agissent de manière ciblée sur un sous-ensemble de caractéristiques [90].

2.5.3.1 Ensemble de modèles

Cette approche soumet chaque exemple aux six classifieurs (RandomForest, GradientBoosting, XGBoost, LightGBM, SVM, MLP) et retient la classe majoritaire selon l'équation (2.22) avec $m = 6$. Une attaque optimisée contre un modèle spécifique ne trompe pas nécessairement les autres modèles [91, 92], dont les frontières de décision diffèrent structurellement.

2.5.3.2 Bootstrap ensemble

Dans nos expériences, $m = 10$ copies du modèle original sont réentraînées, chacune sur un échantillon bootstrap $\mathcal{D}^{(i)}$ tiré avec remise depuis $\mathcal{D}_{train}$ [93, 94]. Chaque échantillon est construit en tirant n indices aléatoirement avec remise :

$$\mathcal{D}^{(i)} = \left\{ \left(\mathbf{x}_{s_j^{(i)}}, y_{s_j^{(i)}} \right) \right\}_{j=1}^n, \quad s_j^{(i)} \stackrel{\text{i.i.d.}}{\sim} \text{Uniform}\{1, \dots, n\} \quad (2.23)$$

où $s_j^{(i)}$ est le j -ième indice tiré pour le i -ième bootstrap. La prédiction finale est obtenue par vote majoritaire selon l'équation (2.22).

2.5.3.3 Bagging defense

Dix modèles RandomForest simples (50 arbres) sont entraînés sur des sous-ensembles aléatoires représentant $r = 70\%$ des données d'entraînement tirés avec remise [95, 96].

Pour chaque modèle $i \in \{1, \dots, m\}$, un sous-ensemble $\mathcal{D}^{(i)}$ de taille $\lfloor r \cdot n \rfloor$ est tiré avec remise depuis $\mathcal{D}_{train}$:

$$\mathcal{D}^{(i)} = \left\{ \left(\mathbf{x}_{s_j^{(i)}}, y_{s_j^{(i)}} \right) \right\}_{j=1}^{\lfloor r \cdot n \rfloor}, \quad s_j^{(i)} \stackrel{\text{i.i.d.}}{\sim} \text{Uniform}\{1, \dots, n\} \quad (2.24)$$

La prédiction finale est obtenue par vote majoritaire selon l'équation (2.22) avec $m = 10$. Contrairement au Bootstrap Ensemble qui réentraîne le modèle original, le Bagging entraîne des modèles indépendants de complexité réduite, accentuant la diversité des prédictions.

2.5.3.4 Feature subspace voting

Pour chaque itération $t \in \{1, \dots, T\}$, $\lfloor s \cdot d \rfloor$ features sont sélectionnées aléatoirement et les autres sont masquées [97] :

$$\tilde{x}_j^{(t)} = \begin{cases} x_j & \text{si } j \in \mathcal{I}_t \\ 0 & \text{sinon} \end{cases} \quad (2.25)$$

où $\mathcal{I}_t \subseteq \{1, \dots, d\}$ est un sous-ensemble aléatoire de taille $\lfloor s \cdot d \rfloor$ tiré uniformément sans remise. Dans nos expériences, $T = 25$ et $s = 0.7$.

2.5.3.5 Gaussian smoothing

T prédictions sont effectuées sur des versions bruitées de l'exemple [98, 99, 100, 101], et le vote majoritaire est retenu :

$$\tilde{\mathbf{x}}^{(t)} = \mathbf{x}_{adv} + \boldsymbol{\varepsilon}^{(t)}, \quad \boldsymbol{\varepsilon}^{(t)} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}) \quad (2.26)$$

Dans nos expériences, $T = 25$ et $\sigma = 0.05$.

2.5.4 Catégorie 4 : Entraînement robuste

Les défenses basées sur l'entraînement modifient le processus d'optimisation du modèle afin de le rendre intrinsèquement plus robuste aux perturbations comportementales [102, 103]. L'*adversarial training* constitue l'approche la plus répandue dans cette famille : elle consiste à augmenter l'ensemble d'entraînement avec des exemples pertur-

bés, forçant ainsi le modèle à apprendre des frontières de décision plus robustes aux perturbations [9, 104]. Cette approche a été initialement proposée dans le contexte des réseaux de neurones profonds, mais son principe a été étendu à d'autres familles de classifieurs [105].

2.5.4.1 Adversarial training

Les modèles sont réentraînés sur un dataset augmenté composé des données d'entraînement originales et de $K = 7$ versions perturbées de chaque exemple, une par type d'attaque [106, 107, 108] :

$$\mathcal{D}_{aug} = \mathcal{D}_{train} \cup \bigcup_{k=1}^K \left\{ \left(\mathbf{x}_{adv}^{(i,k)}, y_i \right) \right\}_{i=1}^n \quad (2.27)$$

où $\mathbf{x}_{adv}^{(i,k)} = A_k(\mathbf{x}_i)$ est l'exemple $\mathbf{x}_i$ perturbé par l'attaque A_k . Le dataset augmenté atteint $n \cdot (1 + K) = 8n$ exemples. En exposant le modèle aux perturbations comportementales lors de l'entraînement avec les vrais labels, il apprend à ignorer ces perturbations lors de l'inférence.

2.5.4.2 Label smoothing

Au lieu d'utiliser des labels durs $y \in \{0, 1\}$, les modèles sont entraînés en réduisant la confiance accordée à chaque exemple via une pondération uniforme [109, 110]. Pour chaque exemple, un poids $w_i = 1 - \epsilon$ est appliqué lors de l'entraînement :

$$w_i = 1 - \epsilon, \quad \forall i \in \{1, \dots, n\} \quad (2.28)$$

où $\epsilon = 0.1$ dans nos expériences. Cette pondération réduit l'influence de chaque exemple sur la fonction de perte, rendant le modèle moins confiant dans ses prédictions. À noter que cette technique n'est pas applicable au MLP dans l'implémentation scikit-learn, qui ne supporte pas la pondération des exemples via `sample_weight`.

2.5.4.3 Mixup training

Des exemples synthétiques sont créés en interpolant linéairement des paires d'exemples d'entraînement selon un coefficient $\lambda \sim \text{Beta}(\alpha, \alpha)$, induisant des frontières

de décision plus lisses [111, 112, 113, 114] :

$$\tilde{\mathbf{x}} = \lambda \mathbf{x}_i + (1 - \lambda) \mathbf{x}_j \quad (2.29)$$

$$\tilde{y} = \lambda y_i + (1 - \lambda) y_j \quad (2.30)$$

où $(\mathbf{x}_i, y_i)$ et $(\mathbf{x}_j, y_j)$ sont deux exemples tirés aléatoirement depuis $\mathcal{D}_{train}$. Dans nos expériences, $\alpha = 0.2$ et 100 exemples mixés sont générés et ajoutés à $\mathcal{D}_{train}$.

Chapitre 3

État de l’art

Ce chapitre présente une revue critique des travaux existants portant sur l’analyse des signaux de mouvement en réalité virtuelle et sur la robustesse des modèles d’apprentissage automatique face aux perturbations comportementales. Il examine les principales approches proposées dans la littérature pour l’inférence d’attributs ou l’identification des utilisateurs à partir de données de tracking en réalité virtuelle, ainsi que les méthodes développées pour analyser ou améliorer la robustesse des modèles d’apprentissage face à des perturbations intentionnelles ou non des données d’entrée.

À travers cette synthèse des travaux existants, ce chapitre vise à situer la problématique étudiée dans ce mémoire par rapport aux recherches antérieures, à identifier les limites des approches actuelles et à mettre en évidence les questions encore ouvertes concernant la robustesse des modèles appliqués aux signaux de réalité virtuelle.

3.1 Prédiction du genre à partir de signaux de réalité virtuelle

La prédiction du genre à partir de signaux capturés en réalité virtuelle s’inscrit dans un champ de recherche plus large consacré à l’inférence d’attributs sensibles à partir de données comportementales et physiologiques. De nombreux travaux ont montré que les signaux de mouvement, de posture et de dynamique motrice, ainsi que les données d’eye-tracking, peuvent contenir des informations suffisamment distinctives pour permettre l’identification ou l’inférence d’attributs privés des utilisateurs, dont le genre,

soulevant des enjeux importants en matière de confidentialité et de robustesse des modèles d'apprentissage automatique [115].

Dans ce cadre, plusieurs études se sont intéressées spécifiquement à la classification du genre à partir de données de tracking en réalité virtuelle. Wang et al. proposent une analyse systématique de la prédiction du genre à partir de différentes combinaisons de capteurs (tête, contrôleurs) et de représentations du mouvement, et mettent en évidence une forte dépendance des performances aux signaux exploités ainsi qu'au contexte expérimental [116, 117]. Leurs résultats montrent notamment que les scénarios *cross-domain*, dans lesquels l'entraînement et l'évaluation sont réalisés dans des environnements ou des tâches distincts, restent nettement plus complexes que les scénarios *same-domain*, soulignant les limites de généralisation des modèles actuels. Des travaux complémentaires ont exploré l'apport de l'eye-tracking en réalité virtuelle pour l'inférence du genre, notamment à partir de jeux de données de référence tels que FAST [30] et Who is Alyx [118]. Ces études indiquent que, bien que certaines modalités, telles que la dynamique de la tête ou des descripteurs issus du regard, puissent contribuer à la performance des modèles, celle-ci se dégrade significativement lors de changements de domaine [119]. Ces observations confirment la sensibilité des approches existantes aux biais de contexte et renforcent l'intérêt de travaux visant à évaluer la robustesse des modèles dans des conditions variées.

3.2 Perturbations comportementales et limitations des signaux en RV

Dans les environnements de réalité virtuelle, les données utilisées pour l'analyse du mouvement peuvent être affectées par différentes sources de perturbations liées aux caractéristiques physiques et techniques des dispositifs immersifs. Plusieurs travaux se sont notamment intéressés aux effets de la latence, du jitter ou des pertes de tracking sur les performances des systèmes interactifs et sur la qualité des signaux capturés [120, 15]. Ces phénomènes sont généralement étudiés comme des limitations techniques ou ergonomiques des systèmes de réalité virtuelle.

Parallèlement, plusieurs études ont analysé la variabilité motrice humaine et ses implications pour l'interaction homme-machine en réalité virtuelle [13]. Des phénomènes tels que les tremblements, la fatigue ou les déviations biomécaniques naturelles ont été

identifiés comme des sources majeures de variabilité dans les signaux de mouvement [14], susceptibles d’affecter la fiabilité des modèles de reconnaissance comportementale.

Bien que ces perturbations ne soient pas considérées, dans la littérature, comme des attaques adversariales au sens strict, elles modifient néanmoins les signaux utilisés par les modèles d’analyse comportementale et peuvent ainsi affecter leurs performances. L’étude de l’impact de telles modifications du signal sur les prédictions des classifieurs s’apparente donc à une analyse de sensibilité des modèles face à des perturbations des données d’entrée. À notre connaissance, ces effets n’ont pas été étudiés dans une perspective adversariale appliquée aux données de mouvement en réalité virtuelle.

3.2.1 Attaques adversariales en biométrie comportementale

Des travaux en biométrie comportementale ont montré que des modifications plausibles du comportement humain peuvent suffire à tromper certains systèmes de classification. Par exemple, des attaques comportementales ont été étudiées sur des signaux électrocardiographiques (ECG), où des variations physiologiques ou comportementales peuvent altérer les performances des systèmes de reconnaissance biométrique [121].

Les travaux fondateurs sur la classification adversariale ont ainsi montré que la suppression ou la neutralisation ciblée d’attributs discriminants pouvait induire des erreurs de classification [122], tandis que des perturbations progressives et réalistes des variables d’entrée permettaient de franchir la frontière de décision en contexte *black-box* [123]. D’autres recherches ont mis en évidence la vulnérabilité particulière des modèles à base d’arbres face à des modifications coordonnées de variables corrélées [124], ainsi que l’efficacité de stratégies d’attaque reposant uniquement sur l’observation des sorties du modèle [125].

Plus récemment, ces analyses ont été étendues aux séries temporelles et aux données de mouvement humain, avec des attaques adversariales visant à perturber la décision tout en préservant la structure globale du signal et sa plausibilité [126]. Des attaques *black-box* ont également été proposées pour la reconnaissance d’actions squelettiques, exploitant la coordination spatiale et temporelle des membres [127], ainsi que pour des systèmes fondés sur des capteurs portés, en introduisant du bruit, des dérives ou des pertes partielles de signal [128].

Quelques travaux récents ont commencé à étudier les menaces adversariales dans des contextes proches de la réalité virtuelle, en analysant des attaques ciblant les pipelines

de capture, de transmission ou d'analyse des données immersives [10, 12]. Ces travaux mettent en évidence la nécessité de considérer des attaques combinant des facteurs numériques, physiques et comportementaux.

3.3 Stratégies de défense appliquées aux données de RV

Les principales familles de défenses : entraînement robuste, transformation des données, détection et rejet, et méthodes basées sur le vote ont été présentées au chapitre 2. Ces défenses ont été développées principalement pour des données d'images ou de texte. Leur application aux données de mouvement issues de la réalité virtuelle reste très peu explorée dans la littérature. Quelques travaux ont commencé à proposer des mécanismes pour protéger les données de mouvement en RV. Plutôt que de modifier le modèle de classification, ces approches modifient les données transmises : par exemple, ne transmettre que les rotations des articulations au lieu des positions complètes, ou transmettre les vitesses plutôt que les positions, ce qui réduit significativement la capacité d'un système à identifier un utilisateur [129]. Nair et al. ont aussi montré que des attributs sensibles comme le genre peuvent être inférés à partir des mouvements en RV, et ont appelé à développer des défenses capables de masquer ces attributs dans les données de mouvement, en suggérant notamment l'utilisation de réseaux génératifs adversariaux [117]. Cependant, à notre connaissance, aucune étude n'a évalué de manière systématique des défenses adversariales classiques comme l'entraînement robuste, les transformations des données ou les méthodes de vote appliquées directement à la tâche de classification du genre à partir de signaux de mouvement en RV. Ce travail comble cette lacune en testant dix-neuf défenses fixes couvrant ces quatre catégories, en évaluant des combinaisons de défenses empilées en séquence, et en développant un système adaptatif capable de choisir automatiquement la meilleure défense selon le type d'attaque et le modèle utilisé.

3.4 Synthèse critique et positionnement du mémoire

L'analyse de la littérature présentée dans les sections précédentes met en évidence quatre directions principales de recherche. Premièrement, plusieurs travaux ont montré que les données de mouvement issues de la réalité virtuelle peuvent être exploitées pour

inférer des attributs personnels tels que l'identité ou le genre. Ces études se concentrent principalement sur la conception de modèles performants et sur leur capacité de généralisation.

Deuxièmement, un ensemble de travaux s'est intéressé à l'impact des perturbations techniques des dispositifs immersifs, telles que la latence, le jitter ou les pertes de tracking, sur la qualité des signaux capturés et sur les performances des systèmes d'analyse du mouvement. Ces phénomènes sont généralement étudiés comme des limitations techniques des systèmes de réalité virtuelle, mais ils mettent également en évidence la sensibilité des modèles aux modifications des données d'entrée.

Troisièmement, la littérature en biométrie comportementale et en apprentissage adversarial a montré que des modifications du comportement humain peuvent être utilisées pour tromper des systèmes de classification, notamment sur d'autres types de signaux biologiques tels que les données électrocardiographiques (ECG). Ces travaux illustrent la possibilité d'attaques reposant sur des modifications réalistes des signaux plutôt que sur des perturbations numériques artificielles.

Quatrièmement, la littérature en robustesse des modèles face aux perturbations propose un large éventail de mécanismes de défense visant à protéger les modèles contre les perturbations, couvrant quatre catégories principales : l'entraînement robuste, les transformations des données d'entrée, la détection et le rejet des exemples perturbés, et les méthodes basées sur le vote. Ces défenses sont généralement évaluées sur des données images ou textuelles, et leur efficacité sur des données comportementales spatio-temporelles issues de la réalité virtuelle reste largement inexplorée.

À notre connaissance, aucune étude n'a encore démontré expérimentalement la possibilité de réaliser des perturbations comportementales contre des classifieurs reposant sur des données de mouvement en réalité virtuelle, ni évalué l'efficacité de mécanismes de défense adaptés à ce type de données. Le présent mémoire se positionne précisément à l'intersection de ces quatre directions de recherche, en analysant dans quelle mesure des modifications comportementales des signaux de mouvement peuvent affecter les performances de modèles de classification du genre en réalité virtuelle, et en développant un système adaptatif de détection et de défense spécifiquement conçu pour ce domaine. Ce système, combinant détection par signatures comportementales et sélection dynamique de défenses optimisées par modèle, constitue une contribution originale qui dépasse les approches fixes de la littérature existante en termes de gain de robustesse.

Chapitre 4

Analyse et prétraitement des données

La méthodologie développée dans ce mémoire repose sur un pipeline permettant de transformer les données de mouvement issues de la réalité virtuelle en représentations exploitables par des méthodes d'apprentissage automatique. Ce chapitre se concentre sur la description des jeux de données utilisés ainsi que sur les étapes de structuration, de préparation et de nettoyage des signaux, nécessaires avant toute phase de modélisation ou d'évaluation.

4.1 Présentation des jeux de données

4.1.1 Présentation générale

Les expériences menées dans ce projet s'appuient sur deux jeux de données réels reconnus dans la littérature pour l'étude de l'identifiabilité, de la biométrie comportementale et de la classification démographique en réalité virtuelle : le dataset **FAST** [29] et le dataset **Who is Alyx** [1]. Ces deux sources de données offrent une grande richesse cinématique en enregistrant, à haute fréquence, les mouvements de la tête (HMD), des contrôleurs gauche et droit, ainsi que pour Who is Alyx, des signaux oculaires complets (eye tracking).

Chaque ligne de ces datasets correspond à un **échantillon temporel** (acquisition RV à un instant donné) contenant :

- des **positions 3D** et **rotations** du casque et des mains ;
- des **signaux complémentaires** liés aux interactions (boutons, trackpad, triggers) dans Who is Alyx ;

- des **mesures oculaires** (direction du regard, diamètre pupillaire, ouverture des yeux) pour le dataset Who is Alyx ;
- les **métadonnées** associées (identifiant joueur, identifiant session, temps écoulé) ;
- un **label démographique (gender)** pour les tâches de classification étudiées.

Ces deux jeux de données présentent des caractéristiques complémentaires : FAST offre un suivi exhaustif du corps complet avec plusieurs représentations (position, orientation Euler, quaternion et 6D), tandis que Who is Alyx fournit un suivi précis tête-mains enrichi par l’eye tracking, ce qui permet d’étudier des aspects comportementaux plus fins. Leur combinaison constitue ainsi un cadre expérimental robuste pour analyser la variabilité inter-individuelle, les propriétés biométriques du mouvement RV et la vulnérabilité des modèles aux perturbations comportementales.

4.1.2 Jeu de données FAST

Le jeu de données **FAST** (*Full-Body Action and Sensor Tracking*) constitue l’un des ensembles de données les plus complets disponibles pour l’analyse des mouvements en réalité virtuelle. Il fournit un suivi précis et synchronisé de trois points clés du corps humain : la tête (HMD), la main gauche et la main droite. Chaque échantillon temporel est accompagné d’un horodatage (**Timestamp**), permettant une reconstruction fine de la dynamique du mouvement.

Pour chaque segment corporel, FAST enregistre simultanément :

- **Positions 3D** : coordonnées (x, y, z) décrivant la localisation dans l’espace ;
- **Rotations Euler** : trois angles (`euler_x`, `euler_y`, `euler_z`) capturant l’orientation dans un repère local ;
- **Quaternions** : (x, y, z, w) offrant une représentation sans singularité de la rotation ;
- **Six degrees of freedom (6D)** : six composantes (`sixD_a` à `sixD_f`) permettant d’exprimer les orientations dans un espace continu sans discontinuité angulaire.
- **Données démographiques** : la variable `gender`, codée de façon binaire (0/1), essentielle pour les tâches de classification du genre étudiées dans ce projet.

Les trois groupes de signaux HMD, main gauche, main droite sont enregistrés à une fréquence élevée, ce qui permet d’analyser la fluidité des gestes, la coordination inter-membres et des propriétés biomécaniques fines. Le dataset inclut également la

variable cible **gender**, ce qui rend possible la classification démographique à partir des caractéristiques cinématiques du corps.

Une particularité importante du dataset FAST est la présence de **plusieurs sessions par utilisateur**, chacune représentant un enregistrement indépendant dans le temps. Cette structure multi-sessions permet d'étudier la stabilité des caractéristiques biométriques, la variabilité intra-individuelle, et les capacités de généralisation des modèles, en assurant un découpage Train/Validation/Test strict par joueur afin d'éviter les fuites d'information. Elle offre également la possibilité d'analyser l'évolution du comportement d'un même utilisateur au fil du temps, en tenant compte de phénomènes tels que la fatigue, l'adaptation à la tâche ou les variations naturelles du style de mouvement. Par ailleurs, cette diversité temporelle renforce la pertinence du dataset pour l'étude de la robustesse des modèles, puisque elle permet de tester leur capacité à rester performants face à des conditions d'enregistrement hétérogènes et non contrôlées.

La figure suivante 4.1 présente une vue synthétique et structurée de l'ensemble des informations capturées dans le dataset FAST, en illustrant la manière dont les données du casque, des mains et les annotations de genre sont organisées au sein d'un même échantillon temporel.

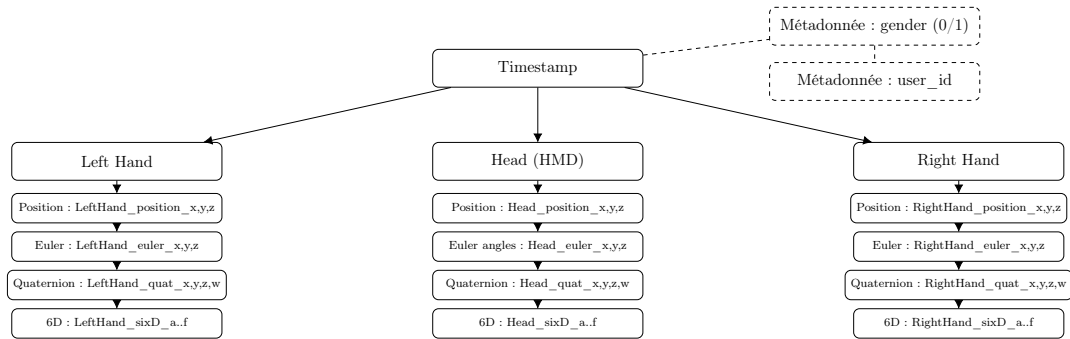


FIGURE 4.1 – Structure du dataset FAST : données de tracking capturées à chaque timestamp (tête, main gauche, main droite) et métadonnées de session (genre et identifiant utilisateur) ajoutées une fois par session.

4.1.3 Organisation et annotation des labels dans le jeu de données FAST

Le jeu de données FAST est fourni sous la forme d'un ensemble de fichiers CSV, chaque fichier correspondant à un utilisateur et à une tâche ou un scénario de réalité

virtuelle donné. Pour un même joueur, plusieurs fichiers peuvent être présents, tels que `BuildA`, `BuildB` ou `BuildTutorial`, reflétant différentes conditions d'enregistrement au sein d'un même identifiant.

Les fichiers FAST ne contiennent pas explicitement de métadonnées indiquant le genre des utilisateurs. L'information de genre est en revanche encodée directement dans le nom des fichiers, où la lettre `F` ou `M` apparaît dans l'identifiant du joueur (par exemple `FAB10068F_BuildA.csv`), indiquant respectivement un utilisateur féminin ou masculin. Afin de rendre cette information exploitable dans un cadre d'apprentissage supervisé, un script a été développé pour extraire automatiquement le genre à partir du nom de fichier et l'ajouter comme colonne explicite (`gender`) lors de la construction de la base de données finale.

Cette étape d'annotation permet de garantir la cohérence des labels et de rendre les données directement exploitables pour les expériences de classification du genre présentées dans les chapitres suivants.

4.1.4 Jeu de données Alyx

Le jeu de données **Who Is Alyx** constitue un ensemble riche et structuré de signaux de mouvement captés en réalité virtuelle, comprenant le suivi du casque de réalité virtuelle (*Head-Mounted Display*, HMD), des deux contrôleurs manuels ainsi qu'un module d'eye-tracking de haute précision. Le HMD correspond à un casque porté sur la tête, équipé d'écrans stéréoscopiques et de capteurs inertiels, permettant de mesurer en temps réel la position et l'orientation de la tête de l'utilisateur dans l'environnement virtuel.

Chaque échantillon temporel inclut des métadonnées telles que `player_id`, `session_id` et `delta_time_ms`, permettant de reconstituer finement l'évolution du comportement moteur et visuel au fil du temps, tout en garantissant une structuration explicite des données par utilisateur et par session.

Contrairement au dataset FAST, Who is Alyx met fortement l'accent sur la **cinématique locale détaillée** et sur l'intégration du **regard** comme signal biométrique. Les données fournies couvrent plusieurs catégories :

- **HMD (Head Mounted Display)** : positions (x, y, z) et rotations en quaternions (x, y, z, w) , permettant une reconstruction précise de l'orientation de la tête en 3D ;

- **Contrôleurs gauche et droit** : positions 3D, rotations en quaternions, ainsi qu’un ensemble complet d’interactions boutons (trigger, grip, menu, trackpad, touches UL), permettant d’étudier la dynamique des mains, la coordination et les micro-mouvements liés aux actions utilisateur ;
- **Eye Tracking** : direction du regard (gauche, droite, combinée) en (x, y, z) , origine du regard, diamètre pupillaire, position dans le capteur et ouverture des yeux. Ces signaux constituent des marqueurs comportementaux riches, utilisés dans de nombreux travaux sur l’identifiabilité biométrique ;
- **Données démographiques** : la variable **gender** (0/1), indispensable pour les tâches de classification du genre étudiées dans ce projet.

Le dataset Who is Alyx est structuré en **sessions indépendantes par utilisateur**, ce qui permet d’évaluer la stabilité intra-individuelle des mouvements RV et d’éviter toute fuite d’information en appliquant un découpage strict Train/Validation/Test par joueur. Grâce à la combinaison unique de signaux tête-mains-yeux, Who is Alyx constitue un support expérimental particulièrement pertinent pour étudier la **reconnaissance du genre** ainsi que les **propriétés biométriques et vulnérabilités comportementales** associées aux mouvements RV. Il se distingue par son niveau de granularité, la richesse des mesures oculaires et la présence d’annotations démographiques, ce qui en fait un cadre idéale pour l’analyse des attributs sensibles et des perturbations comportementales dans les environnements immersifs.

Pour mieux comprendre la richesse multimodale du dataset Who is Alyx, la figure 4.2 synthétise l’ensemble des signaux capturés à chaque instant : métadonnées, trajectoires du casque et des contrôleurs, ainsi que les mesures oculaires. Cette représentation schématique permet de visualiser clairement comment les différentes sources d’information s’organisent au sein d’un même échantillon temporel.

Il est à noter que les features d’eye-tracking ne sont extraites que pour le dataset Who is Alyx, le dataset FAST ne contenant pas ce type de données.

4.1.5 Organisation spécifique du jeu de données *Who Is Alyx*

Le jeu de données *Who Is Alyx* est distribué sous la forme d’une arborescence de dossiers organisée par utilisateur. Chaque dossier principal est associé à un identifiant unique de joueur (*player_id*) et contient un ou plusieurs sous-dossiers correspondant aux différentes sessions d’enregistrement. Ces sous-dossiers sont nommés selon la date

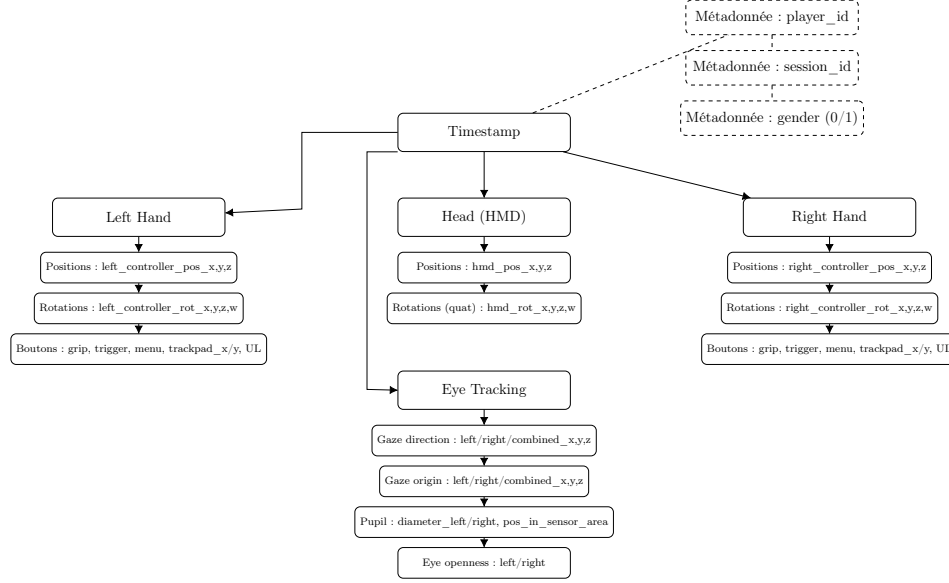


FIGURE 4.2 – Structure du dataset Who is Alyx : données de tracking capturées à chaque timestamp (tête, main gauche, main droite, eye tracking) et métadonnées de session (identifiant joueur, session et genre).

de la session, reflétant le fait qu’un même utilisateur peut avoir participé à plusieurs sessions d’enregistrement distinctes.

Pour chaque session, les données de mouvement et les données d’eye-tracking sont fournies dans des fichiers CSV séparés. Les fichiers de mouvement (`vr-controllers.csv`) contiennent les trajectoires et orientations du casque de réalité virtuelle et des contrôleurs, ainsi que les états des boutons, tandis que les fichiers d’eye-tracking regroupent les informations relatives à la direction du regard et aux paramètres oculaires. Cette séparation des modalités, bien que pertinente du point de vue de l’acquisition, nécessite une étape de fusion préalable afin de produire une représentation cohérente de l’état de l’utilisateur à chaque instant.

Avant la fusion des modalités, une étape d’interpolation temporelle est appliquée aux données de chaque session afin d’harmoniser les fréquences d’échantillonnage. En effet, les flux de données issus du tracking de mouvement et de l’eye-tracking ne partagent pas nécessairement le même pas temporel. Dans ce travail, un index temporel cible est défini à partir d’un taux d’images par seconde constant, et les signaux sont interpolés sur cet axe temporel commun. Cette étape garantit une synchronisation fine entre les modalités et permet de préserver la cohérence temporelle des signaux lors de leur fusion.

Une fois l'interpolation réalisée, les fichiers de mouvement et d'eye-tracking sont concaténés pour chaque session, produisant un tableau unique regroupant l'ensemble des signaux disponibles. Les différentes sessions associées à un même joueur sont ensuite concaténées afin d'obtenir une représentation continue du comportement de l'utilisateur sur l'ensemble de ses sessions d'enregistrement. Enfin, l'ensemble des joueurs est agrégé dans un fichier global, constituant la base de données finale utilisée pour les étapes ultérieures de préparation des données et de modélisation.

Cette procédure de structuration et de fusion des données permet de transformer l'organisation initiale du jeu de données *Who Is Alyx*, orientée acquisition, en un format homogène et exploitable pour l'apprentissage automatique, tout en respectant la structure multi-sessions et multi-utilisateurs du jeu de données [1].

4.2 Analyse descriptive des jeux de données

L'analyse descriptive constitue une étape essentielle avant toute modélisation, car elle permet de comprendre la structure, l'équilibre et la variabilité des données issues des environnements RV FAST et Who is Alyx. Ces jeux de données présentent une richesse importante, mêlant positions, rotations, vitesses, accélérations, signaux oculaires et métadonnées comportementales, mais aussi une forte hétérogénéité inter-joueurs. Les sous-sections suivantes décrivent les principales statistiques globales, la distribution des classes, ainsi que les visualisations exploratoires réalisées.

4.2.1 Statistiques descriptives

Les bases FAST et Who is Alyx comportent plusieurs centaines de milliers de pas temporels, chaque frame décrivant l'état complet du casque, des contrôleurs, de l'œil dominant, ainsi que des signaux dérivés. Les principales statistiques montrent :

- une variance importante sur les positions du HMD (*head height*), reflétant des morphologies différentes entre joueurs ;
- des distributions asymétriques sur les vitesses et accélérations, typiques des gestes RV comprenant des phases de mouvements rapides suivies de phases plus statiques ;
- une amplitude de mouvement plus élevée pour la main dominante, cohérente avec les travaux existants sur l'asymétrie gestuelle en réalité virtuelle ;

- un taux moyen de saccades oculaires par seconde stable entre joueurs, mais une variabilité notable dans la dynamique du regard.

Ces statistiques initiales confirment la présence de motifs comportementaux distinctifs, appuyant la pertinence d’une approche de classification.

4.2.2 Distribution des classes

La variable cible *gender* présente une distribution légèrement déséquilibrée dans les deux datasets. Le dataset Who is Alyx contient 76 joueurs au total, dont 40 masculins (52.6%) et 36 féminins (47.4%). Le dataset FAST contient 107 participants, dont 56 masculins (52.3%) et 51 féminins (47.7%).

Pour le dataset Who is Alyx, ce déséquilibre est préservé après le split train/test groupé par joueur : l’ensemble d’entraînement contient 53 joueuses (53%) et 47 joueurs (47%), tandis que l’ensemble de test contient 23 joueuses (52.3%) et 21 joueurs (47.7%). Bien que ce déséquilibre soit modéré, l’utilisation de la *balanced accuracy* comme métrique d’évaluation est justifiée : elle garantit que les performances ne sont pas artificiellement gonflées par la légère surreprésentation d’une classe.

Les analyses montrent également que :

- chaque joueur appartient exclusivement à une seule classe ;
- les sessions ne sont pas homogènes en durée, mais ce biais est corrigé lors de l’agrégation statistique par session ;
- la distribution des labels est similaire dans les splits entraînement (53% / 47%) et test (52.3% / 47.7%), grâce au partitionnement strict par joueur.

Cette homogénéité garantit la validité statistique des protocoles d’entraînement et d’évaluation utilisés dans la suite.

4.2.3 Visualisations exploratoires

Avant l’entraînement des modèles, plusieurs visualisations ont été réalisées sur les deux datasets afin d’examiner la structure interne des signaux RV et d’identifier des patterns cinématiques caractéristiques.

4.2.3.1 Dataset Who is Alyx

La Figure 4.3 présente la distribution de la variable `hmd_pos_y`, représentative de la hauteur de la tête. La densité observée montre une distribution unimodale concentrée

autour de 160 unités, avec une légère asymétrie positive et quelques valeurs extrêmes au-delà de 300. Ces valeurs aberrantes correspondent à des événements rares de déplacement brusque du casque.

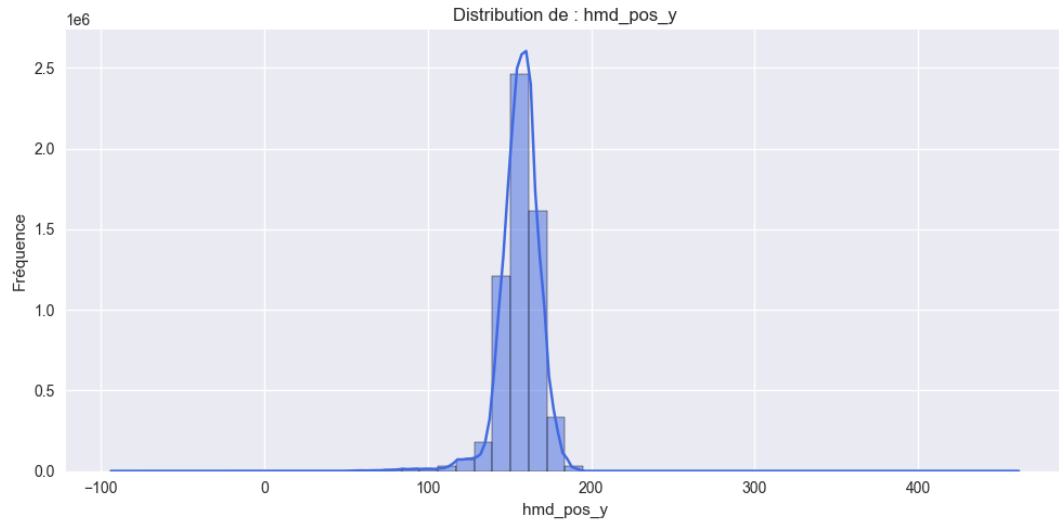


FIGURE 4.3 – Distribution de la variable `hmd_pos_y` - dataset Who is Alyx.

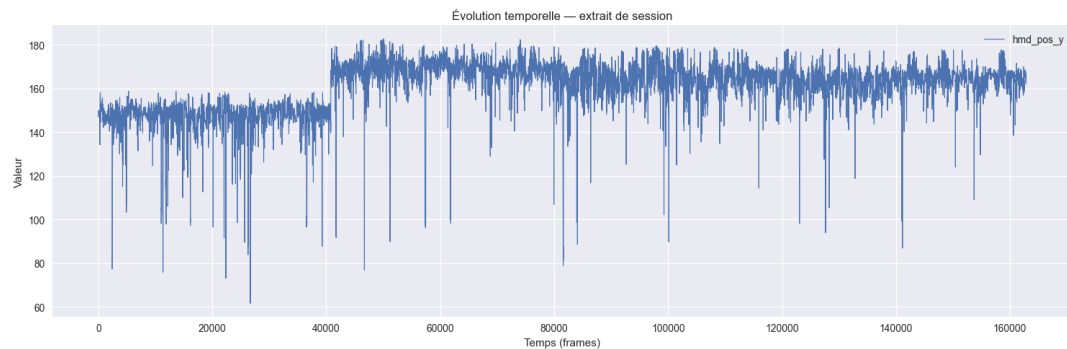


FIGURE 4.4 – Évolution temporelle de la hauteur de tête (extrait de session) - dataset Who is Alyx.

La Figure 4.4 illustre l'évolution temporelle de la hauteur de tête sur une session complète de plus de 160 000 frames. On observe une forte variabilité intra-session avec des pics descendants marqués, traduisant des baisses brusques de la tête lors de certaines actions du jeu. Un changement de niveau notable apparaît vers la frame 40 000, où la hauteur moyenne passe d'environ 150 à 165 unités, suggérant un changement de posture ou de position du joueur au cours de la session.

La Figure 4.5 propose une matrice de corrélation entre plusieurs variables brutes du dataset Who is Alyx.

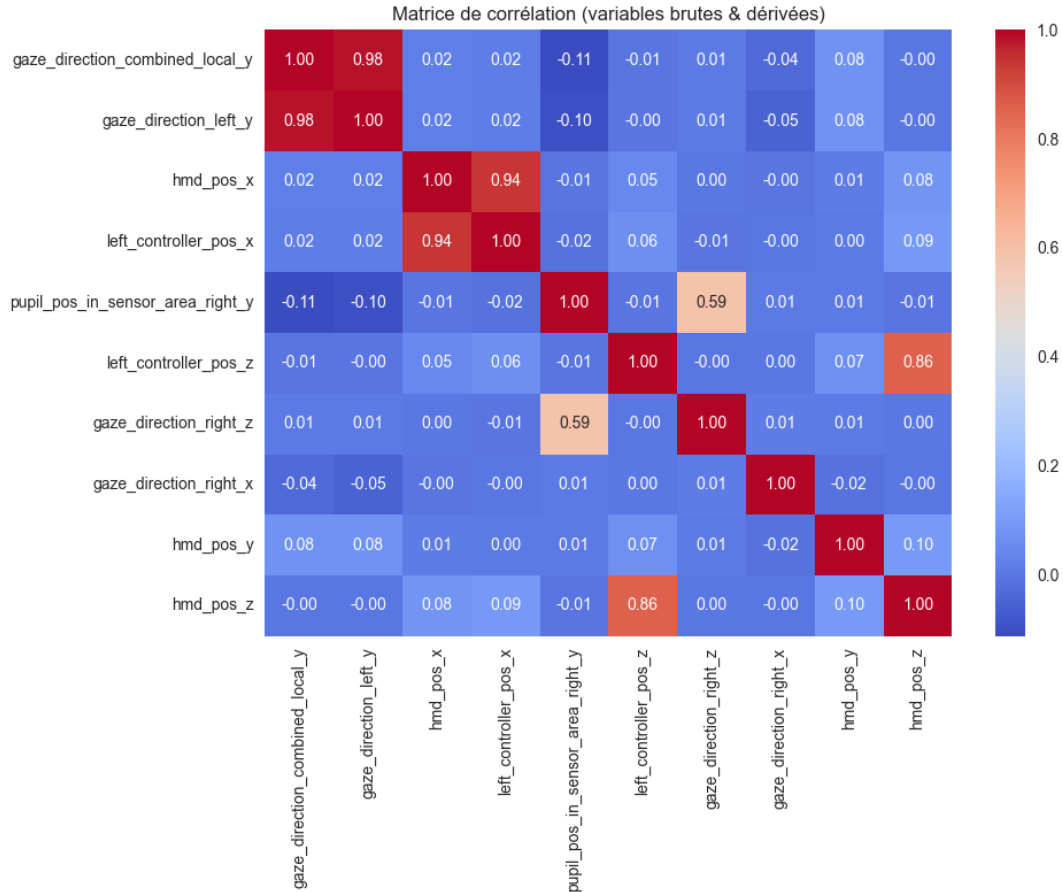


FIGURE 4.5 – Matrice de corrélation des variables brutes et dérivées - Who is Alyx.

Trois corrélations notables sont observées : les directions du regard combiné et gauche selon l'axe y présentent une forte redondance ($r = 0.98$), attendue car ces deux signaux mesurent la même direction de regard depuis des capteurs distincts. La position latérale du casque et du contrôleur gauche sont également fortement corrélées ($r = 0.94$), reflétant le déplacement coordonné de l'ensemble du corps dans l'axe x . Enfin, les positions selon l'axe z du contrôleur gauche et du casque présentent une corrélation élevée ($r = 0.86$), ce qui reflète la tendance naturelle des joueurs à déplacer simultanément la main et la tête dans la même direction de profondeur. En revanche, les signaux oculaires (**gaze**, **pupil**) conservent une indépendance quasi-totale vis-à-vis des signaux de position du casque et des contrôleurs, confirmant que les modalités motrices et oculaires

transportent des informations complémentaires et justifiant leur intégration conjointe dans les modèles.

4.2.3.2 Dataset FAST

La Figure 4.6 présente la distribution de la variable `Head_position_y`. La densité observée présente une distribution bimodale concentrée autour de 1.5m, reflétant la hauteur standard du casque RV. Les queues fines suggèrent peu d'événements extrêmes comparativement au dataset Who is Alyx.

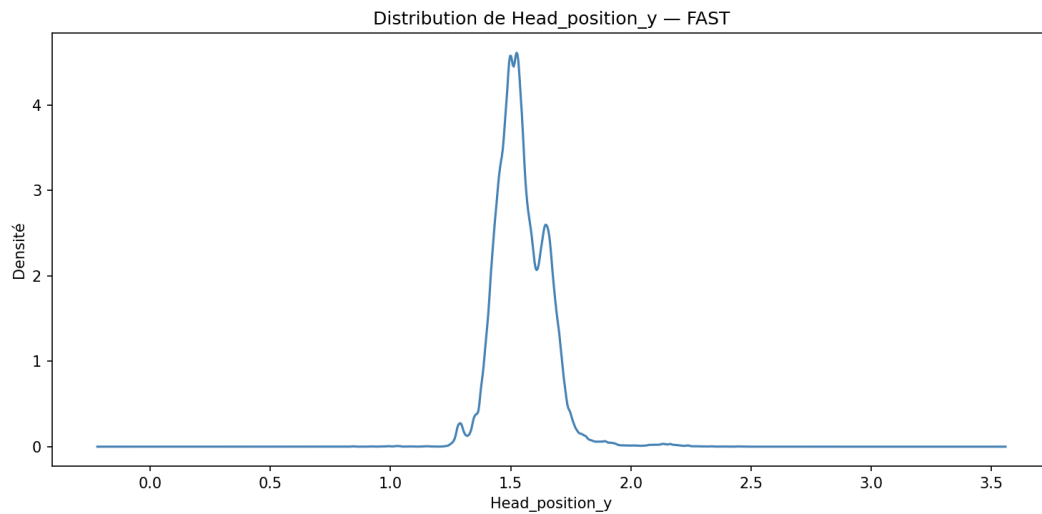


FIGURE 4.6 – Distribution de la variable `Head_position_y` - dataset FAST.

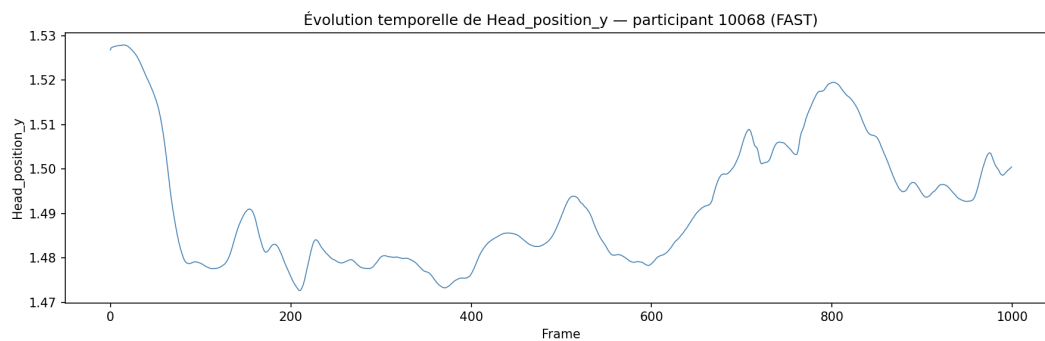


FIGURE 4.7 – Évolution temporelle de `Head_position_y` - participant 10068 (FAST).

La Figure 4.7 illustre l'évolution temporelle de la hauteur de tête pour le participant 10068. On observe une variabilité modérée ($\approx 0.06\text{m}$ d'amplitude) avec des fluctuations lentes et continues, traduisant des ajustements posturaux progressifs.

La Figure 4.8 présente la matrice de corrélation entre les variables de position et d'orientation du casque et de la main gauche. Deux corrélations notables sont observées : `Head_position_x` et `LeftHand_position_x` ($r = 0.70$), indiquant un fort couplage latéral entre la tête et la main gauche, et `Head_position_z` et `LeftHand_position_z` ($r = 0.58$).



FIGURE 4.8 – Matrice de corrélation des variables de position et d'orientation - dataset FAST.

Ces visualisations mettent en évidence la richesse et l'hétérogénéité des signaux RV dans les deux datasets. Si les deux datasets partagent des propriétés structurelles communes (variabilité intra-session, couplage entre membres), des différences notables appa-

raissent : une amplitude de mouvement plus restreinte dans FAST, une distribution de hauteur bimodale liée au protocole d’acquisition, et une absence de signaux oculaires. Ces observations justifient les adaptations méthodologiques décrites dans les sections suivantes.

4.3 Préparation et nettoyage des données

Cette section décrit le nettoyage des données, dont les étapes varient selon le type de modèle utilisé.

Pour les modèles classiques (Random Forest, XGBoost, LightGBM, GradientBoosting, MLP), chaque session est résumée par un ensemble de statistiques (moyenne, écart-type, médiane, etc.) et stockée comme une seule ligne dans le dataset tabulaire. Le nettoyage s’applique donc sur ces statistiques agrégées.

Pour les modèles temporels (LSTM, CNN–LSTM, ResNet1D, TimesNet), les données brutes frame par frame sont utilisées directement. Les valeurs manquantes qui apparaissent lors du calcul des vitesses et accélérations notamment à la première frame de chaque session où la différence ne peut pas être calculée sont remplacées par zéro. Les valeurs infinies ($\pm\infty$) sont d’abord converties en valeurs manquantes, puis également remplacées par zéro.

Les données brutes issues des sessions de réalité virtuelle présentent plusieurs particularités susceptibles d’affecter l’apprentissage automatique, notamment le bruit sensoriel, les pertes de suivi, la présence de valeurs aberrantes, des différences d’échantillonnage entre dispositifs, ainsi qu’une structure multimodale intégrant positions, rotations, états des contrôleurs et signaux oculaires. Un pré-traitement rigoureux est donc nécessaire afin de garantir la qualité, la cohérence et l’exploitabilité des données avant toute phase de modélisation.

Les sous-sections suivantes décrivent le nettoyage appliqué aux données tabulaires. Le traitement spécifique aux données brutes pour les modèles temporels a été décrit ci-dessus.

4.3.1 Nettoyage des valeurs manquantes et aberrantes

Les fichiers bruts comportent des valeurs manquantes (NaN), des valeurs non finies ($\pm\infty$) ainsi que des discontinuités liées à des micro-coupures du système de tracking.

Conformément aux travaux existants sur le traitement des signaux de mouvement en réalité virtuelle [15, 16], toutes les valeurs non finies sont d’abord remplacées par NaN. Les colonnes entièrement manquantes sont ensuite supprimées.

Les valeurs manquantes résiduelles sont imputées à l’aide de la médiane de chaque colonne, calculée globalement sur l’ensemble du jeu de données via `X.fillna(X.median())`. Cette approche présente deux avantages : d’une part, la médiane est robuste aux valeurs extrêmes contrairement à la moyenne [130] ; d’autre part, cette imputation est appliquée en phase de prétraitement hors-ligne sur des features déjà agrégées par session (moyenne, écart-type, médiane, etc.), et non sur des séries temporelles brutes frame par frame. La notion de fenêtre glissante ou de causalité temporelle ne s’applique donc pas ici : chaque ligne du dataset représente une session entière résumée par ses statistiques, ce qui en fait une table tabulaire classique plutôt qu’une série temporelle.

Enfin, les valeurs aberrantes résiduelles sont tronquées dans l’intervalle $[-10^6, 10^6]$ via une opération de *clipping*, afin de prévenir l’instabilité numérique lors de l’entraînement des modèles.

4.3.2 Harmonisation temporelle

Les différents dispositifs de suivi utilisés en réalité virtuelle (casque, contrôleurs, eye-tracker) ne partagent pas nécessairement le même taux d’échantillonnage. Afin d’assurer une cohérence temporelle entre les signaux, les vitesses et accélérations sont recalculées à partir de l’intervalle temporel effectif `delta_time_ms`. Cette harmonisation permet de rendre les mesures comparables entre utilisateurs et sessions, et de réduire les biais liés aux variations d’échantillonnage [131].

4.3.3 Nettoyage structurel

Les colonnes ne devant pas être utilisées comme entrées des modèles d’apprentissage, telles que `player_id`, `session_id` et `gender`, sont supprimées du pipeline. Cette étape est essentielle pour éviter toute fuite d’information identitaire et garantir une évaluation réaliste des performances des modèles, conformément aux bonnes pratiques recommandées dans la littérature [2, 27].

4.3.4 Ajout de caractéristiques dérivées

Afin d'enrichir la représentation des données, plusieurs caractéristiques biomécaniques explicites sont dérivées des signaux bruts.

4.3.4.1 Caractéristiques morphologiques

La première caractéristique dérivée est la largeur des épaules (`shoulder_width`), calculée comme la distance absolue entre les positions horizontales des deux contrôleurs :

$$\text{shoulder_width} = |x_{\text{right}} - x_{\text{left}}| \quad (4.1)$$

où x_{right} et x_{left} désignent respectivement les coordonnées en x du contrôleur droit et du contrôleur gauche. Cette mesure constitue un indicateur morphologique direct de la corpulence du joueur, les hommes présentant en moyenne une envergure des épaules plus large que les femmes.

Le ratio tête/épaules (`head_to_shoulder_ratio`) est ensuite défini comme le rapport entre la hauteur du casque et la largeur des épaules :

$$\text{head_to_shoulder_ratio} = \frac{y_{\text{hmd}}}{\text{shoulder_width} + \epsilon} \quad (4.2)$$

où y_{hmd} est la coordonnée verticale du casque de réalité virtuelle et $\epsilon = 10^{-6}$ est un terme de stabilité numérique ajouté pour éviter toute division par zéro. Ce ratio capture la proportion entre la taille perçue et l'envergure du joueur, deux dimensions morphologiques connues pour présenter des différences significatives entre genres.

4.3.4.2 Caractéristiques d'asymétrie

Une troisième caractéristique, l'asymétrie de vitesse d'activation des mains (`hand_asymmetry_speed`), est calculée comme la différence entre les valeurs de pression des gâchettes droite et gauche :

$$\text{hand_asymmetry_speed} = \text{trigger}_{\text{right}} - \text{trigger}_{\text{left}} \quad (4.3)$$

Cette mesure quantifie la dominance manuelle du joueur et l'asymétrie de son comportement d'interaction dans l'environnement virtuel. Une valeur positive indique une dominance de la main droite, tandis qu'une valeur négative indique une dominance de la

main gauche. Des études en biomécanique ont montré que la latéralité manuelle présente des patterns distincts selon le genre [132].

Ces trois caractéristiques morphologiques et comportementales viennent compléter les features statistiques et dynamiques extraites par agrégation temporelle, offrant ainsi une représentation plus riche et plus interprétable des comportements des joueurs en réalité virtuelle.

4.3.5 Normalisation des données

Après l'ensemble des étapes de nettoyage et d'enrichissement, un jeu final de colonnes numériques est constitué. Les colonnes constantes ou fortement corrélées ($r > 0.95$) sont supprimées afin de réduire la redondance des informations. Une normalisation standardisée de type *z-score* est ensuite appliquée, en respectant strictement la séparation entre les ensembles d'apprentissage et de test, conformément aux bonnes pratiques en apprentissage automatique [133].

4.4 Synthèse du pipeline expérimental

Ce chapitre a présenté les jeux de données utilisés ainsi que les différentes étapes de préparation nécessaires à l'exploitation des signaux de mouvement en réalité virtuelle. À partir des données brutes issues des dispositifs de suivi, un pipeline de traitement a été mis en place afin de structurer les signaux, d'harmoniser leur dimension temporelle et d'extraire des représentations cohérentes et exploitables pour l'apprentissage automatique. Les données ainsi préparées constituent une base fiable et homogène, permettant d'étudier les capacités des modèles à reconnaître le genre à partir du mouvement. Le chapitre suivant s'appuie directement sur ces représentations pour analyser les méthodes de classification du genre en réalité virtuelle et évaluer leurs performances en l'absence de perturbations.

Chapitre 5

Reconnaissance du genre à partir du mouvement en RV

Ce chapitre est consacré à l'étude de la classification du genre à partir des données de mouvement issues de la réalité virtuelle. L'objectif principal est d'évaluer dans quelle mesure les signaux de mouvement captés lors de l'interaction avec un environnement immersif permettent d'identifier le genre des utilisateurs à l'aide de modèles d'apprentissage automatique.

Deux types de représentations des signaux de mouvement sont explorés. D'une part, des représentations tabulaires sont construites en agrégeant chaque session en un vecteur de descripteurs statistiques et biomécaniques (moyenne, écart-type, médiane, etc.) une ligne par session. Ces représentations sont utilisées par les modèles classiques (Random Forest, XGBoost, LightGBM, GradientBoosting, MLP). D'autre part, des représentations temporelles préservent les données brutes frame par frame afin de capturer explicitement la dynamique des gestes et des trajectoires. Ces représentations sont utilisées par les modèles séquentiels (LSTM, CNN-LSTM, ResNet1D, TimesNet), pour lesquels le nettoyage des valeurs manquantes est effectué directement sur les séries temporelles brutes, comme décrit au chapitre précédent.

Deux protocoles d'évaluation sont utilisés dans ce chapitre. Le premier est un protocole intra-dataset, où l'entraînement et le test sont effectués sur le même dataset (FAST ou Who is Alyx). Le second est un protocole cross-dataset, où le modèle est entraîné sur un dataset et évalué sur l'autre, afin de tester la capacité de généralisation des modèles au-delà des conditions d'enregistrement spécifiques à chaque jeu de données. Ce second

protocole est particulièrement important car il reflète des conditions d'utilisation réelles où le domaine d'entraînement et le domaine de déploiement peuvent différer.

L'analyse conduite dans ce chapitre vise ainsi à répondre à la première question de recherche de ce mémoire : dans quelle mesure les données de mouvement en réalité virtuelle constituent une source d'information pertinente pour la classification du genre des utilisateurs. Les résultats obtenus définissent un point de référence essentiel sur lequel s'appuieront les chapitres suivants.

5.1 Expérimentations Cross-Dataset : Entraînement sur Alyx, Test sur FAST

Dans cette expérimentation, les modèles sont entraînés sur le dataset Who is Alyx et évalués sur le dataset FAST. Cette approche *cross-dataset* vise à évaluer la capacité de généralisation des modèles à un environnement RV différent, avec des participants distincts, un dispositif de capture différent et des conditions expérimentales propres à chaque dataset.

Une contrainte importante de cette configuration est que le dataset FAST ne contient pas de données d'eye-tracking, contrairement au dataset Alyx. Les features liées au regard (direction du regard, vitesse oculaire, diamètre des pupilles, ouverture des yeux) ne peuvent donc pas être utilisées dans ce protocole cross-dataset. Les modèles sont ainsi entraînés et évalués uniquement sur les features communes aux deux datasets, c'est-à-dire les données de mouvement de la tête et des contrôleurs.

L'enjeu principal de cette évaluation est de déterminer si les signatures comportementales discriminantes apprises sur Alyx telles que les patterns de mouvement et la dynamique corporelle restent exploitables pour prédire le genre sur des données issues d'un contexte différent. Un modèle présentant une bonne généralisation cross-dataset indiquerait que les différences comportementales liées au genre sont suffisamment universelles pour transcender les spécificités techniques et contextuelles de chaque environnement RV.

5.1.1 Préparation des données

Dataset Who is Alyx : Le dataset Who is Alyx est chargé depuis les fichiers bruts de chaque joueur, organisés en sessions distinctes. Pour chaque joueur, les fichiers

`vr-controllers*.csv` de toutes les sessions sont concaténés en un seul `DataFrame`. Une hétérogénéité du schéma de colonnes a été identifiée pour le joueur 65, dont la session du 2022-08-03 contient 112 colonnes au lieu de 41 pour les autres sessions. Cette différence est gérée automatiquement par le paramètre `usecols` qui ne retient que les colonnes communes définies dans `BASE_COLS`, un ensemble prédéfini de 41 colonnes correspondant aux signaux de mouvement disponibles pour tous les joueurs : positions et rotations du casque (HMD), positions, rotations et états des boutons des deux contrôleurs (gauche et droit), ainsi que l’horodatage et le temps delta entre frames. Ce filtrage garantit la cohérence des features chargées pour l’ensemble des joueurs. Les métadonnées (identifiant du joueur et genre) sont extraites depuis le fichier `summary.csv`, le genre étant encodé binaires (masculin $\rightarrow$ 1, féminin $\rightarrow$ 0). Le dataset final contient 75 joueurs avec leurs sessions complètes.

Dataset FAST : Le dataset FAST est organisé différemment : chaque participant dispose de trois fichiers CSV correspondant aux environnements `BuildA`, `BuildB` et `BuildTutorial`. L’identifiant du participant et son genre sont extraits directement depuis le nom du fichier (par exemple `FAB10068F_BuildA.csv`, où `F` indique le genre féminin et `M` le genre masculin). Les trois fichiers de chaque participant sont concaténés en un seul `DataFrame`. Le dataset final contient 107 participants.

Harmonisation des colonnes : Les deux datasets utilisent des conventions de nommage différentes pour les mêmes signaux physiques. Un renommage est appliqué aux colonnes du dataset FAST pour les aligner sur celles d’Alyx. Le Tableau 5.1 présente la correspondance entre les colonnes des deux datasets.

Après harmonisation, seules les colonnes communes aux deux datasets sont conservées, constituant un espace de features partagé de 19 signaux bruts : la position et l’orientation du casque (`hmd`), ainsi que la position et l’orientation des deux contrôleurs (gauche et droit). Cette réduction à l’intersection des colonnes disponibles garantit la cohérence des features utilisées pour l’entraînement sur Alyx et l’évaluation sur FAST.

TABLEAU 5.1 – Correspondance des colonnes entre le dataset FAST et le dataset Alyx

Colonne FAST	Colonne Alyx
Timestamp	delta_time_ms
Head_position_x	hmd_pos_x
Head_position_y	hmd_pos_y
Head_position_z	hmd_pos_z
Head_euler_x	hmd_rot_x
Head_euler_y	hmd_rot_y
Head_euler_z	hmd_rot_z
LeftHand_position_x	left_controller_pos_x
LeftHand_position_y	left_controller_pos_y
LeftHand_position_z	left_controller_pos_z
LeftHand_euler_x	left_controller_rot_x
LeftHand_euler_y	left_controller_rot_y
LeftHand_euler_z	left_controller_rot_z
RightHand_position_x	right_controller_pos_x
RightHand_position_y	right_controller_pos_y
RightHand_position_z	right_controller_pos_z
RightHand_euler_x	right_controller_rot_x
RightHand_euler_y	right_controller_rot_y
RightHand_euler_z	right_controller_rot_z

5.1.2 Approche : Agrégation Temporelle par Seconde sur 49 Combinaisons

Cette Approche propose une représentation alternative basée sur des statistiques simples calculées par seconde, indépendantes de la fréquence d'échantillonnage.

5.1.2.1 Construction des Features

La fonction `featurize_by_second` découpe le signal temporel en fenêtres d'une seconde et calcule cinq statistiques agrégées sur chaque fenêtre :

$$\mathbf{f}_{pid} = \frac{1}{S} \sum_{s=1}^S \left[\min_s, \max_s, \mu_s, \sigma_s, \text{med}_s \right] \quad (5.1)$$

où S est le nombre de secondes de la session, $\min_s$, $\max_s$, μ_s , σ_s et med_s sont respectivement le minimum, le maximum, la moyenne, l'écart-type et la médiane des valeurs de la feature dans la seconde s . Le vecteur final $\mathbf{f}_{pid}$ est obtenu en moyennant ces statistiques sur toutes les secondes de la session, produisant un vecteur de features unique par participant.

5.1.2.2 Combinaisons de Trackers et Représentations Spatiales

Deux axes de variation sont explorés :

Configurations de trackers (7) :

- **H+L+R** : tête + main gauche + main droite
- **H+R** : tête + main droite
- **H+L** : tête + main gauche
- **L+R** : main gauche + main droite (sans tête)
- **H** : tête seule
- **R** : main droite seule
- **L** : main gauche seule

Représentations spatiales (7) :

- **Pos+Rot** : position et rotation de tous les trackers sélectionnés
- **Pos** : position uniquement
- **Rot** : rotation uniquement
- **Pos_H** : position de la tête uniquement
- **Rot_H** : rotation de la tête uniquement
- **Pos_LC** : position de la main gauche uniquement
- **Pos_RC** : position de la main droite uniquement

La combinaison de ces deux axes produit 49 configurations au total, dont 26 sont effectivement générées après filtrage des combinaisons vides (intersection vide entre tracker et représentation spatiale). Le nombre de features par combinaison varie de 15 (position d'un seul tracker, 3 axes $\times$ 5 statistiques) à 45 (position des trois trackers).

5.1.2.3 Résultats

Le Tableau 5.2 présente les résultats des 26 combinaisons évaluées avec un SVM optimisé ($C = 10$, $\gamma = 0.001$) dans les deux sens de transfert.

Plusieurs observations se dégagent. Premièrement, les combinaisons incluant la position de la tête obtiennent les meilleures performances, notamment **H+L_Pos** et **H+L_Pos+Rot** (0.730 en moyenne), suggérant que la position du casque constitue la source d'information la plus universelle pour la prédiction du genre cross-dataset. Deuxièmement, les combinaisons basées uniquement sur la main droite (**R_Pos**,

R_Pos+Rot) produisent les performances les plus faibles (0.499), proches du hasard, indiquant que ce signal seul ne suffit pas à capturer des signatures comportementales généralisables. Troisièmement, l'ajout de la rotation aux positions (**Pos+Rot** vs **Pos**) n'améliore pas systématiquement les performances, suggérant que les rotations apportent peu d'information supplémentaire pour la généralisation cross-dataset.

TABLEAU 5.2 – Résultats cross-dataset par combinaison de trackers et représentation spatiale (SVM optimisé, triés par moyenne décroissante)

Combinaison	Alyx→FAST	FAST→Alyx	Moyenne
H+L_Pos	0.723	0.736	0.730
H+L_Pos+Rot	0.723	0.736	0.730
H_Pos	0.703	0.738	0.720
H+R_Pos_H	0.703	0.738	0.720
H+L+R_Pos_H	0.703	0.738	0.720
H_Pos_H	0.703	0.738	0.720
H_Pos+Rot	0.703	0.738	0.720
H+L_Pos_H	0.703	0.738	0.720
H+L+R_Pos+Rot	0.712	0.726	0.719
H+L+R_Pos	0.712	0.726	0.719
H+R_Pos+Rot	0.712	0.689	0.700
H+R_Pos	0.712	0.689	0.700
L+R_Pos	0.596	0.550	0.573
L+R_Pos+Rot	0.596	0.550	0.573
H+L_Pos_LC	0.525	0.615	0.570
H+L+R_Pos_LC	0.525	0.615	0.570
L+R_Pos_LC	0.525	0.615	0.570
L_Pos	0.525	0.615	0.570
L_Pos+Rot	0.525	0.615	0.570
L_Pos_LC	0.525	0.615	0.570
H+R_Pos_RC	0.491	0.507	0.499
H+L+R_Pos_RC	0.491	0.507	0.499
L+R_Pos_RC	0.491	0.507	0.499
R_Pos	0.491	0.507	0.499
R_Pos+Rot	0.491	0.507	0.499
R_Pos_RC	0.491	0.507	0.499

5.1.2.4 Visualisation par Heatmap

La Figure 5.1 présente une vue synthétique des résultats sous forme de carte de chaleur, permettant d'identifier visuellement les combinaisons optimales de trackers et de représentations spatiales pour la généralisation cross-dataset.

La heatmap confirme et complète les observations du Tableau 5.2. Trois zones distinctes se dégagent. La zone verte (supérieure gauche) regroupe les combinaisons incluant la tête avec les représentations **Pos+Rot**, **Pos** et **Pos_H**, atteignant des performances entre 0.700 et 0.730. La zone orange (milieu) regroupe les combinaisons basées sur la main gauche seule (**Pos_LC**), avec des performances autour de 0.570. La zone rouge (droite) correspond aux combinaisons basées sur la main droite seule (**Pos_RC**), dont les performances proches de 0.499 indiquent une absence de généralisation. Les cellules N/A correspondent aux combinaisons dont l'intersection entre la configuration de trackers et la représentation spatiale est vide.

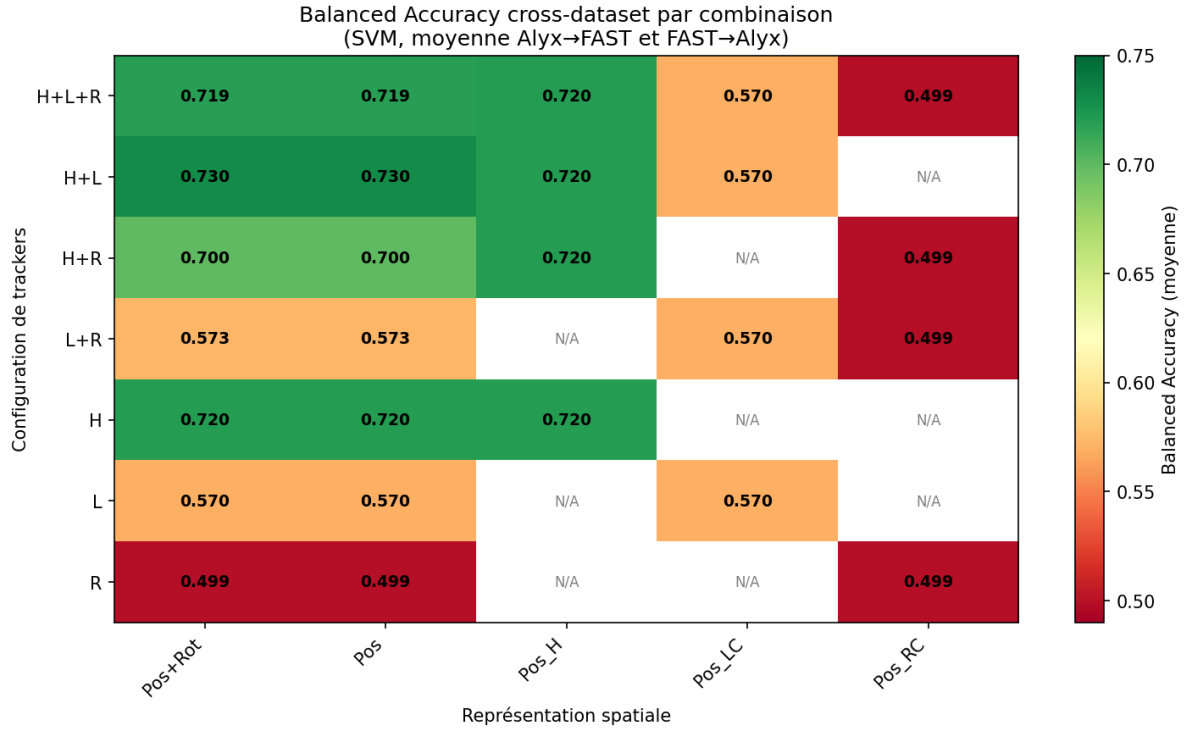


FIGURE 5.1 – Balanced Accuracy moyenne (Alyx→FAST et FAST→Alyx) par combinaison de trackers et représentation spatiale, évaluée avec un SVM optimisé ($C = 10$, $\gamma = 0.001$). Les cellules vertes indiquent les meilleures performances, les cellules rouges les plus faibles, et N/A les combinaisons inexistantes.

Il convient de noter que ces valeurs correspondent à la moyenne des deux sens de transfert, présentés séparément dans le Tableau 5.2. Pour les meilleures combinaisons (tête et main gauche, avec les représentations **Pos** ou **Pos+Rot**), le transfert FAST→Alyx (0.736) est systématiquement légèrement supérieur au transfert

Alyx→FAST (0.723), avec un écart d'environ 0.01 à 0.04 point selon la combinaison considérée. Cette asymétrie reste toutefois modérée et n'est pas systématique dans le même sens : pour certaines combinaisons impliquant les deux mains sans la tête (par exemple **H+R_Pos**), c'est au contraire le transfert Alyx→FAST qui se montre supérieur (0.712 contre 0.689). Aucun des deux sens de transfert ne s'avère donc systématiquement meilleur ou pire que l'autre à l'échelle de l'ensemble des combinaisons testées.

Les expérimentations présentées dans la suite de ce travail sont conduites sur le dataset *Who is Alyx*, retenu comme cadre expérimental principal pour plusieurs raisons. Premièrement, avec 75 joueurs enregistrés sur plusieurs sessions, ce dataset offre une diversité comportementale suffisante pour l'entraînement et l'évaluation de modèles de classification robustes. Deuxièmement, la richesse des signaux capturés : position et orientation du casque, des deux contrôleurs, boutons, trackpad et triggers permet d'extraire un large éventail de features comportementales couvrant la posture, la dynamique et la coordination inter-mains. Troisièmement, la disponibilité des métadonnées (genre, identifiant du joueur) dans le fichier `summary.csv` facilite la construction d'un protocole d'évaluation rigoureux.

5.2 Représentation tabulaire des signaux de mouvement

Pour les modèles de classification classiques, les signaux de mouvement en réalité virtuelle, initialement représentés sous forme de séquences temporelles, sont transformés en vecteurs de caractéristiques de dimension fixe. Cette représentation repose sur l'extraction de descripteurs statistiques et dynamiques permettant de résumer les propriétés globales du mouvement au sein d'une session. Une telle approche est largement utilisée en biométrie comportementale basée sur le mouvement, notamment pour la reconnaissance du genre, et permet l'entraînement efficace de modèles tabulaires tels que les Random Forests, les SVM ou les méthodes de boosting [6].

5.2.1 Extraction statistique

Les signaux bruts issus des dispositifs de suivi en réalité virtuelle (positions, rotations, états des contrôleurs et mesures oculométriques) présentent une forte variabilité temporelle et une dimensionnalité élevée. Ces données sont généralement représentées

sous forme de séries temporelles multivariées, ce qui rend leur exploitation directe par des modèles de classification tabulaire difficile. Afin de rendre ces signaux exploitables par des algorithmes d'apprentissage supervisé classiques, une étape de *featurisation statistique* est appliquée à chaque variable continue. Cette étape consiste à résumer l'évolution temporelle du signal par un ensemble de descripteurs numériques compacts capturant ses propriétés statistiques et dynamiques.

Pour chaque variable temporelle $x_1, \dots, x_n$, des statistiques de premier ordre sont calculées afin de caractériser la tendance centrale du signal. Celles-ci incluent notamment la moyenne

$$\mu = \frac{1}{n} \sum_{i=1}^n x_i$$

ainsi que la médiane, qui constitue une mesure robuste de la tendance centrale moins sensible aux valeurs aberrantes. Des statistiques de dispersion sont également extraites, notamment l'écart-type

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2}$$

qui quantifie la variabilité globale du signal. Le minimum et le maximum permettent également de caractériser les bornes observées du signal au cours de la session.

Afin de capturer plus précisément l'amplitude des variations, l'étendue (*range*) et l'intervalle interquartile (IQR) sont également calculés. Ces indicateurs permettent de mesurer la dispersion effective des données tout en limitant l'influence des valeurs extrêmes. Le coefficient de variation est également introduit afin de normaliser la dispersion par rapport à la moyenne du signal :

$$CV = \frac{\sigma}{\mu}$$

Ce descripteur facilite la comparaison entre variables de différentes échelles et permet d'identifier les signaux présentant une variabilité relative importante. Ce type de featurisation statistique est largement utilisé dans l'analyse de séries temporelles et dans les approches d'apprentissage automatique appliquées à des données biomécaniques ou comportementales [134].

Afin de caractériser plus finement la forme de la distribution des signaux, des indicateurs statistiques de plus haut ordre sont également calculés, notamment l'asymétrie (*skewness*) et l'aplatissement (*kurtosis*). Ces mesures décrivent respectivement le degré d'asymétrie et la concentration de la distribution d'un signal et sont couramment utilisées dans l'analyse de séries temporelles et de signaux biomécaniques [135]. Pour une variable temporelle $x_1, \dots, x_n$ de moyenne μ et d'écart-type σ , ces mesures sont définies par :

$$\text{skewness} = \frac{1}{n} \sum_{i=1}^n \left(\frac{x_i - \mu}{\sigma} \right)^3$$

$$\text{kurtosis} = \frac{1}{n} \sum_{i=1}^n \left(\frac{x_i - \mu}{\sigma} \right)^4$$

Enfin, des indicateurs avancés de dynamique du mouvement sont intégrés afin de capturer des propriétés comportementales plus fines. L'entropie du signal permet d'évaluer la complexité et l'imprévisibilité des séries temporelles issues des mouvements de l'utilisateur [136]. Le ratio de rectitude (*straightness ratio*) quantifie le degré de linéarité des déplacements et est défini comme le rapport entre la distance euclidienne entre le point initial et final et la longueur totale de la trajectoire [137] :

$$\text{straightness ratio} = \frac{\|x_{\text{end}} - x_{\text{start}}\|}{\sum_{i=1}^{n-1} \|x_{i+1} - x_i\|}$$

Un indice de lissage (*smoothness index*) est également calculé afin d'estimer la continuité du mouvement et de détecter des gestes saccadés ou irréguliers. Ces mesures sont couramment utilisées dans l'analyse biomécanique du mouvement humain [134, 138].

5.2.2 Caractéristiques morphologiques et dynamiques dérivées

En complément des descripteurs statistiques, des caractéristiques morphologiques et dynamiques dérivées sont intégrées au vecteur tabulaire final. Ces attributs incluent notamment des mesures liées à la morphologie, à l'asymétrie des mouvements, aux vitesses et à la fluidité gestuelle. Comme présenté dans la Sous-section 4.3.4. Ils permettent de capturer des propriétés biomécaniques associées à la posture, à la coordination et au style moteur des utilisateurs, qui ont été identifiées comme pertinentes pour la reconnaissance du genre à partir du mouvement dans la littérature existante [6, 132].

Au total, 1 242 caractéristiques sont initialement extraites à partir des signaux bruts par agrégation statistique et dynamique sur l'ensemble de chaque session. Après suppression des 64 colonnes constantes et des 494 colonnes présentant une corrélation supérieure à 0.95, 684 caractéristiques sont retenues, organisées en sept catégories décrites dans le Tableau 5.3.

TABLEAU 5.3 – Résumé des caractéristiques extraites par catégorie après nettoyage

Catégorie	Nb	Description
HMD - position	34	Statistiques sur les coordonnées (x, y, z) du casque : moyenne, écart-type, min, max, médiane, étendue, asymétrie, kurtosis, IQR, coefficient de variation, vitesse moyenne, accélération, entropie de mouvement
HMD - rotation	52	Statistiques sur l'orientation du casque (quaternions w, x, y, z) : mêmes agrégats que la position, complétés par des indicateurs de vitesse angulaire et de fluidité
Contrôleur gauche	153	Statistiques sur la position, l'orientation, la vitesse et l'accélération du contrôleur gauche, ainsi que les états des boutons, gâchettes et trackpad
Contrôleur droit	146	Mêmes caractéristiques que le contrôleur gauche, appliquées au contrôleur droit
Regard (gaze)	206	Statistiques sur les directions du regard (gauche, droit, combiné) selon les axes locaux et globaux, ainsi que l'ouverture des yeux : entropie de mouvement, kurtosis, asymétrie, IQR et vitesse moyenne
Oculométrie	64	Diamètre pupillaire gauche et droit, position de la pupille dans l'espace capteur (x, y) : moyenne, écart-type, étendue, asymétrie, entropie de mouvement
Morphologique	29	Features dérivées : largeur des épaules (<code>shoulder_width</code>), ratio tête/épaules (<code>head_to_shoulder_ratio</code>), asymétrie d'activation des mains (<code>hand_asymmetry_speed</code>), avec leurs agrégats statistiques
Total	684	Après suppression de 64 colonnes constantes et 494 colonnes très corrélées ($r > 0.95$) sur les 1 242 features initiales

5.2.3 Agrégation par session

L'ensemble des caractéristiques extraites est ensuite agrégé à l'échelle de chaque session, identifiée par le couple `player_id/session_id`, conformément aux pratiques courantes en biométrie comportementale continue [139]. Cette étape permet de résumer l'ensemble des signaux associés à une session par un unique vecteur de caractéristiques de grande dimension, directement exploitable par les modèles de classification.

5.2.4 Sélection d'attributs

Afin de réduire la forte dimensionnalité du jeu de données (plusieurs centaines de variables dérivées après featurisation) et de limiter le risque de sur-apprentissage dans un contexte de faible nombre d'observations, une sélection supervisée des attributs a été intégrée directement au pipeline d'apprentissage.

Compte tenu du rapport défavorable entre le nombre de variables et la taille du jeu d'entraînement (situation dite $p \gg n$), une approche univariée supervisée a été privilégiée pour garantir la stabilité du processus de sélection.

La sélection des caractéristiques est réalisée à l'aide du sélecteur `SelectKBest` de la bibliothèque `scikit-learn` [140], combiné au test statistique ANOVA (`f_classif`) [141]. Cette méthode évalue individuellement chaque caractéristique en mesurant sa capacité à discriminer les classes à travers une statistique F.

Les variables sont ensuite classées selon leur score et seules les k meilleures caractéristiques sont conservées. La valeur de k n'est pas fixée a priori, mais optimisée au sein de la validation croisée imbriquée, permettant d'identifier la dimension la plus adaptée à chaque modèle tout en évitant une sélection excessive.

Contrairement aux approches *model-based* intégrées (embedded methods), cette stratégie présente l'avantage de garantir qu'un nombre fixe de caractéristiques est conservé à chaque itération, évitant ainsi les cas où aucune variable ne serait sélectionnée en présence de données bruitées ou de seuils trop stricts. Cette propriété est particulièrement importante dans un contexte de données comportementales de haute dimension et de taille d'échantillon limitée.

La sélection des attributs est intégrée au pipeline d'apprentissage et est réapprise à chaque itération de la validation croisée imbriquée, assurant ainsi une évaluation sans fuite de données et une estimation fiable des performances de généralisation ¹.

1. https://scikit-learn.org/stable/modules/feature_selection.html

5.2.5 Validation croisée imbriquée et prévention du surapprentissage

L'optimisation conjointe des hyperparamètres des modèles et du nombre de variables sélectionnées pose un risque important de surapprentissage lorsque l'évaluation des performances est réalisée sur les mêmes données que celles utilisées pour le tuning. Il a été démontré que ce type de procédure conduit à des estimations optimistes et biaisées des performances, en particulier lorsque la sélection d'attributs est intégrée au processus d'optimisation du modèle [142, 143].

Afin d'obtenir une estimation fiable et non biaisée des capacités de généralisation des modèles, une stratégie de **validation croisée imbriquée** (*nested cross-validation*) est adoptée. Cette approche, largement recommandée dans la littérature en apprentissage automatique [144, 145, 146, 147], permet de dissocier rigoureusement les phases d'optimisation des hyperparamètres et d'évaluation des performances.

La validation croisée imbriquée repose sur deux niveaux distincts de validation croisée :

- une **validation croisée interne**, dédiée à l'optimisation des hyperparamètres propres au modèle ainsi qu'au choix du nombre de variables sélectionnées (k) ;
- une **validation croisée externe**, utilisée exclusivement pour l'évaluation des performances, sur des données jamais vues lors de la phase d'optimisation.

Dans notre protocole expérimental, les deux niveaux de validation croisée sont implémentés à l'aide d'un schéma *GroupKFold* à 5 plis, garantissant une séparation stricte par joueur. Ce choix empêche toute fuite d'information entre sessions appartenant à un même individu, ce qui est essentiel dans un contexte biométrique et comportemental en réalité virtuelle.

Concrètement, pour chaque pli de la validation croisée externe, le jeu d'entraînement est soumis à une validation croisée interne au sein de laquelle une recherche aléatoire d'hyperparamètres (*RandomizedSearchCV*) est effectuée. Cette recherche inclut à la fois les hyperparamètres propres au classifieur et le paramètre k du sélecteur *SelectKBest*. À chaque itération interne, la sélection d'attributs et l'entraînement du modèle sont réalisés exclusivement sur les données d'entraînement du pli considéré, conformément aux recommandations de Cawley et Talbot [143].

Les performances rapportées pour la validation croisée externe correspondent ainsi à une évaluation honnête et indépendante de toute décision d'optimisation. Cette approche

permet de mesurer la stabilité des modèles, de quantifier la variance des performances entre plis et de limiter efficacement le risque de surapprentissage lié à la recherche d’hyperparamètres.

Enfin, une fois la phase de validation croisée imbriquée terminée, un entraînement final est réalisé sur l’ensemble du jeu d’apprentissage en utilisant les hyperparamètres optimaux sélectionnés. Les performances finales sont ensuite évaluées sur un jeu de test indépendant, strictement séparé par joueur, fournissant une estimation réaliste des performances en conditions opérationnelles.

L’ensemble du pipeline d’apprentissage, incluant la validation croisée imbriquée, la sélection d’attributs et l’optimisation des hyperparamètres, est implémenté à l’aide de la bibliothèque *scikit-learn*² et s’inspire des bonnes pratiques documentées dans des implémentations open-source de référence³.

5.2.6 Modèles classiques et résultats

Les vecteurs de caractéristiques statistiques extraits des signaux de mouvement sont utilisés pour entraîner plusieurs modèles de classification, incluant Random Forest, Gradient Boosting, XGBoost, LightGBM, SVM à noyau linéaire et RBF, ainsi qu’un perceptron multicouche implémenté avec *scikit-learn*. Ces modèles représentent des approches de référence largement utilisées pour l’analyse de données biométriques et comportementales, en raison de leur capacité à exploiter des représentations tabulaires de grande dimension tout en offrant un bon compromis entre performance et complexité [4, 5].

L’objectif de cette section est d’évaluer les performances des modèles classiques appliqués à une représentation tabulaire des signaux de mouvement en réalité virtuelle (RV). Ces modèles constituent une baseline essentielle avant l’étude des approches neuronales et temporelles.

Dans ce cadre, les séries temporelles brutes issues des dispositifs de suivi RV sont transformées en vecteurs de caractéristiques fixes à l’aide d’une étape d’extraction et d’agrégation. Pour chaque variable continue, un ensemble étendu de descripteurs statistiques et dynamiques est calculé (moyennes, variances, amplitudes, vitesses, asymétries, mesures d’énergie et de fluidité), complété par des indicateurs biomécaniques dérivés décrivant la posture et la coordination du mouvement. L’ensemble de ces informations

2. https://scikit-learn.org/stable/modules/cross_validation.html

3. <https://github.com/scikit-learn/scikit-learn>

est ensuite agrégé à l'échelle de chaque session, produisant un vecteur tabulaire unique par couple `player_id/session_id`.

Cette stratégie permet d'adapter automatiquement la dimensionalité effective de l'espace de représentation aux propriétés inductives de chaque famille de modèles. Les classifieurs à base d'arbres tendent ainsi à exploiter un nombre plus élevé de variables afin de capturer des relations non linéaires complexes, tandis que les modèles plus sensibles à la dimensionnalité, tels que les SVM à noyau RBF ou les réseaux de neurones multicouches, privilégient des sous-ensembles de caractéristiques plus compacts, favorisant la stabilité et la capacité de généralisation.

À l'issue de ce processus, six modèles d'apprentissage supervisé ont été évalués : Random Forest, Gradient Boosting, SVM à noyau radial (SVM RBF), XGBoost, LightGBM, ainsi qu'un perceptron multicouche (MLP) entraîné avec la bibliothèque *scikit-learn*. L'architecture du MLP est optimisée automatiquement par `RandomizedSearchCV` parmi quatre configurations candidates de couches cachées : (64, 32), (128, 64), (256, 128) et (128, 128, 64), combinées avec deux fonctions d'activation (`relu`, `tanh`), deux optimiseurs (`adam`, `lbfgs`), et une régularisation L2 de paramètre $\alpha \in [10^{-6}, 10^{-2}]$.

La sélection des attributs et le protocole de validation croisée imbriquée utilisés pour l'entraînement et l'optimisation des modèles sont décrits respectivement dans les sections 5.2.4 et 5.2.5. À l'issue de cette optimisation, les architectures et hyperparamètres retenus pour chaque modèle sont les suivants : Random Forest (191 arbres, profondeur maximale 12, 589 features), Gradient Boosting (221 estimateurs, taux d'apprentissage 0.122, 150 features), SVM à noyau RBF ($C = 4.76$, $\gamma = 0.0012$, 100 features), MLP à deux couches cachées (64, 32) avec activation `tanh` et optimiseur `lbfgs` ($\alpha = 5.34 \times 10^{-6}$, 150 features), XGBoost (410 estimateurs, profondeur 5, 50 features) et LightGBM (269 estimateurs, 33 feuilles, 75 features).

Après la procédure de validation croisée imbriquée, les hyperparamètres optimaux sont sélectionnés pour chaque modèle. Un modèle final est ensuite réentraîné sur l'ensemble complet des données d'entraînement en appliquant la même procédure de sélection d'attributs. Ce modèle final est finalement évalué sur le jeu de test indépendant, strictement séparé par joueur.

Les performances finales sur le jeu de test indépendant, strictement séparé par joueur, sont présentées dans la Table 5.4.

Les résultats mettent en évidence des performances globalement élevées pour l'ensemble des modèles évalués, tout en révélant des différences notables selon la famille de

TABLEAU 5.4 – Performances des modèles classiques après validation croisée imbriquée (sans attaques).

Modèle	k optimal	CV BAcc	BAcc	Precision	Recall	F1
Random Forest	589	0.80 ± 0.04	0.88	0.86	0.88	0.89
Gradient Boosting	150	0.79 ± 0.06	0.93	0.93	0.93	0.93
SVM (RBF)	100	0.87 ± 0.09	0.93	0.93	0.93	0.93
MLP	150	0.82 ± 0.05	0.84	0.84	0.84	0.84
XGBoost	50	0.76 ± 0.07	0.77	0.78	0.77	0.77
LightGBM	75	0.77 ± 0.08	0.86	0.87	0.86	0.86

classifieurs considérée. Les meilleures performances sur le jeu de test sont obtenues par GradientBoosting et SVM (RBF), qui atteignent tous deux une *balanced accuracy* de **0.93**, avec une precision et un recall respectivement de 0.93 et 0.93 pour GradientBoosting et 0.93 pour SVM. Ces résultats traduisent une excellente capacité de généralisation dans un contexte inter-joueurs, malgré la forte variabilité comportementale des données de mouvement en réalité virtuelle.

Le SVM à noyau RBF présente des performances élevées sur le jeu de test (BAcc= 0.93, F1= 0.93) tout en s'appuyant sur la sélection de features la plus parcimonieuse parmi tous les modèles évalués (100 features). Sa variabilité en validation croisée (0.87 ± 0.09) est la plus élevée parmi tous les modèles, illustrant une certaine instabilité inter-plis malgré ses excellentes performances sur le jeu de test. Le Random Forest présente le meilleur score de stabilité (0.80 ± 0.04), avec une faible variabilité inter-plis confirmant sa robustesse face à l'hétérogénéité inter-individuelle. Sa performance sur le jeu de test (BAcc= 0.88, F1= 0.89) reste inférieure à celle de GradientBoosting et SVM, suggérant une capacité de généralisation plus conservatrice. Il utilise le plus grand nombre de features (589 sur 684). XGBoost en revanche s'appuie sur seulement 50 features, atteignant BAcc= 0.77 et F1= 0.77 sur le jeu de test avec une variabilité en validation croisée de 0.76 ± 0.07 . Le MLP affiche des performances intermédiaires (BAcc= 0.84, F1= 0.84, 0.82 ± 0.05 en validation croisée). Son écart-type modéré suggère une bonne stabilité relative, mais ses performances en test restent inférieures à celles des modèles ensemblistes, avec une precision de 0.84 et un recall de 0.84 indiquant un bon équilibre entre les deux classes. LightGBM présente des performances correctes sur le jeu de test (BAcc= 0.86, F1= 0.86) avec un score de validation croisée de 0.77 ± 0.08 , indiquant une bonne cohérence entre validation et test.

Les écarts-types observés lors de la validation croisée externe varient de ± 0.04 (Random Forest) à ± 0.09 (SVM), indiquant une variabilité non négligeable entre les différents plis. Cette variabilité s'explique par la forte hétérogénéité inter-joueurs présente dans les données de mouvement en réalité virtuelle ainsi que par le nombre relativement limité de participants (75 joueurs, 52 en entraînement et 23 en test).

On observe également une différence notable entre les performances moyennes obtenues lors de la validation croisée (0.76–0.87) et celles mesurées sur le jeu de test indépendant (0.77–0.93). Dans un contexte comportant un nombre restreint d'utilisateurs et une variabilité comportementale élevée, la validation croisée peut produire des plis particulièrement difficiles à généraliser. Le jeu de test indépendant peut ainsi présenter une distribution légèrement différente, conduisant à des performances plus élevées pour certains modèles, notamment GradientBoosting et SVM.

Vérification de l'absence de surapprentissage : Pour s'assurer que les performances obtenues ne proviennent pas d'un surapprentissage lors de l'optimisation des hyperparamètres, la BAcc moyenne en validation croisée imbriquée (nested CV) a été comparée, pour chacun des six modèles, à la BAcc mesurée sur le jeu de test final. Cette comparaison est présentée dans la Figure 5.2.

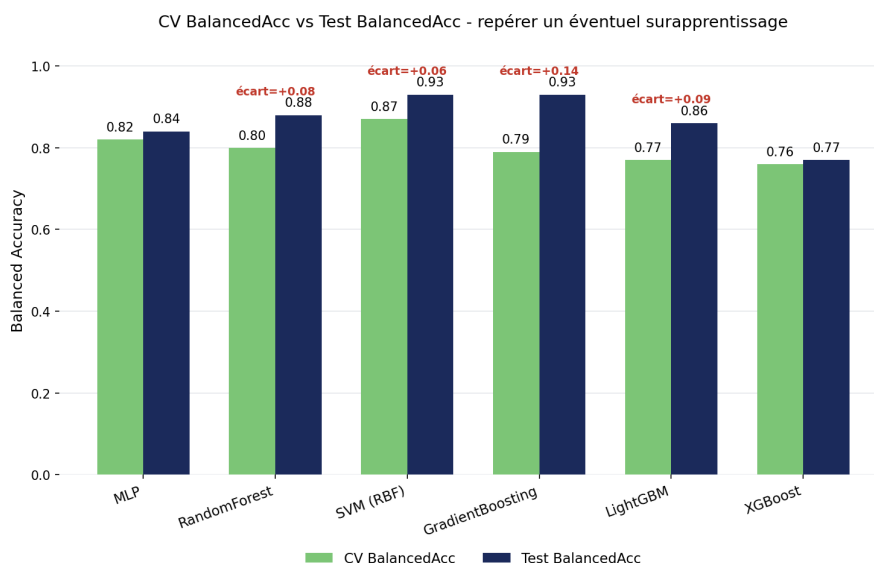


FIGURE 5.2 – Comparaison entre la BAcc moyenne en validation croisée imbriquée (CV BalancedAcc, avec écart-type) et la BAcc sur le jeu de test final (Test BalancedAcc), pour les six modèles évalués.

Les résultats ne révèlent pas un signe classique de surapprentissage, qui se manifesterait par une BAcc de validation croisée nettement supérieure à la BAcc de test. Le MLP et XGBoost présentent au contraire l'écart le plus faible (respectivement 0.02 et 0.01 point), confirmant une bonne cohérence entre validation croisée et test final. Pour Random Forest, SVM et LightGBM, un écart modéré est observé (entre 0.06 et 0.09 point), la BAcc de test étant supérieure à la BAcc de validation croisée ce comportement, inverse du surapprentissage classique, suggère plutôt une variance élevée entre les plis de validation croisée (notamment pour le SVM, avec un écart-type de ± 0.09) plutôt qu'un problème de généralisation du modèle. Le Gradient Boosting présente l'écart le plus important (0.14 point), ce qui appelle une vigilance particulière quant à la représentativité du jeu de test pour ce modèle, sans toutefois remettre en question la validité de la procédure d'évaluation par validation croisée imbriquée.

5.2.7 Expérimentation : split aléatoire

Afin d'évaluer l'impact du protocole de division des données sur les performances des modèles, une expérimentation complémentaire a été conduite en remplaçant le split par joueur par un split aléatoire stratifié (**StratifiedKFold**, $k = 5$). Dans ce protocole, des sessions d'un même joueur peuvent se retrouver simultanément dans le train et dans le test, contrairement au split par joueur qui garantit l'indépendance totale entre individus.

Le Tableau 5.5 présente les résultats obtenus avec ce protocole.

TABLEAU 5.5 – Performances des modèles avec split aléatoire stratifié (**StratifiedKFold**, $k = 5$). k désigne le nombre de features sélectionnées par **SelectKBest**.

Modèle	k	CV (moyenne $\pm$ écart-type)	BAcc	Precision	Recall	F1
Random Forest	400	0.86 $\pm$ 0.09	0.81	0.83	0.81	0.81
Gradient Boosting	150	0.82 $\pm$ 0.07	0.93	0.93	0.93	0.93
SVM (RBF)	100	0.89 $\pm$ 0.07	0.93	0.93	0.93	0.93
MLP	100	0.88 $\pm$ 0.05	0.80	0.80	0.80	0.80
XGBoost	589	0.83 $\pm$ 0.04	0.80	0.80	0.80	0.80
LightGBM	75	0.82 $\pm$ 0.08	0.86	0.87	0.86	0.86

Interprétation des résultats : Les résultats révèlent des performances globalement plus élevées en validation croisée qu'avec le split par joueur, avec des moyennes com-

prises entre 0.82 et 0.89. GradientBoosting et SVM (RBF) maintiennent leurs excellentes performances sur le jeu de test (BAcc = 0.93), confirmant leur robustesse indépendamment du protocole d'évaluation. LightGBM conserve également une performance stable (BAcc = 0.86).

En revanche, Random Forest subit une baisse notable ($0.88 \rightarrow 0.81$) malgré une meilleure CV moyenne (0.86), ce qui indique une sensibilité accrue à la composition des plis et une tendance à overfit sur les sessions de certains joueurs présents en train. MLP et XGBoost obtiennent tous deux BAcc = 0.80, performances inférieures à celles observées avec le split par joueur pour le MLP ($0.84 \rightarrow 0.80$), mais légèrement supérieures pour XGBoost ($0.77 \rightarrow 0.80$).

Comparaison avec le split par joueur : La comparaison du Tableau 5.6 met en évidence trois comportements distincts selon les modèles :

TABLEAU 5.6 – Comparaison des performances (BAcc) selon le protocole de split.

Modèle	Split joueur	Split aléatoire	Différence
Random Forest	0.88	0.81	-0.07
Gradient Boosting	0.93	0.93	0.00
SVM (RBF)	0.93	0.93	0.00
MLP	0.84	0.80	-0.04
XGBoost	0.77	0.80	+0.03
LightGBM	0.86	0.86	0.00

GradientBoosting, SVM, LightGBM : Ces modèles maintiennent des performances identiques dans les deux protocoles (BAcc = 0.93, 0.93 et 0.86 respectivement), témoignant d'une généralisation robuste indépendante de la stratégie de split. Ils capturent des patterns comportementaux de genre suffisamment stables entre sessions pour ne pas être affectés par le choix du protocole d'évaluation.

Random Forest, MLP : Ces deux modèles voient leurs performances baisser avec le split aléatoire (Random Forest : $0.88 \rightarrow 0.81$, MLP : $0.84 \rightarrow 0.80$). Ce résultat est contre-intuitif : on attendrait normalement que le split aléatoire produise de meilleures performances, puisqu'il permet au modèle de voir plusieurs sessions d'un même joueur pendant l'entraînement.

La raison est que le jeu de test est identique dans les deux cas (23 joueurs distincts, jamais vus). La différence ne vient donc pas du test lui-même, mais de la sélection des

hyperparamètres lors de la validation croisée interne : le split par joueur, en imposant l'indépendance inter-individuelle à chaque pli, guide la recherche vers des hyperparamètres qui généralisent mieux à de nouveaux utilisateurs. Le split aléatoire, en mélangeant les sessions d'un même joueur entre les plis, sélectionne des hyperparamètres optimisés pour une distribution légèrement différente de celle du jeu de test, produisant des configurations sous-optimales pour la généralisation inter-joueurs. Cela explique notamment la différence de features sélectionnées : $k = 589$ (split joueur) contre $k = 400$ (split aléatoire) pour Random Forest, et $k = 150$ contre $k = 100$ pour MLP.

XGBoost est le seul modèle à s'améliorer légèrement avec le split aléatoire ($0.77 \rightarrow 0.80, +0.03$). Contrairement aux autres modèles, le split aléatoire lui permet de voir une plus grande diversité de comportements intra-individuels pendant l'entraînement, ce qui améliore sa capacité à capturer des patterns robustes. Avec le split par joueur, il était davantage contraint par la séparation stricte inter-individuelle, le pénalisant légèrement.

Ces résultats mettent en évidence une observation importante sur la nature des données comportementales en réalité virtuelle : les sessions d'un même joueur présentent une variabilité intra-individuelle significative, au point que le protocole de split influence directement la qualité des hyperparamètres sélectionnés et les performances finales.

Cette variabilité inter-sessions suggère que chaque session de jeu constitue un échantillon comportemental partiellement indépendant : un même joueur ne reproduit pas exactement les mêmes patterns moteurs d'une session à l'autre, en raison de facteurs contextuels tels que la fatigue, l'humeur, le niveau d'expérience acquis ou les conditions d'utilisation du casque RV. Les signatures comportementales de genre restent globalement stables, mais leur expression précise varie suffisamment entre sessions pour que les modèles les plus sensibles (Random Forest, MLP) soient affectés par le choix du protocole d'évaluation.

Cette indépendance partielle entre sessions justifie et renforce le choix du split par joueur comme protocole de référence : si les sessions étaient parfaitement identiques d'un individu à l'autre, les deux protocoles produiraient les mêmes résultats. La différence observée confirme que la variabilité inter-sessions est réelle et non négligeable, et que le split par joueur est le seul protocole capable d'évaluer honnêtement la généralisation à de nouveaux individus dans ce contexte.

5.3 Représentation temporelle des signaux de mouvement

À l'inverse de la représentation tabulaire, qui condense chaque session en un vecteur de caractéristiques statiques, la représentation temporelle exploite explicitement la structure séquentielle des signaux de mouvement issus de la réalité virtuelle. Cette approche permet de préserver l'évolution temporelle des trajectoires, des vitesses et des interactions, et s'avère particulièrement adaptée à la modélisation du mouvement humain continu, où la dynamique intra-session incluant les variations de vitesse, les saccades et la coordination motrice constitue un facteur discriminant majeur [40, 43].

5.3.1 Caractéristiques dynamiques dérivées

Dans le cadre de la représentation temporelle, les signaux de mouvement ne sont pas résumés par des statistiques globales à l'échelle d'une session. À l'inverse, les modèles temporels opèrent directement sur des séquences multivariées où chaque pas de temps correspond à un ensemble de caractéristiques calculées *frame-par-frame*. Cette approche vise à préserver la dynamique fine du mouvement humain et les variations locales des signaux, essentielles pour la modélisation des comportements continus en réalité virtuelle.

À partir des positions brutes des contrôleurs et des signaux d'eye-tracking, plusieurs dérivées temporelles sont calculées par session. Les **vitesse**s instantanées sont estimées par différences finies normalisées par l'intervalle temporel :

$$v_{k,\alpha}^{(h)} = \frac{p_{k,\alpha}^{(h)} - p_{k-1,\alpha}^{(h)}}{\Delta t_k} \quad (5.2)$$

où $h \in \{\text{left}, \text{right}\}$ désigne le contrôleur, $\alpha \in \{x, y, z\}$ l'axe spatial, k le pas de temps courant, et Δt_k l'intervalle temporel en millisecondes entre deux frames consécutives.

Les **accélérations** correspondent aux différences successives de vitesses :

$$a_{k,\alpha}^{(h)} = v_{k,\alpha}^{(h)} - v_{k-1,\alpha}^{(h)} \quad (5.3)$$

où $v_{k,\alpha}^{(h)}$ est la vitesse du contrôleur h selon l'axe α au pas de temps k .

Le **jerk** capture les changements brusques de mouvement en mesurant la variation de l'accélération :

$$\dot{j}_{k,\alpha}^{(h)} = a_{k,\alpha}^{(h)} - a_{k-1,\alpha}^{(h)} \quad (5.4)$$

où $a_{k,\alpha}^{(h)}$ est l'accélération du contrôleur h selon l'axe α au pas de temps k .

Les **normes** euclidiennes des vecteurs de vitesse et d'accélération caractérisent l'intensité instantanée du mouvement indépendamment de sa direction :

$$\|v_k^{(h)}\| = \sqrt{\left(v_{k,x}^{(h)}\right)^2 + \left(v_{k,y}^{(h)}\right)^2 + \left(v_{k,z}^{(h)}\right)^2} \quad (5.5)$$

$$\|a_k^{(h)}\| = \sqrt{\left(a_{k,x}^{(h)}\right)^2 + \left(a_{k,y}^{(h)}\right)^2 + \left(a_{k,z}^{(h)}\right)^2} \quad (5.6)$$

où $v_{k,\alpha}^{(h)}$ et $a_{k,\alpha}^{(h)}$ désignent respectivement la vitesse et l'accélération du contrôleur h selon l'axe α au pas de temps k .

La **vitesse du regard** est calculée par différences finies sur les composantes de la direction du regard, et sa norme est définie par :

$$\|v_k^{\text{gaze}}\| = \sqrt{\left(v_{k,x}^{\text{gaze}}\right)^2 + \left(v_{k,y}^{\text{gaze}}\right)^2 + \left(v_{k,z}^{\text{gaze}}\right)^2} \quad (5.7)$$

où $v_{k,\alpha}^{\text{gaze}}$ est la vitesse de la direction du regard selon l'axe α au pas de temps k , obtenue par différences finies sur le signal `gaze_direction_right_α`.

Afin de modéliser les **asymétries motrices**, le ratio normalisé de vitesse entre les deux mains est calculé sur chaque axe :

$$r_{k,\alpha}^{\text{speed}} = \frac{v_{k,\alpha}^L - v_{k,\alpha}^R}{v_{k,\alpha}^L + v_{k,\alpha}^R + \varepsilon} \quad (5.8)$$

où $v_{k,\alpha}^L$ et $v_{k,\alpha}^R$ sont les vitesses de la main gauche et droite selon l'axe α , et $\varepsilon = 10^{-6}$ est un terme de stabilisation numérique.

La **symétrie globale** des normes de vitesse entre les deux mains est définie par :

$$s_k^{\text{speed}} = \frac{\|v_k^L\| - \|v_k^R\|}{\|v_k^L\| + \|v_k^R\| + \varepsilon} \quad (5.9)$$

où $\|v_k^L\|$ et $\|v_k^R\|$ désignent les normes de vitesse de la main gauche et droite au pas de temps k .

Les asymétries brutes entre les deux mains sont également conservées :

$$\Delta v_k = \|v_k^R\| - \|v_k^L\|, \quad \Delta a_k = \|a_k^R\| - \|a_k^L\| \quad (5.10)$$

où $\|v_k^R\|$, $\|v_k^L\|$, $\|a_k^R\|$ et $\|a_k^L\|$ désignent respectivement les normes de vitesse et d'accélération de la main droite et gauche au pas de temps k .

Une **saccade** est détectée lorsque la norme de vitesse du regard dépasse un seuil fixé :

$$\mu_k^{\text{saccade}} = \mathbf{1} [\|v_k^{\text{gaze}}\| > \tau_s] \quad (5.11)$$

où $\tau_s = 0.05$ est le seuil de détection de saccade, déterminé empiriquement à partir de la distribution de $\|v_k^{\text{gaze}}\|$ dans le dataset.

Un **clignement** est détecté pour chaque œil lorsque son degré d'ouverture passe sous un seuil minimal :

$$\mu_k^{\text{blink},e} = \mathbf{1} [\text{openness}_k^e < \tau_b] \quad (5.12)$$

où $e \in \{\text{left}, \text{right}\}$ désigne l'œil considéré, et $\tau_b = 0.2$ est le seuil d'ouverture minimale en dessous duquel un clignement est détecté.

Les **dérivées temporelles** et la **variabilité locale** des diamètres pupillaires sont calculées pour chaque œil e :

$$d_k^e = d_k^e - d_{k-1}^e, \quad \sigma_k^e = \text{std}_{w=15} (d_{k-w+1}^e, \dots, d_k^e) \quad (5.13)$$

où d_k^e est le diamètre pupillaire de l'œil e au pas de temps k , et $w = 15$ est la taille de la fenêtre glissante.

L'**écart-type glissant** de la vitesse verticale de la main droite est défini par :

$$\sigma_k^{\text{roll}} = \text{std}_{w=15} (v_{k-w+1,y}^R, \dots, v_{k,y}^R) \quad (5.14)$$

où $v_{k,y}^R$ est la vitesse verticale de la main droite au pas de temps k , et $w = 15$ est la taille de la fenêtre glissante avec un minimum de 1 observation.

La **corrélation glissante** entre les vitesses verticales des deux mains est calculée sur une fenêtre de $w = 15$ pas de temps :

$$\rho_k^{\text{hands}} = \text{corr}_{w=15} (v_{k,y}^L, v_{k,y}^R) \quad (5.15)$$

où $v_{k,y}^L$ et $v_{k,y}^R$ sont les vitesses verticales de la main gauche et droite, calculées sur une fenêtre glissante de $w = 15$ pas de temps avec un minimum de 5 observations.

La **coordination œil-main** est modélisée par la corrélation glissante entre le signal de saccade et l'activité de la main droite :

$$\rho_k^{\text{eye-hand}} = \text{corr}_{w=15} \left(\mathbb{I}_k^{\text{saccade}}, \mathbb{I}_k^{\text{move,R}} \right) \quad (5.16)$$

où $\mathbb{I}_k^{\text{move,R}} = \mathbf{1} [\|v_k^R\| > 0.05]$ indique un mouvement actif de la main droite, et $w = 15$ est la taille de la fenêtre glissante avec un minimum de 5 observations.

La **largeur approximative des épaules** est estimée par la distance inter-contrôleurs sur l'axe x :

$$w_k^{\text{shoulder}} = p_{k,x}^R - p_{k,x}^L \quad (5.17)$$

où $p_{k,x}^R$ et $p_{k,x}^L$ sont les positions horizontales de la main droite et gauche au pas de temps k .

Le **ratio tête/épaules**, utilisé comme proxy de la taille corporelle, est défini par :

$$r_k^{\text{head/shoulder}} = \frac{p_{k,y}^{\text{hmd}}}{w_k^{\text{shoulder}} + \varepsilon} \quad (5.18)$$

où $p_{k,y}^{\text{hmd}}$ est la hauteur du casque au pas de temps k , w_k^{shoulder} la largeur estimée des épaules, et $\varepsilon = 10^{-6}$ un terme de stabilisation numérique.

Enfin, la **fréquence dominante** du mouvement vertical de la main droite est extraite par session via une transformée de Fourier rapide (FFT) appliquée au signal centré sur sa moyenne :

$$f_{\text{session}}^* = \arg \max_{f > 0} \mathcal{F} \{ v_{\cdot,y}^R - \bar{v}_{\cdot,y}^R \} (f) \quad (5.19)$$

où $\mathcal{F}$ désigne la transformée de Fourier discrète, $\bar{v}_{\cdot,y}^R$ la moyenne de la vitesse verticale sur la session, et $f_s = 90$ Hz la fréquence d'échantillonnage du dispositif RV. Cette valeur est ensuite répliquée à chaque pas de temps de la session correspondante.

Cette extraction dynamique est cohérente avec les travaux montrant que les modèles temporels exploitent principalement les variations micro-cinématiques et les patrons dynamiques locaux du mouvement, plutôt que des amplitudes globales agrégées [40].

5.3.2 Génération des séquences

Les données nettoyées sont d’abord découpées en fenêtres temporelles de taille fixe $\text{SEQ_LEN} = 100$ avec un chevauchement contrôlé ($\text{STEP} = 25$), conformément aux protocoles couramment utilisés pour les modèles 1D, CNN-LSTM et Transformers [40, 43]. Chaque séquence est extraite au sein d’un seul joueur, ce qui garantit l’absence de fuite d’identité entre les séquences. Une séquence est donc une matrice $(T \times F)$ regroupant T pas de temps et F features dynamiques.

5.3.3 Normalisation des séquences

Les données utilisées pour les modèles temporels sont représentées sous forme de tenseurs tridimensionnels de dimension (N, T, F) , où N désigne le nombre de séquences (sessions ou échantillons), T la longueur temporelle de chaque séquence et F le nombre de variables mesurées à chaque pas de temps.

Étant donné l’importance des différences d’échelle entre variables, une normalisation stricte est appliquée *feature-wise*. Pour ce faire, le tenseur est conceptuellement aplati selon les dimensions temporelles :

$$(N, T, F) \rightarrow (N \cdot T, F)$$

Cette transformation permet d’appliquer une normalisation indépendante pour chaque variable, en estimant la moyenne et l’écart-type sur l’ensemble des observations temporelles.

Dans l’implémentation, cette opération est réalisée de manière incrémentale (par blocs) afin de limiter l’usage mémoire, tout en restant équivalente à un aplatissement global des séquences.

Après transformation, les données sont ensuite remises dans leur forme séquentielle initiale par une opération de *reshape* inverse.

Ce procédé est recommandé dans les pipelines adversariaux séquentiels [17, 18], car il évite qu’un seul canal domine l’ensemble des gradients.

5.3.4 Résultats des modèles temporels

Afin d’évaluer la capacité des architectures séquentielles à capturer les dynamiques fines du mouvement en réalité virtuelle, plusieurs modèles temporels ont été entraînés

sur les séquences extraites des données de tracking. Les séquences, d'une longueur fixe de 100 pas temporels, sont construites indépendamment pour chaque joueur, de manière à garantir une séparation stricte des identités lors du découpage Train/Validation/Test. Cette contrainte est essentielle afin d'éviter toute fuite d'information et d'évaluer la capacité de généralisation des modèles à des utilisateurs jamais observés durant l'entraînement.

Afin d'assurer la comparabilité avec les modèles tabulaires présentés dans la section précédente, les mêmes principes méthodologiques sont appliqués dans ce cadre expérimental. En particulier, les procédures de sélection d'attributs et de validation croisée imbriquée décrites dans les sections 5.2.4 et 5.2.5 sont également utilisées pour l'optimisation des modèles temporels.

Plusieurs architectures spécialisées dans l'analyse de séries temporelles multivariées ont été évaluées :

- **LSTM bidirectionnel amélioré**, intégrant une normalisation de type *Layer-Norm*, du *dropout*, un *gradient clipping*, un *learning rate scheduler* et un mécanisme d'*early stopping*, afin de modéliser les dépendances temporelles longues tout en stabilisant l'apprentissage.
- **CNN-LSTM**, combinant une étape d'extraction locale des motifs temporels par convolution (CNN) suivie d'une modélisation séquentielle globale par LSTM. La couche convolutive 1D applique 64 filtres de taille 5 (avec un padding de 2 pour préserver la longueur de séquence), suivie d'une normalisation par lot (*batch normalization*) et d'une activation ReLU, afin d'extraire des motifs locaux (variations rapides de posture, micro-mouvements) indépendamment de leur position dans la séquence. Ces représentations locales sont ensuite transmises à une couche LSTM à 128 unités cachées, qui modélise les dépendances temporelles à plus long terme entre les motifs successifs. La classification finale est réalisée par une couche entièrement connectée appliquée sur le dernier état caché de la séquence, précédée d'un *dropout* de 0.3 pour la régularisation.
- **ResNet1D**, une adaptation des architectures résiduelles profondes aux séries temporelles, permettant de capturer efficacement des motifs hiérarchiques dans les signaux [40].
- **TimesNet**, une architecture récente basée sur une modélisation variationnelle 2D des séries temporelles, introduite dans [43].

Ces architectures permettent de capturer simultanément la structure temporelle des signaux ainsi que les interactions entre les différents canaux de mouvement (tête, mains et regard). Les performances obtenues par ces différentes architectures sont analysées ci-dessous. La Figure 5.3 présente une comparaison synthétique des performances obtenues par l'ensemble des architectures temporelles.

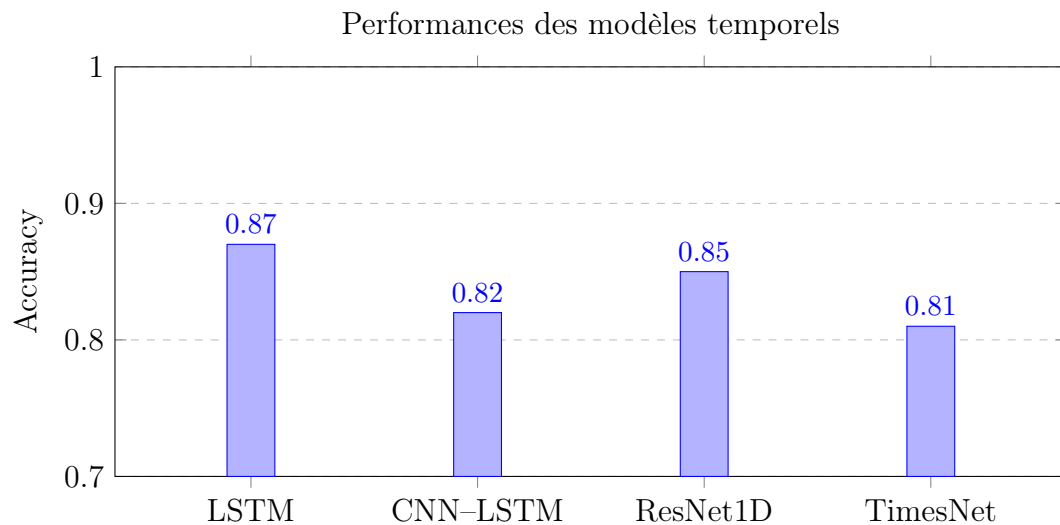


FIGURE 5.3 – Accuracy des différents modèles temporels sur le jeu de test.

LSTM bi-couche : Le modèle LSTM comporte deux couches empilées, associées à une *LayerNorm* stabilisant la distribution des activations et à un *dropout* interne réduisant le surapprentissage. Il atteint une accuracy d'environ **87 %** sur le jeu de test. Ce modèle montre une convergence stable et une capacité notable à extraire des régularités temporelles globales : transitions dans les amplitudes de mouvement, cycles de déplacement des mains, rythmes oculaires et variations progressives de posture. Les résultats suggèrent que les dépendances temporelles à moyen terme (50–100 pas) constituent un signal pertinent pour discriminer le genre en RV.

CNN-LSTM et ResNet1D : Les architectures hybrides, combinant des convolutions 1D pour extraire des motifs locaux suivies d'un traitement séquentiel (LSTM ou blocs résiduels), observent des performances comprises entre **82 % et 85 %**. Les couches convolutionnelles détectent efficacement de micro-motifs comportementaux (accélérations brusques, pics de vitesse, micro-saccades), mais elles peinent à capturer les

relations temporelles plus longues qui caractérisent les styles moteurs individuels. La performance légèrement inférieure au LSTM pur indique que les signaux RV étudiés présentent une structure temporelle longue relativement importante, difficile à exploiter par des architectures centrées sur le local.

TimesNet : Le modèle TimesNet, basé sur des blocs d'inception fréquentielle extrayant les composantes périodiques du signal, obtient une accuracy d'environ **80 %**. Cette performance s'explique par la nature des données RV : les gestes captés sont fortement hétérogènes d'un joueur à l'autre et ne présentent que peu de cycles répétitifs. Le modèle capte efficacement des oscillations de vitesse ou des patterns rythmiques, mais il est moins adapté à des signaux dont la dynamique repose davantage sur la variabilité gestuelle que sur la périodicité.

De manière générale, les résultats obtenus dans ce travail indiquent que, dans ce contexte expérimental, la représentation tabulaire agrégée produit des performances légèrement supérieures aux architectures temporelles. Cette observation peut s'expliquer par la forte variabilité inter-individuelle des mouvements en réalité virtuelle ainsi que par la diversité des styles moteurs observés entre joueurs. Dans ce contexte, l'agrégation statistique des signaux semble fournir une représentation plus stable des comportements, tandis que les architectures séquentielles restent plus sensibles à l'hétérogénéité locale des séquences et à l'absence de schémas temporels fortement répétitifs dans les données étudiées.

5.4 Comparaison des modèles

Afin d'établir une première évaluation des capacités prédictives des différentes familles de modèles, une comparaison systématique a été menée entre les approches tabulaires classiques, les modèles neuronaux fully-connected, ainsi que les architectures temporelles présentées au Chapitre 4. Toutes les expériences ont été réalisées selon un protocole strictement séparé par joueur afin d'éviter toute fuite d'identité et garantir une estimation réaliste de la performance en contexte inter-individuel. Le Tableau 5.7 présente une synthèse des performances sur le jeu de test pour l'ensemble des modèles évalués, incluant des approches classiques basées sur des représentations tabulaires, un réseau de neurones multicouche, ainsi que plusieurs architectures temporelles dédiées à l'analyse de séries temporelles.

TABLEAU 5.7 – Performances comparées des modèles classiques, neuronaux et temporels.

Modèle	Balanced accuracy	Commentaires
Random Forest	0.88	Modèle d'ensemble stable, présentant une bonne cohérence entre validation croisée (0.80 ± 0.04) et jeu de test. Precision= 0.90, Recall= 0.88, F1= 0.89.
Gradient Boosting	0.93	Meilleures performances sur le jeu de test parmi les modèles classiques, avec une validation croisée de (0.79 ± 0.06). Precision= 0.93, Recall= 0.93, F1= 0.93.
XGBoost	0.77	Performances plus faibles malgré le plus petit nombre de features (50), avec une variabilité inter-plis notable (0.76 ± 0.07). Precision= 0.78, Recall= 0.77, F1= 0.77.
LightGBM	0.86	Bonnes performances sur le jeu de test avec un score de validation croisée de (0.77 ± 0.08). Precision= 0.87, Recall= 0.86, F1= 0.86.
SVM (RBF)	0.93	Meilleures performances sur le jeu de test avec le plus faible nombre de features (100), et variabilité en validation croisée (0.87 ± 0.09). Precision= 0.93, Recall= 0.93, F1= 0.93.
MLP	0.84	Performances intermédiaires avec une bonne stabilité (0.82 ± 0.05), mais généralisation inférieure aux modèles ensemblistes. Precision= 0.841, Recall= 0.842, F1= 0.841.
LSTM bi-couche	0.87	Capture bien les dynamiques globales
CNN-LSTM	0.82	Sensible aux variations intra-séquence
ResNet1D	0.85	Bon extracteur local mais limité en long-terme
TimesNet	0.81	Capture les fréquences mais limité par la variabilité

Les performances des modèles sont évaluées à l'aide de plusieurs métriques complémentaires, incluant la *balanced accuracy* moyenne obtenue en validation croisée imbriquée, l'écart-type associé, ainsi que la précision, le rappel et le F1-score mesurés sur le jeu de test indépendant. La validation croisée imbriquée permet d'estimer la capacité de généralisation des modèles dans un cadre inter-joueurs, tandis que l'évaluation sur le jeu de test, strictement séparé par joueur, fournit une mesure réaliste du comportement des modèles dans des conditions proches de l'usage opérationnel.

Les résultats mettent en évidence des performances globalement élevées pour l'ensemble des modèles évalués, tout en révélant des différences notables selon la famille de classifieurs considérée. Les modèles GradientBoosting et SVM (RBF) se distinguent par leurs meilleures performances sur le jeu de test, atteignant tous deux une *balanced accu-*

racy de **0.93**. Le GradientBoosting présente une excellente cohérence entre les métriques (Precision = 0.93, Recall = 0.93, F1 = 0.93) avec une bonne stabilité en validation croisée (0.79 ± 0.06). Le SVM atteint le même niveau de performance (Precision = 0.93, Recall = 0.93, F1 = 0.93) avec seulement 100 features sélectionnées, illustrant l’efficacité d’une réduction parcimonieuse de la dimensionnalité (0.87 ± 0.09).

Le Random Forest présente une bonne stabilité globale (0.80 ± 0.04), avec une performance sur le jeu de test de BAcc = 0.88 (Precision = 0.90, Recall = 0.88, F1 = 0.89), légèrement inférieure à celle de GradientBoosting et SVM.

LightGBM présente des performances correctes sur le jeu de test (BAcc = 0.86, Precision = 0.87, Recall = 0.86, F1 = 0.86) avec un score de validation croisée de 0.77 ± 0.08 .

XGBoost présente les performances les plus faibles parmi les modèles classiques, avec BAcc = 0.77 sur le jeu de test (Precision = 0.78, Recall = 0.77, F1 = 0.77) et une variabilité inter-plis notable (0.76 ± 0.07), malgré un nombre de features relativement faible (50 sur 684). Le MLP atteint BAcc = 0.84 (Precision = 0.84, Recall = 0.84, F1 = 0.84) avec une stabilité acceptable (0.82 ± 0.05), mais ses performances restent inférieures à celles des modèles ensemblistes.

Les architectures temporelles présentent des performances globalement inférieures à celles obtenues par les modèles tabulaires. Le modèle *LSTM bi-couche* atteint une *balanced accuracy* d’environ **0.87**, indiquant une capacité correcte à exploiter les dynamiques temporelles du mouvement. Les architectures hybrides telles que *CNN-LSTM* obtiennent des performances légèrement plus faibles (**0.82**), suggérant que les motifs locaux extraits par les convolutions ne suffisent pas à capturer pleinement les dépendances temporelles longues présentes dans les données.

Les modèles *ResNet1D* et *TimesNet* obtiennent respectivement des *balanced accuracies* d’environ **0.85** et **0.81**. Bien que ces architectures soient capables d’extraire des motifs temporels complexes, leurs performances restent limitées dans ce contexte expérimental. Cette observation peut s’expliquer par la forte variabilité inter-individuelle des mouvements en réalité virtuelle ainsi que par l’absence de structures temporelles fortement répétitives dans les séquences étudiées.

5.5 Analyse des variables les plus discriminantes

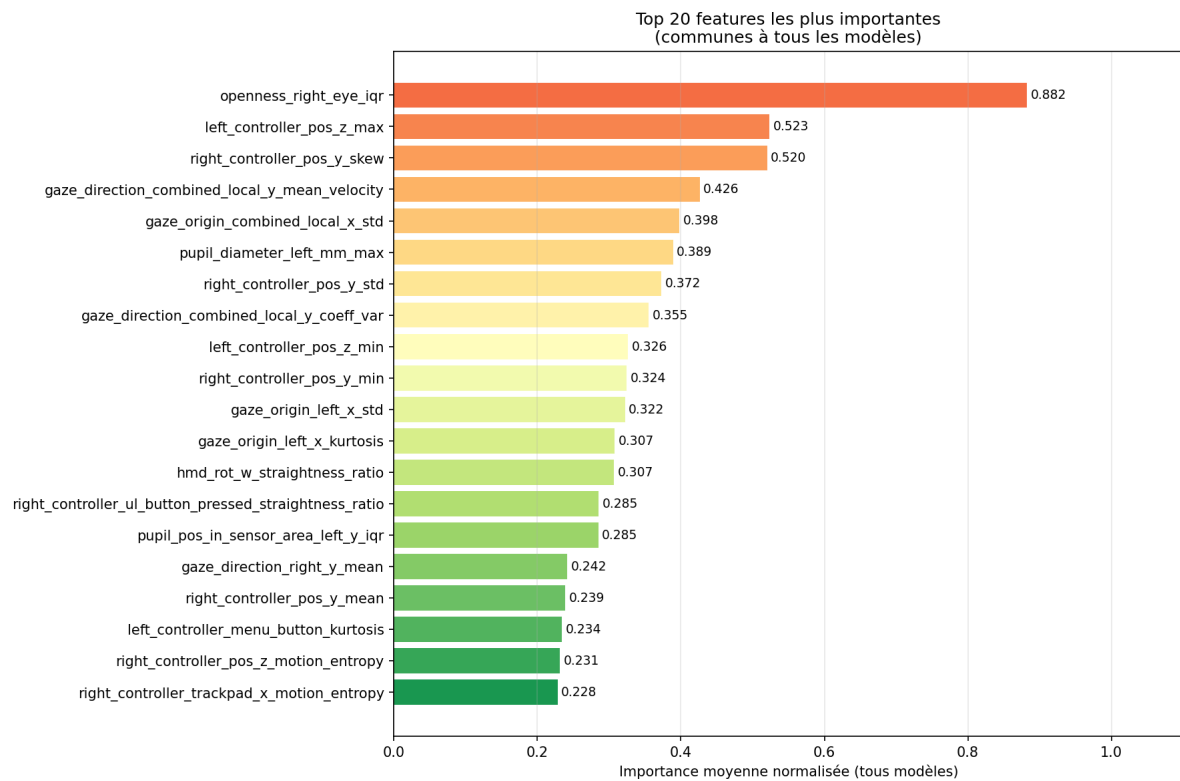


FIGURE 5.4 – Top 20 features les plus importantes en moyenne sur tous les modèles.

Le graphique 5.4 montre les 20 variables les plus importantes communes à tous les modèles testés. On voit que la variable `openness_right_eye_iqr` (qui mesure la variation de l'ouverture de l'œil droit) sort du lot avec un score de 0,882, presque deux fois plus élevé que la deuxième variable, `left_controller_pos_z_max` (0,523). Cela veut dire que cette variable liée à l'œil est de loin la plus utile pour faire les prédictions. Juste derrière, on retrouve des variables liées à la position des manettes (`right_controller_pos_y_skew`, 0,520) et au mouvement du regard (`gaze_direction_combined_local_y_mean_velocity`, 0,426), qui restent assez importantes mais moins que la première. Pour le reste des variables (du 6^e au 20^e rang), les scores baissent doucement et restent proches les uns des autres, entre 0,23 et 0,40. Cela montre qu'après la variable de l'œil qui domine clairement, les autres variables (position des manettes, orientation du casque, mouvement de la souris, boutons pressés) contribuent chacune un peu, sans qu'aucune ne se démarque vraiment.

5.6 Conclusion du chapitre

L'analyse menée dans ce chapitre avait pour objectif d'évaluer dans quelle mesure les données de mouvement issues de la réalité virtuelle constituent une source d'information pertinente pour la classification du genre des utilisateurs à l'aide de modèles d'apprentissage automatique. À partir des jeux de données préparés et des représentations définies au chapitre précédent, plusieurs familles de modèles ont été étudiées, incluant des approches tabulaires, des modèles neuronaux fully-connected et des architectures temporelles capables de prendre en compte la dynamique des signaux.

Des travaux récents ont déjà montré que les données de mouvement issues de la réalité virtuelle peuvent être utilisées pour inférer certains attributs personnels, notamment le genre des utilisateurs. Dans ce contexte, la contribution principale de ce chapitre ne consiste pas à démontrer l'existence de cette information discriminante, mais plutôt à proposer une analyse comparative systématique des différentes familles de modèles susceptibles d'exploiter ces données.

Les résultats obtenus montrent que plusieurs approches d'apprentissage automatique permettent d'atteindre des performances élevées dans ce contexte. Les modèles tabulaires reposant sur des représentations agrégées des signaux temporels se révèlent particulièrement efficaces, atteignant des performances comparables, voire légèrement supérieures, à celles obtenues par des modèles neuronaux plus complexes. Les architectures temporelles entraînées directement sur les séries de mouvement présentent des performances globalement plus modérées, bien qu'un modèle LSTM atteigne néanmoins des résultats proches de **90 %** de *balanced accuracy*. Ces observations mettent en évidence l'importance du choix de la représentation des données, et montrent que l'agrégation statistique des signaux peut constituer une stratégie simple mais très performante pour ce type de tâche.

Dans une perspective applicative, par exemple dans le cadre d'un système d'authentification en réalité virtuelle reposant sur l'analyse comportementale, ces résultats suggèrent qu'il n'existe pas nécessairement un choix unique et évident de modèle à déployer. Plusieurs familles de classifieurs qu'ils soient basés sur des méthodes classiques ou sur des architectures neuronales peuvent raisonnablement être utilisées pour réaliser cette tâche avec des performances comparables.

En réponse à **RQ1**, les résultats obtenus confirment que les données de mouvement issues de la réalité virtuelle constituent bien une source d'information pertinente pour la

classification du genre. Plusieurs modèles atteignent des performances élevées en conditions nominales, notamment GradientBoosting et SVM avec une *balanced accuracy* de 0.93.

Cette observation a également des implications importantes du point de vue de la sécurité. Un utilisateur cherchant à tromper un système de classification du genre ne peut pas supposer à l'avance quel type de modèle est utilisé en arrière-plan. Cette incertitude justifie l'approche adoptée dans le chapitre suivant, qui consiste à étudier la robustesse de plusieurs familles de classifieurs face à différentes perturbations comportementales appliquées aux signaux de mouvement en réalité virtuelle.

Chapitre 6

Menaces et perturbations sur la reconnaissance du genre en RV

Après avoir établi, au chapitre précédent, que les données de mouvement issues de la réalité virtuelle permettent une classification fiable du genre dans un cadre non perturbé, ce chapitre s'intéresse à l'analyse de la vulnérabilité de ces modèles face à des scénarios de perturbation comportementale et contextuelle en environnement de réalité virtuelle. L'objectif est d'évaluer dans quelle mesure les performances et la fiabilité des modèles de reconnaissance du genre peuvent être dégradées lorsque les signaux de mouvement sont intentionnellement altérés.

Contrairement aux données statiques classiquement étudiées dans la littérature sur les attaques adversariales, les signaux de mouvement en réalité virtuelle présentent des caractéristiques spécifiques : ils sont temporels, continus, multimodaux, et limités par ce que le corps humain peut physiquement faire. Ces propriétés ouvrent la voie à des formes d'attaques distinctes, qui ne se limitent pas à des perturbations numériques de faible amplitude, mais incluent également des modifications comportementales plausibles, telles que des variations de posture, de dynamique gestuelle ou de coordination œil-main.

Dans ce chapitre, différents types de perturbations comportementales sont ainsi étudiés, incluant des attaques numériques appliquées directement aux signaux, ainsi que des attaques comportementales visant à simuler des variations réalistes du mouvement pouvant être produites par un utilisateur réel. Ces attaques sont appliquées de manière systématique aux modèles tabulaires, neuronaux et temporels évalués précédemment, afin d'analyser leur sensibilité relative et d'identifier les vulnérabilités propres à chaque famille de modèles.

L'analyse menée vise à répondre à la deuxième question de recherche de ce mémoire : quel est l'impact des perturbations comportementales, numériques et comportementales, sur les performances et la fiabilité des modèles de reconnaissance du genre en réalité virtuelle.

6.1 Calibration des paramètres des perturbations

La conception des perturbations comportementales repose sur une approche méthodique en trois étapes : (i) identification des familles de perturbations selon leur nature physique, (ii) choix de la distribution probabiliste adaptée à chaque famille, et (iii) calibration des amplitudes sur les statistiques descriptives du dataset d'entraînement.

6.1.1 Identification des familles de perturbations

Les perturbations sont regroupées en quatre familles selon leur mécanisme d'action sur les features :

- **Perturbations additives** : ajout d'un terme δ constant ou aléatoire aux features cibles. Exemples : `posture_head_lower` ($\delta = 0.25$), `sensor_offset` ($\delta = 0.08$), `calibration_bias` ($\delta = 0.03$), `jitter` ($\sigma = 0.10$), `sensor_noise_burst` ($\sigma = 0.05$).
- **Perturbations multiplicatives** : multiplication des features cibles par un facteur $f \in \mathbb{R}^+$. Exemples : `speed_change` ($s = 0.15$), `fatigue` ($s = 0.20$), `global_slowdown` ($s = 0.30$), `amplify_motion` ($s = 0.60$), `global_scaling` ($s \in [0.8, 1.2]$), `gender_normalization` ($f = 0.82$), `total_neutralization` ($f \in [0.5, 0.6]$).
- **Perturbations par dropout** : mise à zéro d'un sous-ensemble de features avec probabilité p . Exemples : `sensor_dropout` ($p = 0.10$), `network_packet_loss` ($p = 0.05$), `partial_dropout_gaze` ($p = 0.10$), `tracking_occlusion` ($p = 0.30$).
- **Perturbations structurelles** : modification de la structure des features sans paramètre d'amplitude (swap, symétrie, inversion). Exemples : `mirror_behavior`, `left_right_swap`, `perfect_symmetry`, `feature_collapse`.

6.1.2 Choix des distributions probabilistes

Le choix de la distribution probabiliste pour chaque famille est motivé par la nature physique de la perturbation :

Distribution gaussienne $\mathcal{N}(0, \sigma)$: Utilisée pour les perturbations de type bruit additif (`jitter`, `sensor_noise_burst`, `environmental_noise`, `targeted_jitter_dyn`). Ce choix est justifié par le théorème central limite [130] : la superposition de multiples sources d’erreur indépendantes (vibrations mécaniques, bruit thermique, imprécision de calibration) produit un bruit dont la distribution converge vers une loi normale. C’est le modèle standard de bruit dans les systèmes de suivi inertiel et optique [148].

Distribution uniforme $\mathcal{U}(a, b)$: Utilisée pour les perturbations multiplicatives (`speed_change`, `fatigue`, `global_slowdown`, `amplify_motion`, `light_interference`, `style_randomization`, etc.). Ce choix reflète l’hypothèse que l’attaquant ne connaît pas le niveau exact de la perturbation à appliquer : il opère dans un intervalle borné $[1 - s, 1 + s]$ ou $[1 - s, 1]$, où s est l’amplitude maximale choisie [130]. La distribution uniforme est également adaptée aux modifications comportementales humaines volontaires, dont l’amplitude est naturellement bornée par des contraintes biomécaniques [134].

Distribution de Bernoulli $\mathcal{B}(p)$: Utilisée pour les perturbations de perte de signal (`sensor_dropout`, `network_packet_loss`, `partial_dropout_gaze`). Ce choix est justifié par la nature binaire d’une perte de capteur : à chaque instant, le signal est soit présent (valeur normale) soit absent (valeur nulle), ce qui correspond exactement à une variable de Bernoulli de paramètre p , où p représente la probabilité de perte du signal [130].

6.1.3 Calibration des amplitudes

Les amplitudes des perturbations sont calibrées sur les statistiques descriptives du dataset d’entraînement selon trois règles :

Règle 1 - Perturbations additives : Le paramètre δ est exprimé en fraction de la moyenne ou de l’écart-type de la feature cible :

$$\delta = k \times \sigma_{\text{ref}} \quad (6.1)$$

où σ_{ref} est l'écart-type de la feature dans le dataset d'entraînement et k contrôle l'intensité. Par exemple, $\delta = 0.25$ pour `posture_head_lower` représente 2.3% de l'écart-type de la hauteur du casque ($\sigma_{\text{hmd}_y} = 10.74$), simulant une légère baisse de tête imperceptible. De même, $\sigma = 0.10$ pour `jitter` correspond à $1.5 \times \tilde{\sigma}_{\text{dyn}}$, où $\tilde{\sigma}_{\text{dyn}} = 0.068$ est la médiane des écarts-types des features dynamiques, produisant un bruit légèrement supérieur à la variabilité naturelle du signal.

Règle 2 - Perturbations multiplicatives : Le paramètre *strength* s est exprimé en fraction du coefficient de variation naturel des features :

$$s = k \times \text{CV}_{\text{ref}} \quad (6.2)$$

où $\text{CV}_{\text{ref}} = \sigma_{\text{ref}}/\mu_{\text{ref}}$ est le coefficient de variation médian observé dans le dataset. Le coefficient de variation médian est $\approx 25\%$ sur l'ensemble des catégories de features (HMD, contrôleurs, regard). Ainsi :

- $s = 0.08$ (`hand_instability`) $\rightarrow k = 0.32$: perturbation subtile, en dessous de la variabilité naturelle.
- $s = 0.15$ (`speed_change`) $\rightarrow k = 0.60$: perturbation modérée, correspondant à 60% de la variabilité naturelle.
- $s = 0.30$ (`global_slowdown`) $\rightarrow k = 1.20$: perturbation forte, légèrement supérieure à la variabilité naturelle.
- $s = 0.60$ (`amplify_motion`) $\rightarrow k = 2.40$: perturbation très forte, simulant des gestes extrêmement exagérés.

Règle 3 - Perturbations par dropout : Le taux p est calibré directement sur le taux de valeurs nulles observé dans les features de capteurs du dataset d'entraînement :

$$p \approx \text{zero_rate}_{\text{train}} = \frac{1}{N \cdot F} \sum_{i=1}^N \sum_{j=1}^F \mathbf{1}[x_{ij} = 0] = 0.1046 \quad (6.3)$$

où N est le nombre d'exemples et F le nombre de features de $\mathcal{X}_{\text{train}}$. Cette valeur reflète le taux de perte de signal naturel observé dans les données de capteurs VR :

environ 10.46% des valeurs enregistrées sont nulles, dues aux occlusions partielles ou aux pertes de tracking inhérentes à l’environnement immersif.

Le taux $p = 0.10$ pour `sensor_dropout` et `partial_dropout_gaze` correspond ainsi au taux naturel de valeurs manquantes dans les features oculométriques du dataset, garantissant un scénario de perte de signal physiquement plausible. Le taux $p = 0.05$ pour `network_packet_loss` simule un scénario de perte réseau moins fréquent mais documenté dans la littérature sur les réseaux de capteurs sans fil [149].

Indépendamment de la famille et de l’amplitude choisies, toutes les perturbations sont soumises à un écrêtage systématique dans le range d’entraînement :

$$x_{\text{adv}} \leftarrow \text{clip} \left(x_{\text{adv}}, \min_{\text{train}}, \max_{\text{train}} \right) \quad (6.4)$$

Cette opération, implémentée via la fonction `clip_like_train_cols`, garantit que les valeurs perturbées restent dans le domaine des données réelles et ne produisent pas de valeurs physiquement impossibles, quelle que soit l’intensité de la perturbation appliquée.

La section suivante présente ainsi différentes perturbations comportementales inspirées de la littérature, en établissant un lien explicite entre les stratégies décrites dans les travaux existants et leur instantiation concrète dans le cadre expérimental de ce travail. Certaines de ces perturbations correspondent à des variations naturelles ou non-intentionnelles du comportement, tandis que d’autres peuvent être interprétées comme des modifications intentionnelles visant à tromper le classifieur.

6.2 Scénarios de perturbations comportementales

Afin d’évaluer la vulnérabilité des modèles de reconnaissance du genre étudiés dans ce travail, un ensemble étendu de scénarios de perturbation a été conçu et appliqué aux données de mouvement en réalité virtuelle. Ces scénarios visent à couvrir un large spectre de perturbations affectant les signaux de mouvement en réalité virtuelle. Ils incluent à la fois des variations non intentionnelles liées aux conditions de capture ou aux fluctuations naturelles du comportement, ainsi que des modifications comportementales intentionnelles pouvant être interprétées comme des perturbations comportementales visant à tromper le classifieur.

Il est important de souligner que les scénarios présentés dans cette section constituent des exemples représentatifs de perturbations plausibles, et non une liste exhaustive de

toutes les modifications possibles. De même, les valeurs numériques utilisées pour paramétrer ces perturbations telles que les amplitudes de bruit ou les seuils de modification ont été choisis pour couvrir un éventail de perturbations allant de légères à sévères. Ces valeurs sont données à titre d'exemple et ne constituent pas un choix unique ou définitif : d'autres valeurs auraient pu être utilisées selon le contexte. L'objectif est de démontrer que les modèles sont vulnérables à ce type de perturbations, et non de définir un ensemble de paramètres canoniques.

Un large ensemble de perturbations a été défini et appliqué sans accès aux paramètres internes des modèles, afin d'évaluer la robustesse des modèles tabulaires, neuronaux et temporels dans des conditions réalistes d'utilisation. Ces perturbations ne reposent pas sur l'optimisation explicite d'une fonction de perte, mais visent à reproduire des variations plausibles du comportement de l'utilisateur ou des imperfections du système de capture, telles que le bruit capteur, les dérives progressives, les pertes partielles de signal ou les modifications volontaires de la posture et de la dynamique du mouvement.

Contrairement aux données statiques classiquement étudiées dans la littérature, les signaux de mouvement en réalité virtuelle sont continus, multimodaux et fortement contraints par la biomécanique humaine. Ces spécificités imposent que les perturbations introduites conservent une cohérence temporelle et une plausibilité perceptive afin de rester crédibles dans un contexte immersif. Dans cette optique, les scénarios de perturbation étudiés s'inspirent directement des principes d'analyse de robustesse des classifieurs et de l'étude des vulnérabilités comportementales proposés dans la littérature, tout en les adaptant aux propriétés spécifiques des signaux de mouvement en réalité virtuelle.

6.2.1 Regroupement des perturbations en familles réalistes

Les perturbations étudiées ne sont pas de simples modifications artificielles des données. Elles représentent des situations réelles qui peuvent arriver lors de l'utilisation d'un système de réalité virtuelle, que ce soit de façon naturelle ou volontaire.

Pour rendre l'analyse plus claire, les perturbations ont été regroupées en **familles** selon leur type et leur origine. Plutôt que d'analyser chaque perturbation séparément, ce regroupement permet de voir quels types de modifications ont le plus d'impact sur les modèles. Les familles testées sont :

- les changements naturels de comportement (fatigue, ralentissement, instabilité) ;
- les modifications volontaires du mouvement pour tromper le système ;

- la désynchronisation entre les mouvements de la tête et des mains ;
- les problèmes liés au matériel (bruit, dérive, pertes de signal) ;
- les perturbations liées à l’environnement virtuel ;
- les attaques qui utilisent les réponses du modèle pour se guider ;
- les effets de stylisation du mouvement.

Toutes ces perturbations sont conçues pour rester réalistes et cohérentes avec les mouvements humains normaux.

6.2.2 Famille 1 - Variations comportementales naturelles (non intentionnelles)

TABLEAU 6.1 – Famille 1 - Variations naturelles (non intentionnelles).

Attaque	Scénario réel	Implémentation (code)
fatigue	Session longue / baisse d’effort	Mise à l’échelle des features de variabilité (abs_mean , data_range , std , iqr) : $\mathbf{x}_{cols} \leftarrow \mathbf{x}_{cols} \times (1 - u)$ avec $u \sim \mathcal{U}(0, 0.20)$.
global_slowdown	Fatigue physique / inconfort	Mise à l’échelle des features de dynamique (std , iqr , data_range , abs_mean , mean_velocity , max_velocity) : $\mathbf{x}_{cols} \leftarrow \mathbf{x}_{cols} \times (1 - u)$ avec $u \sim \mathcal{U}(0, 0.30)$.
behavioral_drift	Adaptation naturelle au jeu	Création d’un vecteur de drift croissant sur les features de variabilité (std , range , energy , iqr , coeff_var , abs_mean) : $\mathbf{d} = \text{linspace}(1.0, 1.10, \text{cols})$, puis $\mathbf{x}_{cols} \leftarrow \mathbf{x}_{cols} \odot \mathbf{d}$.
frame_drop_agnostic	Baisse du FPS	Réduction uniforme des features de variabilité (std , range , energy , iqr , coeff_var , abs_mean) : $\mathbf{x}_{cols} \leftarrow 0.70 \cdot \mathbf{x}_{cols}$.
cognitive_distraction_agnostic	Distraction / multitâche	Atténuation uniforme des features de variabilité (std , range , energy , iqr , coeff_var) : $\mathbf{x}_{cols} \leftarrow 0.85 \cdot \mathbf{x}_{cols}$.

Cette première famille, présentée dans le Tableau 6.1, regroupe des perturbations **non intentionnelles** liées aux variations naturelles du comportement de l’utilisateur en réalité virtuelle, telles que la fatigue, le stress, l’adaptation progressive ou les fluctuations d’engagement.

Ces variations induisent des modifications des propriétés statistiques et dynamiques des signaux de mouvement (énergie, amplitude, variabilité), susceptibles d’affecter la stabilité des prédictions et la confiance des modèles. Fréquentes en conditions réelles, elles

constituent un facteur clé de dégradation des performances *in-the-wild* et permettent d'évaluer la robustesse intrinsèque des modèles face à des dérives comportementales réalistes.

6.2.3 Famille 2 - Modifications volontaires du comportement utilisateur

TABLEAU 6.2 – Modifications volontaires du comportement utilisateur.

Attaque	Scénario réel	Implémentation (code)
behavioral_shift	Modification posturale et dynamique volontaire	Perturbation globale des variables sélectionnées par décalage et mise à l'échelle aléatoires : $(x + \delta) \times \alpha$, avec $\delta \in [-0.20, 0.20]$ et $\alpha \in [0.80, 1.20]$
mimic_style_agnostic	Copie d'un autre joueur	Posture ± 0.35 ; morpho $\times (1 \pm 0.25)$; dyn $\times (1 \pm 0.35)$; gaze $\times (1 \pm 0.30)$
behavioral_masking_agnostic	Masquage intentionnel	Réduction énergie/amplitude : dyn $\times 0.60$
total_neutralization_agnostic	Gestuelle volontairement neutre	Contraction conservant le signe : $ x \times factor$, factor $\in [0.50, 0.60]$
self_awareness_agnostic	Conscience d'être observé	Auto-réduction dynamique : dyn $\times U(0.60, 0.90)$
hand_desync	Coordination G/D dégradée	Main gauche : factor $\in [1.00, 1.30]$; main droite : factor $\in [0.70, 1.00]$
temporal_shift_hands_agnostic	Retard moteur asymétrique	Main gauche $\times 0.85$; main droite $\times 1.15$
async_gaze_divergence_agnostic	Regard non aligné	Un œil $\times U(-1.20, -0.70)$ (inversion partielle aléatoire)
vertical_imbalance_hands	Déséquilibre postural	Main gauche -0.15 ; main droite $+0.15$ (shift vertical)
mirror_behavior_agnostic	Effet miroir / inversion latérale	Swap exact des colonnes Left/Right (permutation bilatérale)
amplify_motion	Joueur très expressif	Amplification dynamique : factor $\in [1.00, 1.60]$ (strength=0.60)
expansion_illusion_agnostic	Effet grand geste	Amplification morpho/dyn : factor $\in [1.50, 1.90]$
inverted_style_agnostic	Style inversé	Scale aléatoire : $x \times 0.50$ ou $x \times 1.80$
ghost_gender_hybrid_agnostic	Style ambigu	Posture ± 0.30 ; dyn $\times 1.50$ ou $\times 0.70$
hybrid_morpho_agnostic	Transformation globale	Posture -0.30 ; dyn $\times 0.80$; gaze $\times (-0.80)$

Cette famille regroupe les perturbations liées à des changements intentionnels du comportement de l'utilisateur présentée dans le Tableau 6.2. Elle couvre trois types de

situations : l’auto-masquage du mouvement pour réduire les informations exploitables par le système, la désynchronisation biomécanique entre les différentes parties du corps, et la stylisation ou l’exagération volontaire du mouvement. Ces situations peuvent survenir naturellement en réalité virtuelle par exemple lors de jeux de rôle ou sous l’effet de contraintes de gameplay ou être intentionnellement provoquées par un utilisateur cherchant à tromper le système.

Ce type de perturbations est particulièrement pertinent dans un contexte de sécurité, où un utilisateur averti pourrait chercher à contourner un système d’authentification en modifiant délibérément ses habitudes. Contrairement aux perturbations matérielles, environnementales ou non intentionnelles, ces modifications restent entièrement sous le contrôle de l’utilisateur et peuvent être appliquées de manière discrète et continue.

6.2.4 Famille 3 - Perturbations matérielles, système et environnementales

TABLEAU 6.3 – Perturbations matérielles, système et environnementales.

Attaque	Scénario réel	Implémentation (code)
sensor_offset_agnostic	Calibration imparfaite	Décalage fixe ajouté à toutes les positions : ± 0.08 (seed=123)
sensor_noise_burst_agnostic	Perturbation soudaine du signal (interférence électronique)	Bruit gaussien global : $\epsilon \sim \mathcal{N}(0, 0.05)$ (seed=123)
calibration_bias_agnostic	Mauvaise calibration persistante	Biais additif global : $+0.03$ sur pos/gaze
magnetic_distortion_agnostic	Champ magnétique perturbé	Déformation du signal : factor $\in [0.97, 1.03]$ (seed=123)
thermal_noise_agnostic	Surchauffe du casque RV	Scale $U(0.98, 1.02)$ + bruit $\mathcal{N}(0, 0.01)$ (seed=123)
network_lag_spoof_agnostic	Délai de transmission réseau	Scaling léger sur dyn/velocity : factor $\in [0.92, 1.08]$ (seed=123)
network_packet_loss_agnostic	Pertes réseau	Dropout partiel : mask $\sim$ Bernoulli($p = 0.05$) sur dyn/gaze (seed=123)
light_interference_agnostic	Mauvais éclairage	Jitter multiplicatif léger : factor $\in [0.90, 1.10]$ (seed=123)
tracking_occlusion_agnostic	Capteur masqué	Mise à zéro d’un sous-ensemble de colonnes (rate=0.30, seed=123)
driver_patch_agnostic	Mise à jour logicielle du système	Gain + bruit : $x' = 1.03x + \mathcal{N}(0, 0.02)$ (seed=123)

Cette famille regroupe les perturbations qui ne viennent pas du comportement de l'utilisateur, mais des conditions techniques dans lesquelles l'expérience de réalité virtuelle se déroule, présentée dans le Tableau 6.3. Elle couvre deux types de situations : les dégradations liées aux capteurs et au matériel (bruit, calibration imparfaite, interférences), et les perturbations liées au contexte système et environnemental (réseau, éclairage, occlusions, logiciel). Ces situations sont fréquentes en dehors des conditions de laboratoire, où la configuration matérielle et logicielle peut varier d'un utilisateur à l'autre. Les attaques de cette famille évaluent la robustesse des modèles face aux erreurs de tracking et aux conditions de déploiement réelles, indépendamment du comportement humain.

Une famille supplémentaire, qui regroupe des attaques où l'attaquant ajuste ses modifications en observant les réponses du modèle, est présentée en annexe (voir Annexe A).

6.3 Évaluation de la performance et de la robustesse

L'évaluation de la performance et de la robustesse s'appuie sur des principes largement adoptés dans la littérature sur l'apprentissage adversarial et l'analyse de la robustesse des modèles, en particulier dans des contextes biométriques et comportementaux [150, 21, 2, 27]. Cette approche permet d'analyser non seulement la précision globale des modèles, mais également la stabilité de leurs décisions et l'évolution de leur confiance sous perturbation.

Pour chaque attaque comportementale considérée, une version perturbée du jeu de test, notée X_{adv} , est générée en appliquant l'attaque à l'ensemble des échantillons. L'attaque agit sur un sous-ensemble explicite de variables, correspondant à des dimensions comportementales ou sensorielles ciblées, dont le nombre est enregistré afin de quantifier l'ampleur de la perturbation.

Afin de préserver le réalisme des attaques, les valeurs perturbées sont contraintes à rester dans les intervalles observés sur les données d'entraînement. Cette étape garantit que les dégradations de performance observées résultent de perturbations plausibles, et non de valeurs aberrantes ou physiquement irréalistes.

L'évaluation repose sur une comparaison systématique entre les prédictions obtenues sur les données propres (X_{test}) et sur les données attaquées (X_{adv}). Plusieurs indicateurs complémentaires sont calculés pour chaque modèle et chaque attaque :

-
- **Balanced Accuracy (BAcc)** sur données propres, mesurant la capacité de discrimination du modèle en tenant compte d'un éventuel déséquilibre des classes ;
 - **Balanced Accuracy sous attaque**, évaluant la performance du modèle après perturbation des données ;
 - **Chute de performance (ΔBAcc)**, définie comme la différence entre la balanced accuracy propre et attaquée, fournissant une mesure directe de la vulnérabilité du modèle ;
 - **Taux de flip ou taux de basculement**, correspondant à la proportion d'échantillons pour lesquels la prédiction change entre les données propres et attaquées. Cette métrique permet de quantifier l'instabilité décisionnelle induite par l'attaque, indépendamment de la vérité terrain ;
 - **Confiance moyenne**, définie comme la probabilité (ou pseudo-probabilité) associée à la classe prédite. Pour les modèles probabilistes tels que les Random Forest, Gradient Boosting, XGBoost, LightGBM et les perceptrons multicouches, cette valeur correspond directement à la probabilité estimée pour la classe prédite. Ces modèles produisent en sortie une distribution de probabilités sur les classes, et la confiance est définie comme la probabilité maximale associée à la classe sélectionnée.

Pour les modèles reposant sur un score de décision, notamment les machines à vecteurs de support (SVM), la sortie du modèle correspond à une distance signée par rapport à l'hyperplan de séparation. Cette quantité n'étant pas directement une probabilité, elle est transformée à l'aide d'une fonction logistique (sigmoïde) :

$$p = \frac{1}{1 + e^{-s}}$$

où s représente le score de décision du classifieur. Cette transformation permet d'obtenir une valeur comprise entre 0 et 1, interprétable comme un niveau de confiance comparable aux probabilités produites par les modèles probabilistes ;

- **Chute de confiance**, correspondant à la différence entre la confiance moyenne sur données propres et sur données attaquées, permettant d'identifier des attaques qui fragilisent le modèle même en l'absence de changement explicite de prédiction.

L'utilisation conjointe de ces métriques permet de capturer différents modes de dégradation : perte de précision globale, instabilité locale des décisions et affaiblissement progressif de la certitude du modèle. Cette analyse multi-critère fournit une vision fine

de la robustesse des modèles face à des perturbations comportementales réalistes en environnement de réalité virtuelle.

6.4 Impact des perturbations comportementales sur les modèles classiques

6.4.1 MLP

Analyse des résultats : La Figure 6.1 présente, pour le MLP ($B\text{Acc}_{\text{clean}} = 0.84$), la *balanced accuracy* sous attaque et la confiance de prédiction calculée spécifiquement sur les exemples mal classés, pour les 20 attaques provoquant la plus forte chute de performance.

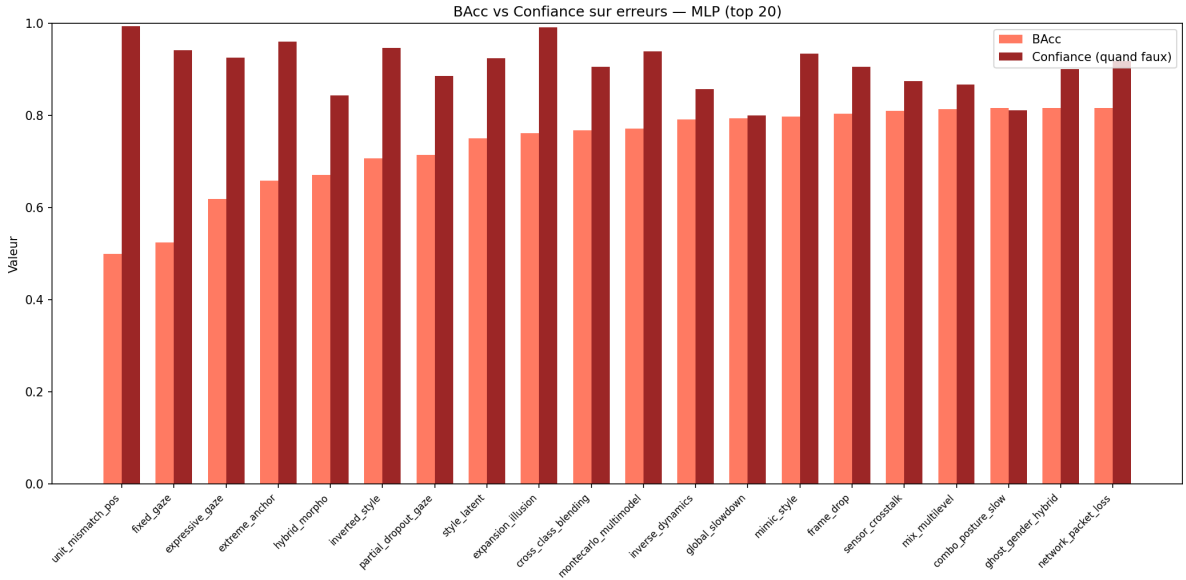


FIGURE 6.1 – BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - MLP.

La Figure 6.1 met en évidence un phénomène marqué de surconfiance pour le MLP : sur l'ensemble des vingt attaques présentées, la confiance sur les erreurs reste systématiquement élevée (entre 0.80 et 0.99), y compris pour les attaques provoquant les chutes de BAcc les plus sévères. Les attaques les plus destructrices sont `unit_mismatch_pos` ($B\text{Acc} \approx 0.50$ pour une confiance sur erreurs de 0.99), `fixed_gaze` ($B\text{Acc} \approx 0.52$ pour

une confiance sur erreurs de 0.94), **expressive_gaze** (BAcc $\approx$ 0.62 pour une confiance sur erreurs de 0.93) et **extreme_anchor** (BAcc $\approx$ 0.66 pour une confiance sur erreurs de 0.96). Dans ces quatre cas, l'écart entre confiance sur erreurs et BAcc dépasse 0.30 point, ce qui indique que le modèle reste très confiant *alors même qu'il se trompe*, et non uniquement confiant en moyenne. La dominance de **unit_mismatch_pos** révèle une sensibilité particulière aux distorsions d'échelle sur les features de position, tandis que la présence de **fixed_gaze** et **expressive_gaze** en deuxième et troisième position confirme une forte dépendance du MLP aux features oculaires pour ses décisions de classification. Pour les attaques restantes du top 20, la BAcc se stabilise progressivement entre 0.79 et 0.82, tandis que la confiance sur les erreurs reste élevée (entre 0.80 et 0.91 selon les attaques), confirmant que le déficit de calibration du MLP persiste même pour des perturbations moins sévères le modèle ne devient jamais réellement "incertain" au moment où il se trompe.

6.4.2 SVM à noyau RBF

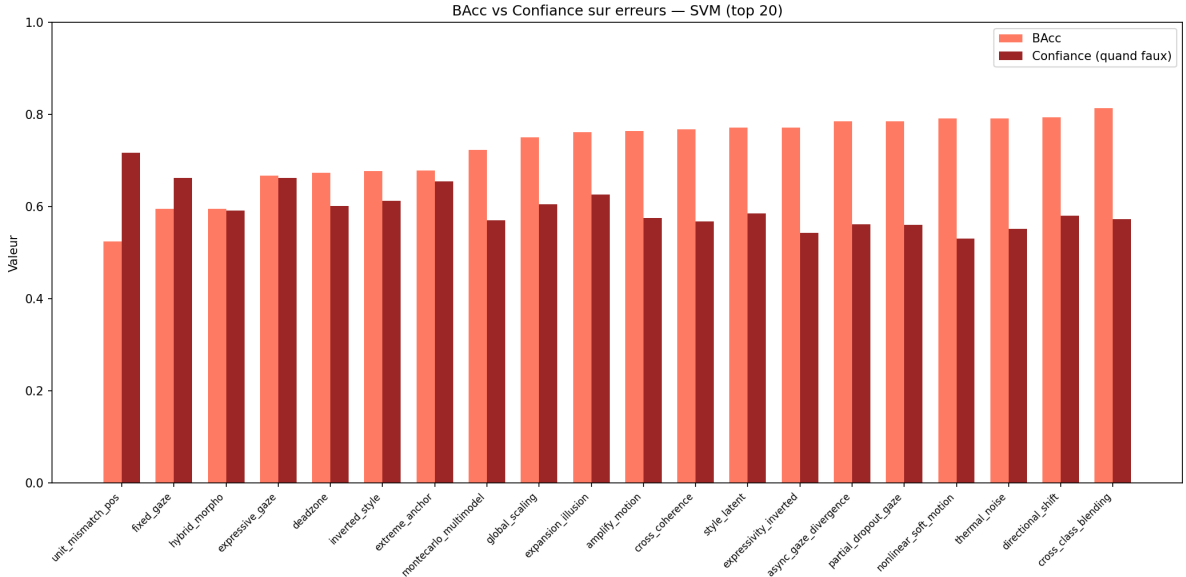


FIGURE 6.2 – BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - SVM (RBF).

Analyse des résultats : La Figure 6.2 présente, pour le SVM à noyau RBF (BAcc_{clean} = 0.93), la *balanced accuracy* sous attaque et la confiance de prédiction cal-

culée spécifiquement sur les exemples mal classés, pour les 20 attaques provoquant la plus forte chute de performance.

Contrairement au MLP, la Figure 6.2 montre que la confiance sur les erreurs du SVM reste plus basse et plus resserrée (entre 0.53 et 0.72), et surtout plus proche de sa BAcc sous attaque signe d’une meilleure calibration. L’attaque `unit_mismatch_pos` se distingue nettement comme la plus destructrice, avec une BAcc chutant à ≈ 0.52 pour une confiance sur erreurs de ≈ 0.72 , l’écart le plus important du top 20 (≈ 0.20 point), révélant ici aussi un début de surconfiance sur cette attaque précise. Cette vulnérabilité extrême s’explique par la sensibilité de la frontière de décision RBF aux distorsions d’échelle : une multiplication par 1000 des features de position déplace massivement les échantillons dans l’espace de feature, traversant la frontière de décision apprise. Les attaques `hybrid_morpho` (BAcc ≈ 0.59 , confiance sur erreurs ≈ 0.59) et `fixed_gaze` (BAcc ≈ 0.59 , confiance sur erreurs ≈ 0.66) constituent le second groupe de vulnérabilité, suivies de `expressive_gaze` (BAcc ≈ 0.67 , confiance sur erreurs ≈ 0.66). Pour ces trois attaques, l’écart entre confiance sur erreurs et BAcc reste faible (inférieur à 0.07 point), confirmant que le SVM, avec seulement 100 features sélectionnées, dépend fortement de quelques dimensions discriminantes dont les features oculaires et multimodales dont la perturbation déstabilise directement la frontière de décision à marge maximale, tout en restant globalement conscient de ses erreurs lorsqu’elles se produisent.

Pour le reste du top 20, la confiance sur les erreurs et la BAcc se rapprochent encore davantage, l’écart devenant quasi nul pour les attaques les moins sévères comme `cross_class_blending` (BAcc ≈ 0.81 , confiance sur erreurs ≈ 0.57), ce qui confirme que le SVM ne présente pas de véritable phénomène de surconfiance, contrairement au MLP.

6.4.3 Gradient Boosting

Analyse des résultats : La Figure 6.3 présente, pour le Gradient Boosting (BAcc_{clean} = 0.93), le modèle le plus performant en conditions nominales avec le SVM, la *balanced accuracy* sous attaque et la confiance de prédiction calculée spécifiquement sur les exemples mal classés, pour les 20 attaques provoquant la plus forte chute de performance. Elle révèle le phénomène de surconfiance le plus marqué parmi les modèles évalués :

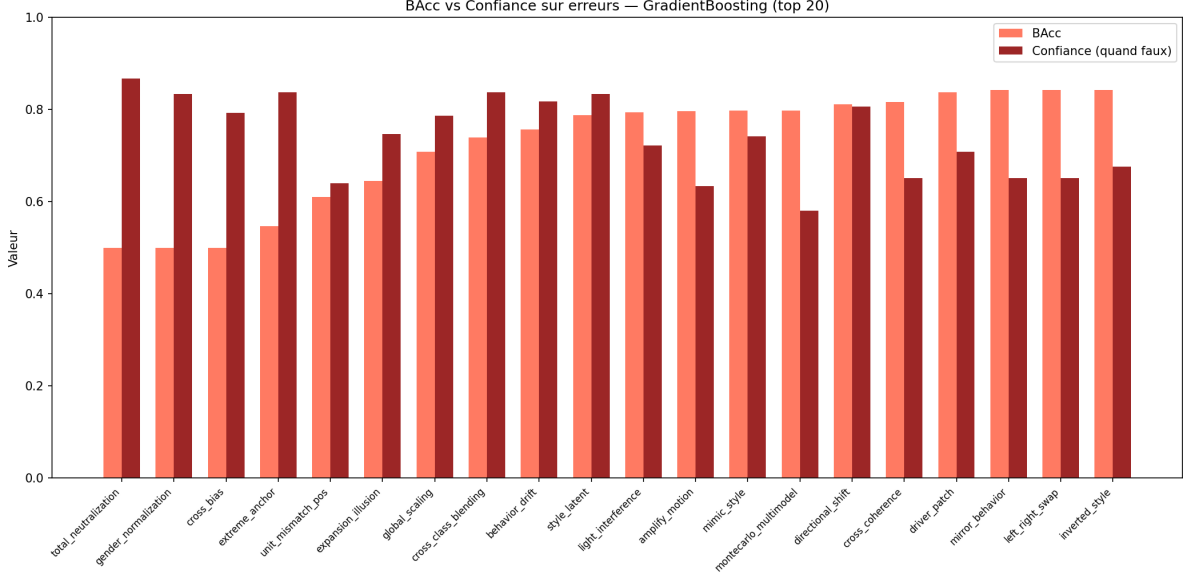


FIGURE 6.3 – BACC sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - Gradient Boosting.

Trois attaques (`total_neutralization`, `gender_normalization` et `cross_bias`) ramènent la BACC à des niveaux proches du hasard (≈ 0.50), alors que la confiance sur les erreurs reste très élevée (entre 0.79 et 0.87), créant un écart de plus de 0.29 point pour ces trois attaques. Ces attaques ciblent directement les marqueurs comportementaux genrés (posture, dynamique, regard) en les amplifiant ou réduisant fortement, effaçant ainsi les signatures discriminantes que le Gradient Boosting exploite pour ses décisions, sans pour autant réduire sa confiance au moment où il se trompe. L'attaque `unit_mismatch_pos` présente un comportement plus modéré : sa confiance sur les erreurs est parmi les plus basses du top 20 (≈ 0.64) pour une BACC ≈ 0.61 , un écart plus faible qui traduit ici une meilleure cohérence entre confiance et performance. Pour `extreme_anchor` (BACC ≈ 0.55 , confiance sur erreurs ≈ 0.84) et `expansion_illusion` (BACC ≈ 0.64 , confiance sur erreurs ≈ 0.75), l'écart reste également marqué, confirmant une dépendance du modèle à l'amplitude absolue des features genrées accompagnée d'une confiance qui ne diminue pas au moment où le modèle se trompe. Pour le reste du top 20, la BACC et la confiance sur les erreurs se rapprochent progressivement, jusqu'à converger presque totalement pour les attaques les moins sévères comme `directional_shift` (BACC ≈ 0.81 , confiance sur erreurs ≈ 0.81) et `inverted_style` (BACC ≈ 0.84 , confiance sur erreurs

≈ 0.68), confirmant que le déficit de calibration du Gradient Boosting se concentre principalement sur les attaques les plus destructrices.

6.4.4 LightGBM

Analyse des résultats : La Figure 6.4 présente, pour LightGBM ($\text{BAcc}_{\text{clean}} = 0.86$), la *balanced accuracy* sous attaque et la confiance de prédiction calculée spécifiquement sur les exemples mal classés, pour les 20 attaques provoquant la plus forte chute de performance. Elle montre un phénomène de surconfiance modéré, plus marqué que pour le SVM mais nettement moins prononcé que pour le Gradient Boosting.

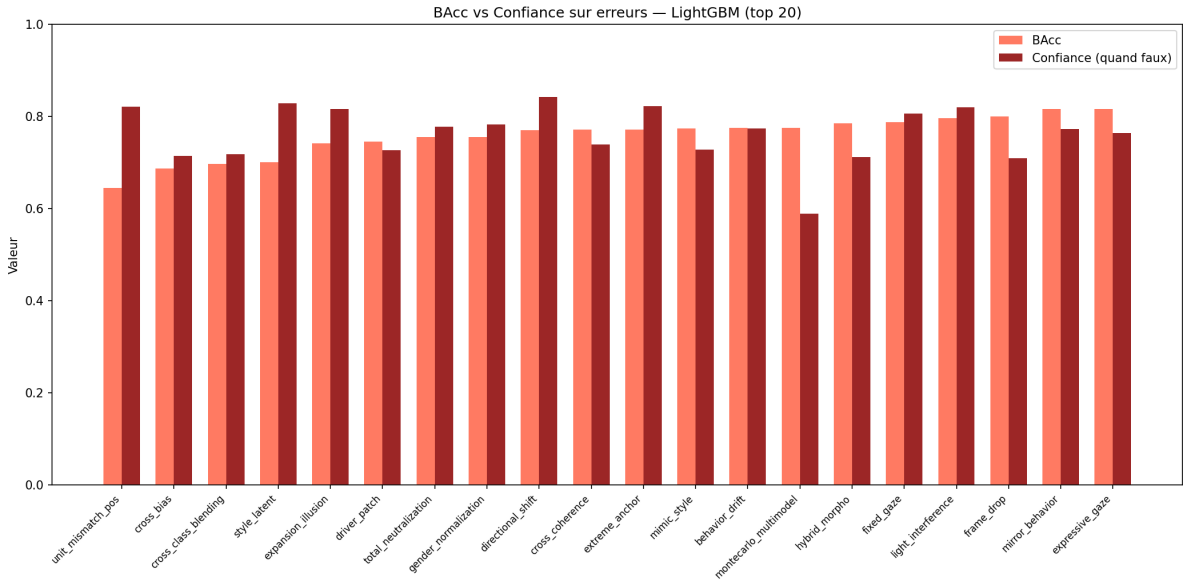


FIGURE 6.4 – BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - LightGBM.

L'attaque `unit_mismatch_pos` présente l'écart le plus important du top 20 : une BAcc chutant à ≈ 0.64 pour une confiance sur erreurs qui reste à ≈ 0.82 , soit un écart de 0.18 point. `style_latent` ($\text{BAcc} \approx 0.70$, confiance sur erreurs ≈ 0.83) et `directional_shift` ($\text{BAcc} \approx 0.77$, confiance sur erreurs ≈ 0.84) présentent des écarts également marqués, entre 0.07 et 0.13 point. Contrairement au Gradient Boosting, LightGBM est particulièrement vulnérable aux attaques d'ancrage extrême et de déplacement vers le centroïde de classe opposée (`style_latent`, `cross_class_blending`), qui exploitent les frontières de décision apprises par le modèle pour interpoler vers des

profils de genre opposé, sans que la confiance sur les erreurs ne baisse en proportion. Pour **extreme_anchor** ($\text{BAcc} \approx 0.77$, confiance sur erreurs ≈ 0.82) et **expansion_illusion** ($\text{BAcc} \approx 0.74$, confiance sur erreurs ≈ 0.82), l'écart reste également marqué, autour de 0.05–0.08 point. Pour le reste du top 20, la BAcc et la confiance sur les erreurs se rapprochent progressivement, l'écart devenant minimal pour les attaques les moins sévères comme **montecarlo_multimodel** ($\text{BAcc} \approx 0.78$, confiance sur erreurs ≈ 0.59 , la seule attaque où la confiance sur erreurs chute nettement sous la BAcc) et **expressive_gaze** ($\text{BAcc} \approx 0.82$, confiance sur erreurs ≈ 0.77), ce qui confirme l'absence d'effondrement total vers le hasard pour LightGBM.

6.4.5 Random Forest

Analyse des résultats : La Figure 6.5 présente, pour le Random Forest ($\text{BAcc}_{\text{clean}} = 0.88$), la *balanced accuracy* sous attaque et la confiance de prédiction calculée spécifiquement sur les exemples mal classés, pour les 20 attaques provoquant la plus forte chute de performance. Elle confirme la robustesse structurelle distinctive du Random Forest : c'est le seul modèle évalué où la confiance sur les erreurs reste systématiquement plus basse que la BAcc sur l'intégralité du top 20, un comportement inverse de celui observé pour les autres modèles.

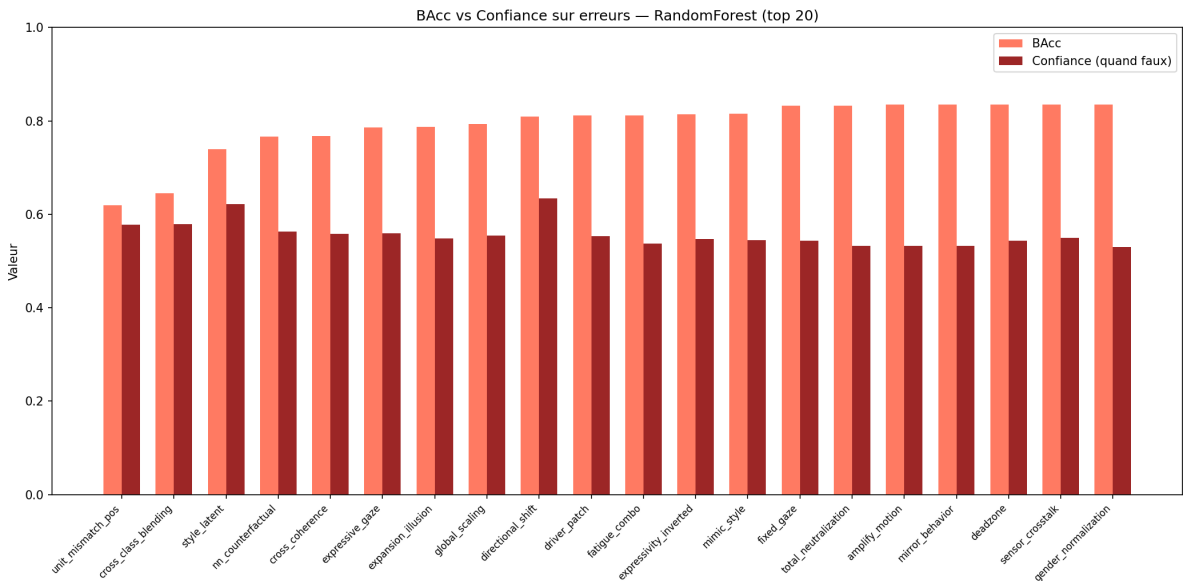


FIGURE 6.5 – BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - Random Forest.

L'attaque `unit_mismatch_pos` présente le cas le plus proche d'une égalité entre confiance sur erreurs et BAcc (≈ 0.58 contre ≈ 0.62), tandis que pour `amplify_motion`, la confiance sur erreurs (≈ 0.53) reste nettement inférieure à la BAcc (≈ 0.84), un écart de 0.31 point dans le sens opposé à la surconfiance observée chez les autres modèles. Les attaques `cross_class_blending` (confiance sur erreurs ≈ 0.58 , BAcc ≈ 0.64) et `style_latent` (confiance sur erreurs ≈ 0.62 , BAcc ≈ 0.74) figurent parmi les cas les plus sévères en BAcc, mais la confiance sur les erreurs y demeure également modérée à faible, révélant que la frontière du Random Forest, bien que robuste aux perturbations locales, peut être franchie par des déplacements dirigés vers des profils de l'autre classe sans que le modèle ne se montre excessivement confiant dans ces cas d'erreur. Pour la majorité des attaques restantes du top 20, la BAcc se stabilise autour de 0.81-0.84 tandis que la confiance sur les erreurs reste nettement inférieure, entre 0.53 et 0.62, confirmant que le Random Forest, contrairement aux autres modèles, ne souffre pas de surconfiance et présente même une tendance à la sous-confiance sur ses erreurs un comportement qui, bien que prudent, pourrait aussi refléter le mécanisme de vote majoritaire des arbres indépendants plutôt qu'une évaluation fine de la difficulté de chaque exemple.

6.4.6 XGBoost

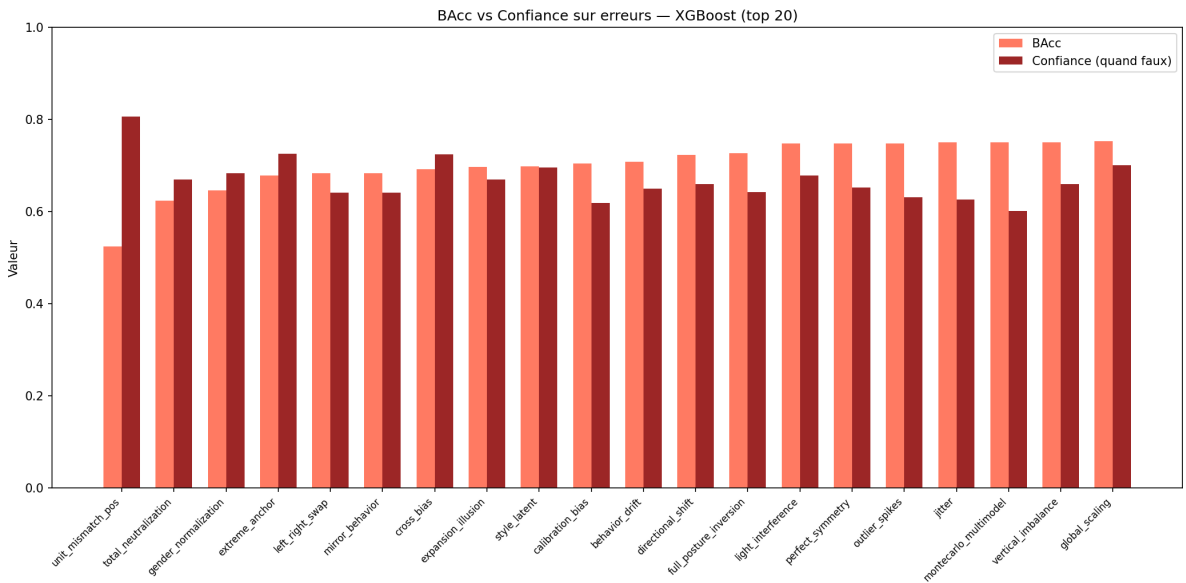


FIGURE 6.6 – BAcc sous attaque et confiance de prédiction sur les exemples mal classés, pour les 20 attaques les plus sévères - XGBoost.

Analyse des résultats : La Figure 6.6 présente, pour XGBoost ($B\text{Acc}_{\text{clean}} = 0.77$), la *balanced accuracy* sous attaque et la confiance de prédiction calculée spécifiquement sur les exemples mal classés, pour les 20 attaques provoquant la plus forte chute de performance. Elle montre un phénomène de surconfiance modéré et relativement homogène sur l’ensemble du top 20.

L’attaque `unit_mismatch_pos` présente l’écart le plus marqué, avec une $B\text{Acc}$ chutant à ≈ 0.52 pour une confiance sur erreurs qui reste à ≈ 0.81 , soit un écart de 0.29 point, le plus important observé pour ce modèle. Les attaques `total_neutralization` ($B\text{Acc} \approx 0.62$, confiance sur erreurs ≈ 0.67) et `gender_normalization` ($B\text{Acc} \approx 0.65$, confiance sur erreurs ≈ 0.68) suivent avec des écarts plus modérés, autour de 0.03–0.05 point, confirmant que seule la distorsion d’échelle par $\times 1000$ sur les positions provoque à la fois la dégradation la plus sévère et le déficit de calibration le plus marqué. Pour le reste du top 20, la confiance sur les erreurs reste globalement supérieure à la $B\text{Acc}$, mais avec un écart plus resserré, entre 0.00 et 0.05 point (par exemple `extreme_anchor` : $B\text{Acc} \approx 0.68$, confiance sur erreurs ≈ 0.73 ; `style_latent` : $B\text{Acc}$ et confiance sur erreurs quasi identiques, ≈ 0.70), ce qui indique une tendance à la surconfiance globalement homogène et modérée, sans effondrement aussi marqué que pour le Gradient Boosting. Cela montre que XGBoost reste souvent un peu plus confiant qu’il ne devrait au moment où il se trompe, mais sans que ce problème s’aggrave brusquement face aux perturbations testées.

6.4.7 Résultats globaux :

L’analyse de l’impact des perturbations sur les modèles montre des comportements très différents selon le type de modèle utilisé. Cela confirme que la robustesse d’un modèle dépend directement de la façon dont il apprend et représente les données. Les modèles évalués incluent des méthodes à base d’arbres (Random Forest, $B\text{Acc}_{\text{clean}} = 0.88$), des approches de *boosting* (Gradient Boosting 0.93, XGBoost 0.77, LightGBM 0.86) ainsi qu’un classifieur à marge maximale (SVM à noyau RBF 0.93) et un perceptron multicouche (MLP 0.84).

Les figures 6.7 et 6.8 présentent respectivement la chute de *balanced accuracy* ($\Delta B\text{Acc}$) et le taux de basculement (*flip rate*) pour chaque combinaison modèle/attaque, et permettent de visualiser d’un seul coup d’œil les profils de vulnérabilité propres à chaque modèle.

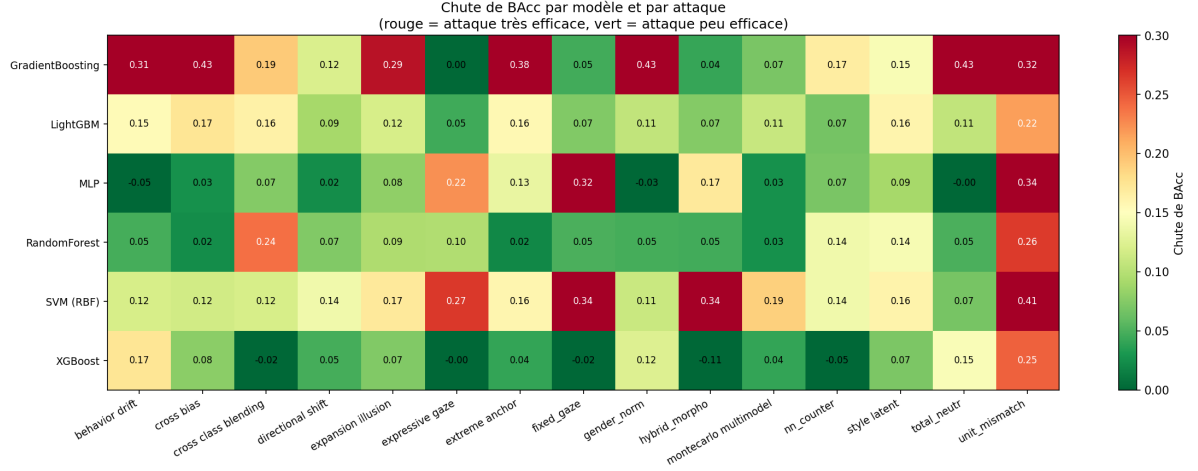


FIGURE 6.7 – Chute de *balanced accuracy* ($\Delta BAcc$) par modèle et par type d'attaque. Le rouge indique une attaque très efficace (forte dégradation des performances), tandis que le vert indique une attaque peu efficace (valeurs négatives).

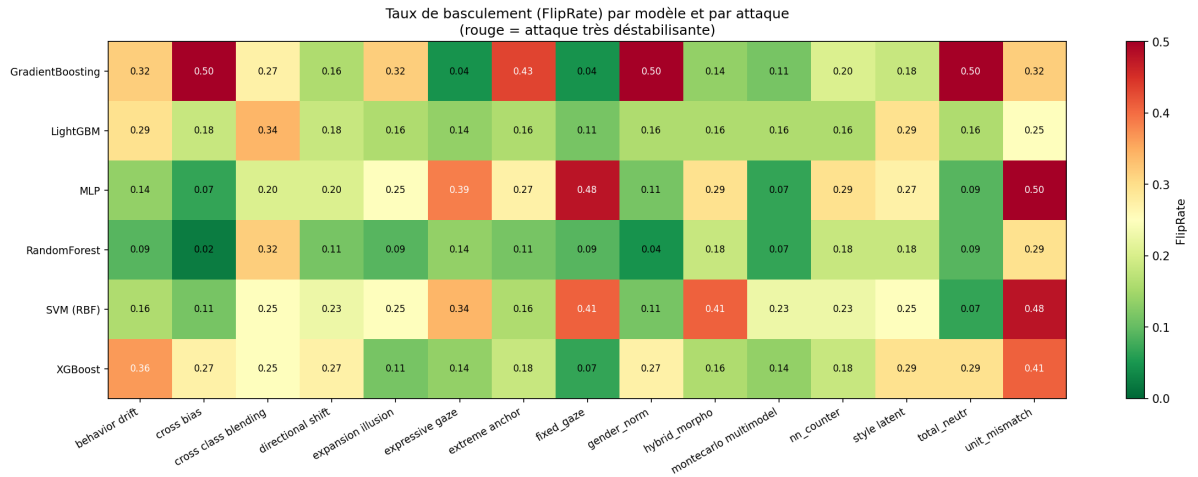


FIGURE 6.8 – Taux de basculement (*FlipRate*) par modèle et par type d'attaque, mesurant la proportion de prédictions dont la classe change sous l'effet de la perturbation. Le rouge indique une attaque très déstabilisante.

Les résultats montrent que la majorité des attaques faiblement structurées ou assimilables à une variabilité comportementale naturelle (*speed_change*, *posture_head_lower*, *async_gaze_divergence*, *hand_instability*, *fatigue_combo_agnostic*) induisent un impact limité sur les performances des modèles

classiques. Dans ces scénarios, la *balanced accuracy* reste proche de celle observée en condition nominale, avec des chutes généralement inférieures à 0.05 et des *flip rates* faibles, voire nuls pour certains modèles, comme le confirment les zones vertes dominantes observées pour RandomForest et XGBoost sur les figures 6.7 et 6.8. Ce constat est particulièrement marqué pour le Random Forest, qui présente le profil de robustesse le plus favorable avec onze attaques sur vingt classées en sévérité faible, et pour XGBoost qui compte huit attaques faibles sur vingt. À l'inverse, le Gradient Boosting et le SVM ne présentent aucune attaque à sévérité faible dans leur top 20, révélant une sensibilité plus uniforme à l'ensemble du spectre des perturbations, comme l'illustrent les nombreuses cases rouges et orangées présentes sur leurs lignes respectives dans les deux heatmaps. Cette stabilité face aux perturbations légères confirme la capacité des méthodes d'ensemble à absorber un certain niveau de bruit ou de variabilité locale grâce à l'agrégation de multiples estimateurs.

Vulnérabilités spécifiques par modèle : Le **Random Forest** se distingue comme le modèle le plus robuste, avec seulement trois attaques à sévérité élevée, onze attaques faibles et un phénomène paradoxal d'amélioration des performances sous certaines perturbations ($B\text{Acc}_{\text{adv}} \geq 0.62$). Les attaques les plus marquées demeurent `unit_mismatch_pos` ($\Delta B\text{Acc} \approx 0.26$) et `cross_class_blending` (≈ 0.24).

Le **SVM (RBF)** et le **Gradient Boosting** atteignent tous deux $B\text{Acc}_{\text{clean}} = 0.93$, mais diffèrent nettement : le SVM est sensible aux distorsions d'échelle (`unit_mismatch_pos`, $\Delta B\text{Acc} \approx 0.41$) et aux attaques multimodales (≈ 0.34). Le Gradient Boosting, lui, s'effondre vers le hasard ($B\text{Acc}_{\text{adv}} \approx 0.50$) pour trois attaques ciblant les marqueurs genrés ($\Delta B\text{Acc} \approx 0.43$).

XGBoost ($B\text{Acc}_{\text{clean}} = 0.77$) et **LightGBM** ($B\text{Acc}_{\text{clean}} = 0.86$) présentent une robustesse intermédiaire, avec `unit_mismatch_pos` comme attaque dominante ($\Delta B\text{Acc} \approx 0.25$ et 0.22 respectivement, sans effondrement vers le hasard).

Le **MLP** présente un profil intermédiaire en $B\text{Acc}$ ($B\text{Acc}_{\text{adv}} \geq 0.61$, sans effondrement), avec `unit_mismatch_pos` ($\Delta B\text{Acc} \approx 0.34$) et `fixed_gaze` (≈ 0.32) comme attaques dominantes.

En résumé, les attaques globales (distorsion d'échelle, neutralisation des marqueurs genrés) provoquent des chutes significatives pour tous les modèles tabulaires ($\Delta B\text{Acc}$ jusqu'à 0.43), contrairement aux perturbations locales, à l'impact marginal.

Analyse de la confiance et phénomène de surconfiance : Les figures 6.9 et 6.10 complètent cette analyse en présentant respectivement la confiance moyenne sous attaque et l'écart absolu entre confiance et *balanced accuracy* pour chaque combinaison modèle/attaque. Ce second indicateur permet d'évaluer la calibration des modèles : un écart proche de zéro signale une confiance cohérente avec la performance réelle, qu'elle soit supérieure ou inférieure à la BAcc, tandis qu'un écart important dans un sens comme dans l'autre révèle que la confiance ne constitue pas un indicateur fiable de la justesse de la prédiction.

Le MLP présente les écarts les plus extrêmes de tous les modèles, avec un écart absolu atteignant 0.50 sur `unit_mismatch` et 0.44 sur `fixed_gaze` (figure 6.10), sa confiance restant quasi maximale (0.94 à 1.00) sur l'ensemble des attaques testées (figure 6.9), indépendamment de la dégradation réelle de ses performances il s'agit ici d'un cas de surconfiance au sens strict.

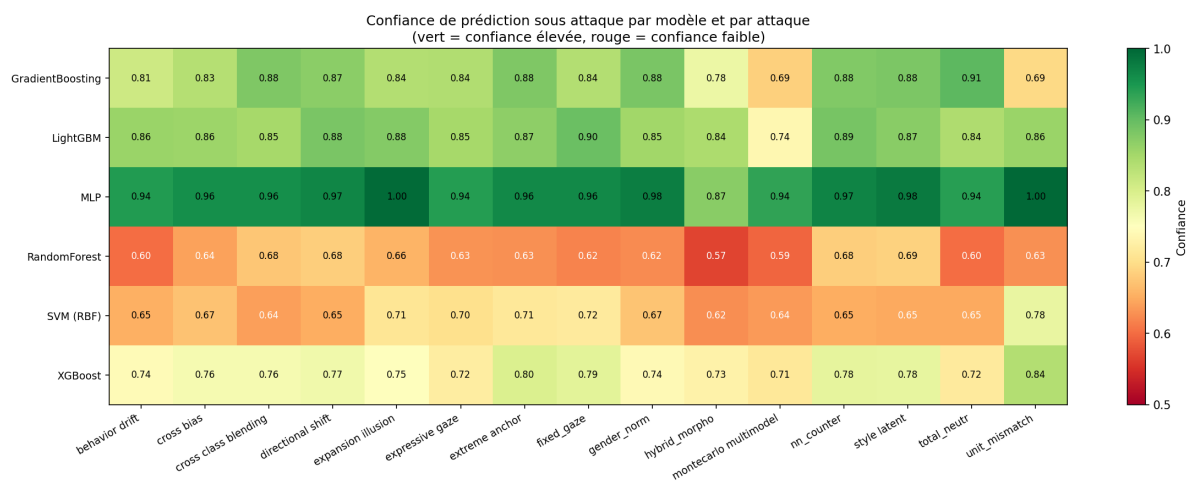


FIGURE 6.9 – Confiance moyenne de prédiction sous attaque par modèle et par type d'attaque. Le vert indique une confiance élevée, le rouge une confiance faible.

Le Gradient Boosting présente un profil tout aussi problématique, avec des écarts absolus atteignant 0.41 sur `total_neutr`, 0.38 sur `gender_norm` et 0.33 sur `cross_bias` et `expressive_gaze` : dans ces scénarios, la confiance moyenne demeure élevée (0.84 à 0.88) alors même que la *balanced accuracy* s'effondre vers 0.50, confirmant un phénomène de surconfiance similaire à celui du MLP.

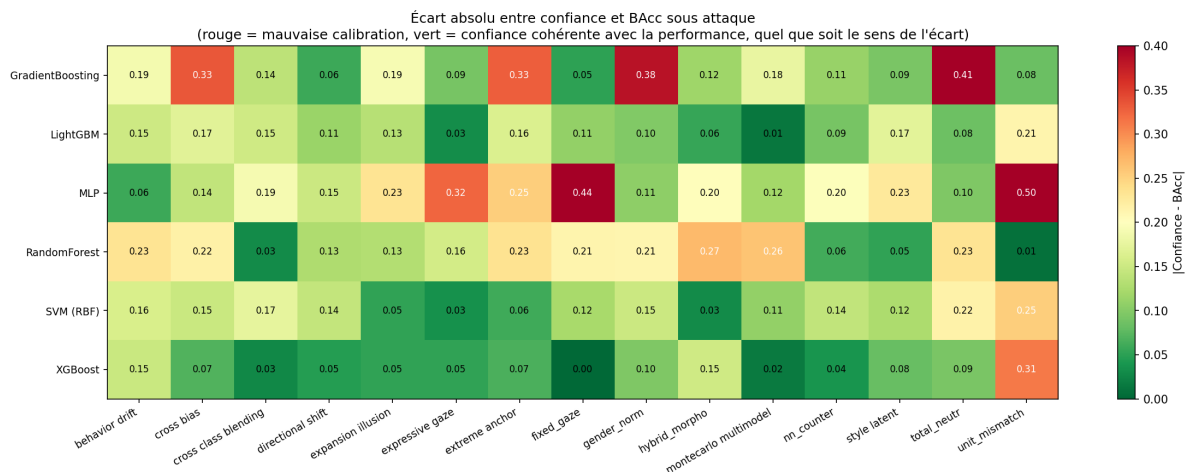


FIGURE 6.10 – Écart absolu entre confiance et *balanced accuracy* sous attaque par modèle et par type d'attaque. Le rouge indique une mauvaise calibration (confiance éloignée de la performance réelle, dans un sens ou dans l'autre), le vert une confiance cohérente avec la performance.

Le Random Forest et le SVM (RBF) présentent, quant à eux, des écarts absolus globalement plus modérés mais loin d'être négligeables : le Random Forest atteint 0.27 sur *hybrid_morpho* et 0.26 sur *montecarlo_multimodel*, tandis que le SVM atteint 0.25 sur *unit_mismatch*. Dans ces deux cas, l'écart provient d'une confiance systématiquement *inférieure* à la BAcc (par exemple 0.57–0.69 de confiance pour le Random Forest, contre une BAcc souvent supérieure à 0.80) : si ce comportement est moins problématique qu'une surconfiance, il signifie néanmoins que la confiance de ces deux modèles ne permet pas de distinguer efficacement une prédiction fiable d'une prédiction non fiable, la confiance restant basse que la prédiction soit correcte ou non. Un classifieur parfaitement calibré devrait au contraire présenter une confiance proche de la BAcc, ce qui n'est le cas pour aucun des six modèles évalués sur l'ensemble du spectre des attaques.

XGBoost et LightGBM affichent les écarts absolus les plus faibles et les plus homogènes parmi les modèles évalués, restant sous 0.10 pour la majorité des attaques (à l'exception notable de *unit_mismatch*, où l'écart atteint 0.31 pour XGBoost et 0.21 pour LightGBM), ce qui en fait les deux modèles dont la confiance reflète le plus fidèlement la performance réelle sur l'ensemble des perturbations testées. Cette hétérogénéité confirme que la calibration de la confiance dépend fortement de l'architecture du modèle, et qu'aucune des deux limites surconfiance excessive (MLP, Gradient Boosting) ou sous-

confiance excessive (Random Forest, SVM) ne constitue un comportement réellement souhaitable pour un déploiement en conditions réelles.

6.5 Impact des perturbations comportementales sur les modèles temporels

Dans le cas des modèles temporels (LSTM, CNN-LSTM, ResNet1D, TimesNet), les attaques comportementales sont appliquées au niveau des *features brutes* avant toute normalisation. Chaque séquence tridimensionnelle (N, T, F) , où N représente le nombre de séquences, T la longueur temporelle et F le nombre de variables par frame, est temporairement reformatée en structure tabulaire afin de permettre l'application des fonctions d'attaque existantes. Les perturbations modifient ainsi directement les signaux comportementaux (posture, dynamique, regard, synchronisation bimanuelle, bruit capteur, etc.) à l'échelle *frame-par-frame*. Après génération des séquences perturbées, celles-ci sont renormalisées à l'aide d'un *StandardScaler* ajusté exclusivement sur les données d'entraînement, garantissant l'absence de fuite d'information. Les prédictions sont ensuite effectuées en mini-batch afin d'assurer la stabilité mémoire. Les métriques de robustesse utilisées *balanced accuracy*, *Drop* (perte de performance), *Flip rate* (taux de basculement) et variation de confiance moyenne permettent d'évaluer précisément l'impact de chaque perturbation sur la dynamique décisionnelle du modèle temporel.

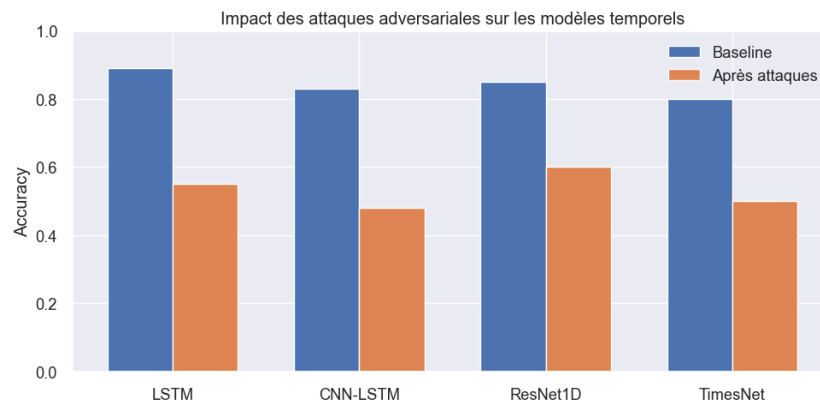


FIGURE 6.11 – Impact des perturbations comportementales sur les modèles temporels.

La Figure 6.11 illustre la dégradation observée pour les principales architectures séquentielles. En condition normale, les performances se situent entre 0.80 et 0.90 de

balanced accuracy. Sous attaques comportementales globales, cette performance chute significativement, atteignant des valeurs comprises entre 0.48 et 0.60 selon l’architecture considérée.

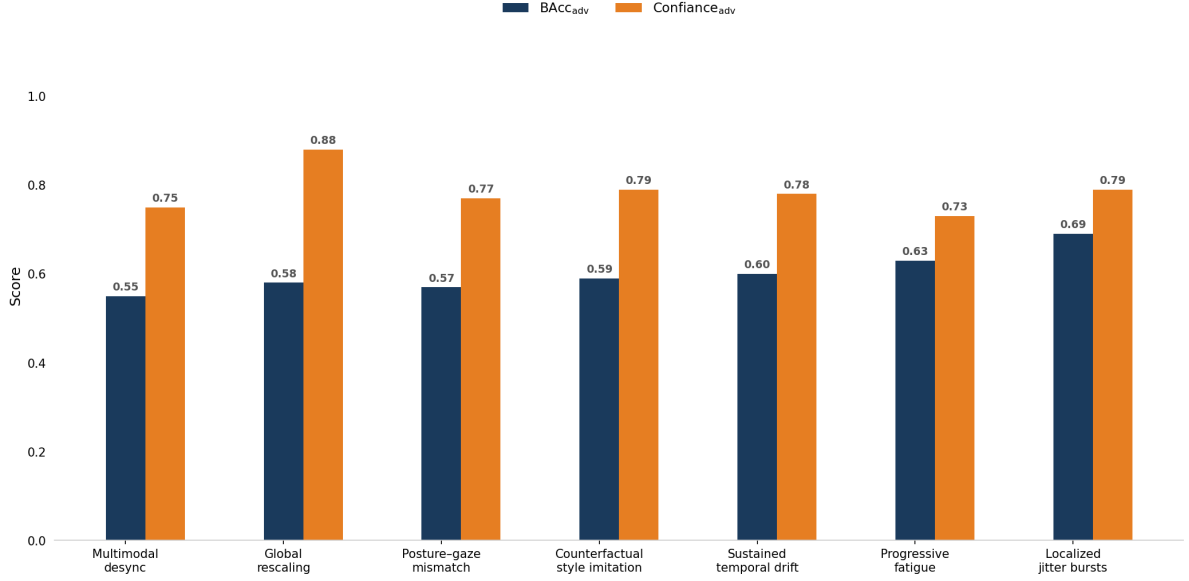


FIGURE 6.12 – Détails de l’impact des sept attaques comportementales les plus représentatives sur les modèles temporels, mesuré en *balanced accuracy* après attaque (BAcc_{adv}, bleu foncé) et en confiance de prédiction sur les exemples mal classés (Confiance_{adv}, orange).

La Figure 6.12 présente en détail l’impact des sept attaques les plus représentatives sur les modèles temporels, ainsi que la confiance de prédiction associée. Les attaques les plus destructrices sont celles qui altèrent simultanément plusieurs modalités comportementales : *Multimodal desync* (BAcc_{adv} = 0.55, Δ BAcc = 0.33, confiance = 0.75), *Global rescaling* (BAcc_{adv} = 0.58, Δ BAcc = 0.31, confiance = 0.88) et *Posture-gaze mismatch* (BAcc_{adv} = 0.57, Δ BAcc = 0.30, confiance = 0.77). Dans chacun de ces cas, la confiance reste nettement supérieure à la BAcc observée, avec un écart maximal de 0.30 pour *Global rescaling*. À l’inverse, les perturbations locales (*Progressive fatigue*, BAcc_{adv} = 0.63, confiance = 0.73 ; *Localized jitter bursts*, BAcc_{adv} = 0.69, confiance = 0.79) maintiennent une performance résiduelle nettement plus élevée, avec un écart confiance-BAcc plus resserré (respectivement 0.10 et 0.10).

L’analyse met en évidence une dynamique de vulnérabilité spécifique aux architectures séquentielles, ainsi qu’un phénomène de surconfiance systématique : sur les sept

attaques testées, la confiance reste supérieure à la BAcc dans tous les cas, avec un écart variant de 0.10 à 0.30 point. Contrairement aux modèles statiques reposant sur des statistiques agrégées, les modèles temporels apprennent des dépendances dynamiques à long terme entre posture, dynamique et regard. Les attaques globales modifiant simultanément un grand nombre de variables au cours du temps (telles que l'inversion comportementale, le *global scaling* ou le *unit mismatch*) déplacent fortement les trajectoires séquentielles dans l'espace latent appris par le réseau, entraînant une chute marquée de la performance, un taux élevé de basculement des prédictions, et l'écart confiance-BAcc le plus important observé (0.30 pour *Global rescaling*).

Les perturbations rompant la cohérence multimodale notamment les désynchronisations bimanuelles, les décalages temporels injectés ou les altérations simultanées posture regard affectent également de manière significative la stabilité décisionnelle, avec des écarts confiance-BAcc comparables (autour de 0.20 pour *Counterfactual style imitation* et *Sustained temporal drift*). Le LSTM exploitant explicitement les corrélations inter-modales au fil du temps, toute rupture artificielle de ces dépendances génère une incohérence séquentielle qui fragilise la frontière de décision, sans pour autant réduire la confiance du modèle dans ses prédictions.

En revanche, les perturbations locales et physiologiquement plausibles (ralentissement modéré, fatigue légère, variations dynamiques faibles) présentent un impact plus limité et un écart confiance-BAcc plus resserré (environ 0.10). Leur effet tend à être partiellement absorbé par la modélisation temporelle, suggérant qu'une partie des fluctuations naturelles du mouvement est intégrée comme variabilité tolérable par le réseau, avec une confiance qui reflète alors plus fidèlement la performance réelle du modèle.

Enfin, la figure met également en évidence des différences inter-architectures : les modèles à représentation convolutionnelle ou fréquentielle (ResNet1D, TimesNet) conservent légèrement plus de stabilité que le LSTM ou le CNN-LSTM sous certaines perturbations globales, bien qu'aucune architecture ne parvienne à neutraliser totalement les attaques séquentielles.

Les architectures séquentielles (LSTM, CNN-LSTM, ResNet1D, TimesNet) présentent un comportement distinct des modèles statiques, dans la mesure où elles intègrent explicitement la dynamique temporelle du mouvement. Contrairement aux modèles classiques fondés sur des statistiques agrégées, ces architectures apprennent des dépendances inter-frames et des corrélations à long terme entre posture, dynamique et regard. Dans ce contexte, les perturbations comportementales n'agissent pas uniquement sur des am-

plitudes ou des moyennes globales, mais sur la cohérence temporelle des séquences. Les attaques introduisant des désynchronisations (mains-tête), des décalages progressifs, des variations non linéaires ou des ruptures de cohérence multimodale affectent directement la structure dynamique encodée par le modèle. Ainsi, même des modifications d’amplitude modérée peuvent induire une instabilité décisionnelle lorsqu’elles altèrent la régularité interne des motifs temporels.

Cette sensibilité s’explique par la nature même des modèles séquentiels : la représentation latente repose sur l’évolution du signal dans le temps. Dès qu’une attaque perturbe la continuité ou la synchronisation des patterns comportementaux, l’encodage global devient incohérent, ce qui peut entraîner une augmentation du taux de basculement et une chute significative de la *balanced accuracy*. En ce sens, la vulnérabilité des modèles temporels ne dépend pas uniquement de l’intensité numérique de la perturbation, mais surtout de sa *forme temporelle* et de sa capacité à rompre les corrélations dynamiques apprises.

6.6 Conclusion du chapitre

Ce chapitre a étudié l’impact des perturbations comportementales sur les modèles de reconnaissance du genre fondés sur les données de mouvement en réalité virtuelle. À travers l’analyse d’un large ensemble d’attaques comportementales réalistes touchant la posture, la dynamique gestuelle, la coordination multimodale et la calibration des capteurs, les résultats montrent qu’une altération structurée du signal suffit à faire chuter nettement les performances, avec des chutes de *balanced accuracy* allant jusqu’à $\Delta \text{BAcc} = 0.43$ pour le Gradient Boosting, notamment sous les attaques `cross_bias`, `gender_norm` et `total_neutr`.

Les expériences menées sur l’ensemble des modèles évalués modèles tabulaires (Random Forest, Gradient Boosting, SVM, XGBoost, LightGBM, MLP) et architectures temporelles (LSTM, CNN-LSTM, ResNet1D, TimesNet) montrent des vulnérabilités différentes selon le type de perturbation et la famille de modèles. Les attaques locales ou proches d’un comportement naturel (`posture_head_lower`, `hand_instability`, *Localized jitter bursts*) ont un impact limité, avec des chutes généralement inférieures à 0.10 pour la majorité des modèles tabulaires, tandis que les modifications globales, multimodales ou structurelles du signal (`unit_mismatch_pos`, `total_neutralization`, `gender_normalization`, *Multimodal desync*) provoquent des chutes marquées de per-

formance et des taux de basculement élevés (*flip rate* jusqu'à 0.50, observé à la fois pour le Gradient Boosting sur `cross_bias`, `fixed_gaze` et `total_neutr`, et pour le MLP sur `unit_mismatch`). Dans plusieurs de ces cas, le modèle reste confiant, voire un peu plus confiant, alors même que sa BAcc s'effondre, avec un écart entre confiance et BAcc qui atteint 0.50 pour le MLP et 0.41 pour le Gradient Boosting les cas les plus marqués observés ce qui montre que ces deux modèles restent sûrs d'eux même quand ils se trompent beaucoup.

La comparaison entre modèles montre que la robustesse varie beaucoup d'une famille à l'autre. Le Random Forest a le meilleur profil de robustesse ($\text{BAcc}_{\text{adv}} \geq 0.62$, seulement deux attaques à sévérité élevée, `unit_mismatch_pos` $\Delta\text{BAcc} \approx 0.26$ et `cross_class_blending` ≈ 0.24), et sa confiance reste même plus basse que sa BAcc sur presque toutes les attaques (écart jusqu'à -0.27), ce qui montre qu'il n'est jamais trop confiant. À l'inverse, le Gradient Boosting s'effondre au niveau du hasard ($\text{BAcc}_{\text{adv}} \approx 0.50$) sous quatre attaques (`cross_bias`, `fixed_gaze`, `gender_norm`, `total_neutr`), ce qui montre qu'il est très sensible aux attaques qui ciblent les marqueurs liés au genre ou qui neutralisent le signal. Les modèles temporels présentent un profil intermédiaire, avec une vulnérabilité particulière aux ruptures de cohérence séquentielle inter-modale (ΔBAcc jusqu'à 0.33 pour *Multimodal desync*).

Ces résultats répondent à la deuxième question de recherche en montrant que des perturbations comportementales réalistes, accessibles dans un environnement de réalité virtuelle sans accès au modèle, peuvent compromettre de manière substantielle la stabilité et la fiabilité des systèmes de reconnaissance du genre en réalité virtuelle. La diversité des profils de vulnérabilité observés implique que les stratégies de défense doivent être adaptées à chaque architecture : régularisation des features discriminantes pour les modèles tabulaires, renforcement de la cohérence séquentielle pour les architectures temporelles. Ces observations motivent naturellement l'étude, au chapitre suivant, de mécanismes de défense et de stratégies de robustification adaptés aux spécificités de chaque architecture.

Chapitre 7

Stratégies de Défense

Face aux menaces et perturbations identifiées dans le chapitre précédent, nous avons évalué un ensemble de stratégies de défense visant à maintenir la robustesse des modèles de prédiction du genre à partir de données de réalité virtuelle. Parmi l'ensemble des perturbations étudiées précédemment, nous avons sélectionné sept attaques représentatives couvrant différents vecteurs de perturbation : *unit mismatch*, *fixed gaze*, *hybrid morpho*, *gender normalization*, *total neutralization*, *nn counterfactual* et *behavioral drift*. Ces sept attaques ont été retenues car elles représentent les scénarios les plus menaçants et les plus diversifiés identifiés lors de la phase d'évaluation précédente. Les défenses ont ensuite été regroupées en quatre catégories distinctes : les transformations des données, les mécanismes de détection, les méthodes basées sur le vote, et les approches d'entraînement robuste. Diverses défenses ont été évaluées sur ces sept types d'attaques. Pour cette phase d'expérimentation, nous avons retenu exclusivement les modèles d'apprentissage automatique classiques, à savoir RandomForest, GradientBoosting, XGBoost, LightGBM, SVM et MLP. Ce choix se justifie par leur niveau de performance relativement élevé sur données non perturbées, ce qui en fait les candidats les plus pertinents pour une étude de robustesse face aux perturbations comportementales. Par ailleurs, ces modèles présentent l'avantage d'être plus interprétables et plus rapides à entraîner que les approches d'apprentissage profond, facilitant ainsi l'évaluation d'un grand nombre de configurations expérimentales dans un cadre contrôlé. L'objectif de ce chapitre est double. D'une part, il s'agit d'identifier les défenses les plus efficaces parmi les stratégies testées. D'autre part, nous proposons un système adaptatif de détection et de défense capable d'identifier automatiquement le type d'attaque subi et d'appliquer la stratégie de protection la plus appropriée selon le modèle et l'attaque concernés.

7.1 Performances de référence

Avant toute perturbation, les six modèles atteignent des niveaux de performance satisfaisants sur les données propres, comme le présente le Tableau 7.1. GradientBoosting et SVM obtiennent les meilleures performances avec une balanced accuracy de 0.93, tandis que XGBoost présente la performance la plus modeste avec 0.77. RandomForest, LightGBM et MLP se situent dans un intervalle intermédiaire entre 0.84 et 0.88. Ces résultats indiquent que les modèles basés sur des méthodes ensemblistes et à marge maximale offrent une meilleure capacité de généralisation sur les données VR non perturbées. Ils servent de référence pour évaluer l’impact des perturbations comportementales ainsi que l’efficacité des différentes stratégies de défense.

TABLEAU 7.1 – Performance de référence des six modèles sur données non perturbées

Modèle	BAcc
RandomForest	0.88
GradientBoosting	0.93
XGBoost	0.77
LightGBM	0.86
SVM (RBF)	0.93
MLP	0.84

7.1.1 Impact des attaques sans défense

Avant d’évaluer les défenses, il est nécessaire d’établir une référence en mesurant l’impact brut de chaque attaque sur les six modèles en l’absence de toute correction. La Figure 7.1 présente la BAcc obtenue pour chaque combinaison modèle $\times$ attaque dans ces conditions.

Plusieurs observations ressortent de cette baseline. *gender_normalization_agnostic* et *total_neutralization_agnostic* sont les attaques les plus dévastatrices pour GradientBoosting, dont la BAcc tombe à 0.477 dans les deux cas la valeur la plus basse observée toutes combinaisons confondues. À l’inverse, GradientBoosting résiste très bien à *atk_behavioral_drift*, *fixed_gaze* et *hybrid_morpho_agnostic* (0.886 dans les trois cas), révélant une sensibilité fortement dépendante du type d’attaque.

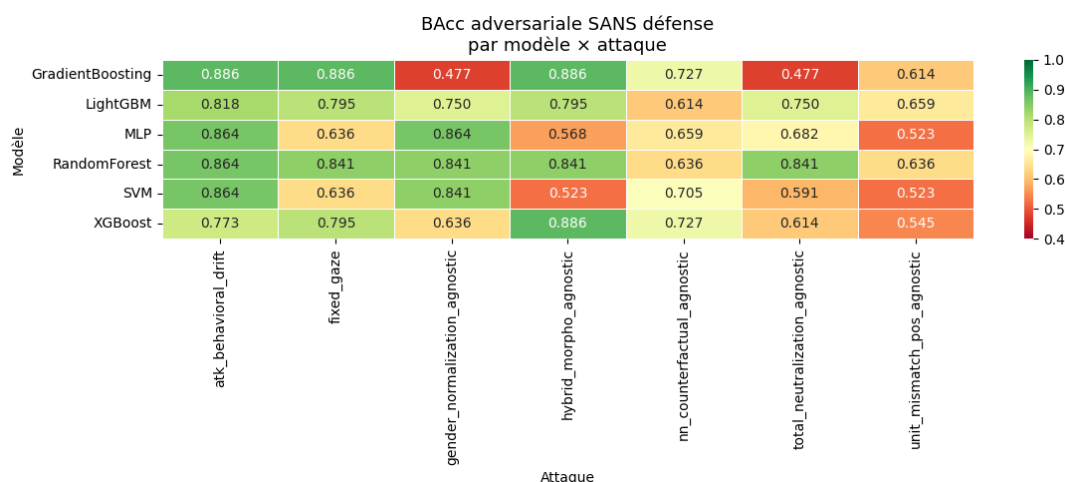


FIGURE 7.1 – BAcc sous perturbation sans défense par modèle et par attaque. Les cellules vertes indiquent une BAcc élevée (faible impact de l’attaque) et les cellules rouges une BAcc faible (fort impact).

RandomForest est le modèle le plus robuste : sa BAcc reste au-dessus de 0.636 pour toutes les attaques, avec cinq valeurs sur sept supérieures ou égales à 0.841. Le MLP et le SVM sont les plus vulnérables face à *unit_mismatch_pos_agnostic* (0.523 dans les deux cas) et face à *hybrid_morpho_agnostic* pour le SVM (0.523). LightGBM et XGBoost affichent des vulnérabilités intermédiaires, avec des BAcc descendant à 0.614 pour LightGBM (*nn_counterfactual_agnostic*) et à 0.545 pour XGBoost (*unit_mismatch_pos_agnostic*).

Cette variabilité des résultats selon le modèle et le type d’attaque justifie l’adoption d’une défense adaptative : une défense efficace contre une attaque peut être inutile, voire nuisible, contre une autre.

7.2 Évaluation des défenses

Cette section évalue l’efficacité de 19 défenses face aux perturbations comportementales testées. Chaque défense est analysée selon trois axes complémentaires : son gain moyen global, son comportement selon le type d’attaque, et son impact selon le modèle ciblé. Les figures d’impact individuel suivent toutes la même structure en trois panneaux : (1) la BAcc sur données propres (baseline clean), (2) la BAcc après attaque sans défense, et (3) la BAcc après application de la défense.

7.2.1 Vue d'ensemble : gain moyen par défense

Dans un premier temps, sept défenses ont été évaluées : Adversarial Training, LOF + Rejet(Local Outlier Factor), PCA Reconstruction(Analyse en Composantes Principales, ACP), Squeezing Défense, Smoothing, Ensemble et Median Filter. Le gain en BAcc est calculé comme la différence entre la BAcc après défense et la BAcc sans défense pour chaque combinaison (modèle, attaque). La Figure 7.2 présente ce gain agrégé sur l'ensemble des attaques et des modèles.

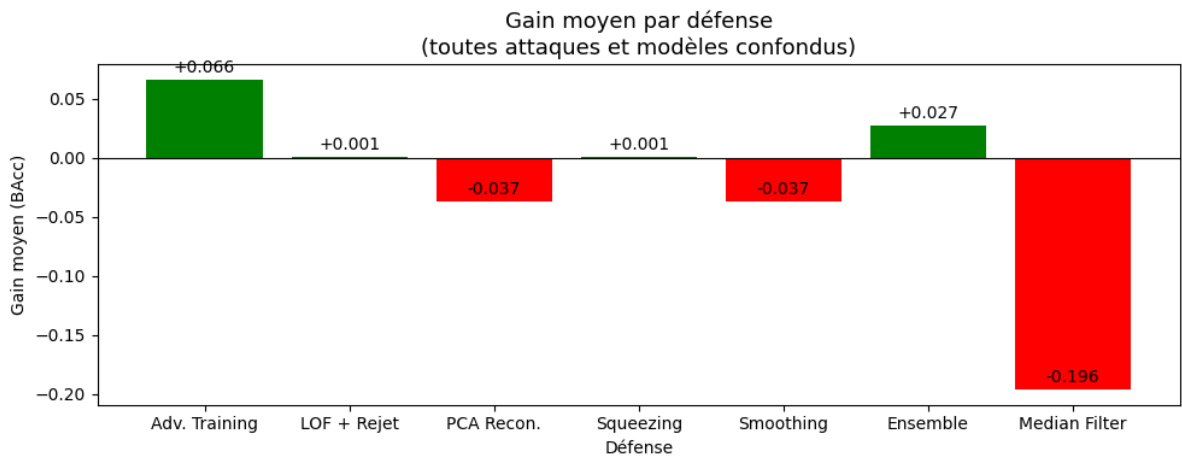


FIGURE 7.2 – Gain moyen en BAcc par défense.

Seuls l'Adversarial Training (+0.066) et l'Ensemble (+0.027) produisent un gain positif en moyenne. Le LOF + Rejet et le Squeezing Défense sont quasi neutres (+0.001 chacun). La PCA Reconstruction et le Smoothing dégradent les performances (−0.037 chacun), et le Median Filter est de loin la défense la plus nuisible avec un gain moyen de −0.196.

Ces résultats globaux masquent cependant une forte variabilité selon le type d'attaque et le modèle ciblé, que les deux sections suivantes permettent de décomposer.

Dans un deuxième temps, douze nouvelles défenses sont aussi évaluées dans un autre lot V2 : Winsorization, Z-score, Log Transform, Feature Normalization, OC-SVM, Mahalanobis, Statistical Test, Bagging, Subspace, Bootstrap, Label Smoothing et Mixup. La même méthodologie que pour le lot précédent est appliquée : le gain en BAcc est calculé comme la différence entre la BAcc après défense et la BAcc sans défense pour chaque combinaison modèle × attaque.

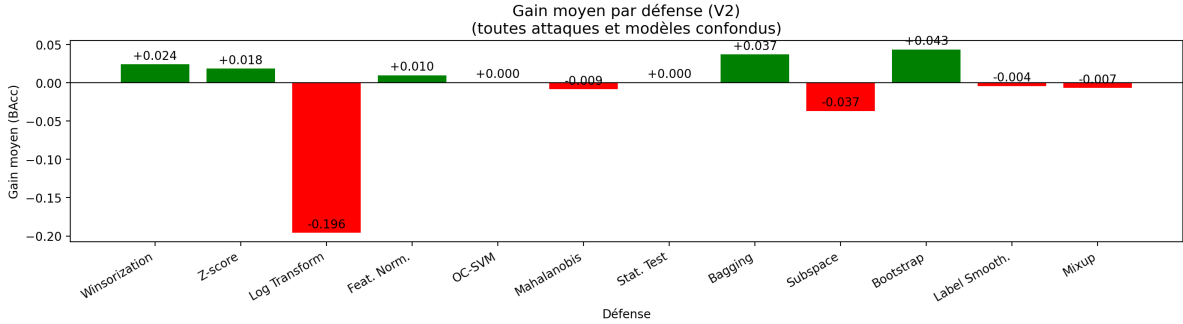


FIGURE 7.3 – Gain moyen en BAcc par défense (lot V2).

La Figure 7.3 présente le gain moyen de chaque défense, agrégé sur l’ensemble des attaques et des modèles. Cinq défenses produisent un gain positif en moyenne : Bootstrap (+0.043), Bagging (+0.037), Winsorization (+0.024), Z-score (+0.018) et Feature Normalization (+0.010). OC-SVM (One-Class SVM, SVM à une classe) et Statistical Test (test statistique de Kolmogorov-Smirnov) sont neutres (+0.000). Cinq défenses dégradent les performances : Label Smoothing (−0.004), Mixup (−0.007), Mahalanobis (−0.009), Subspace (−0.037) et Log Transform (−0.196). Contrairement au lot précédent où une seule défense (Adversarial Training) se détachait clairement, ce lot présente plusieurs défenses compétitives.

7.2.2 Analyse par type d’attaque et par modèle

Les Figures 7.4 et 7.5 décomposent le gain moyen selon deux axes complémentaires : le type d’attaque subi et le modèle ciblé.

L’Adversarial Training est la défense la plus cohérente : elle produit des gains positifs pour cinq attaques sur sept (+0.208 face à *unit_mismatch_pos_agnostic*, +0.144 face à *total_neutralization_agnostic*) et bénéficie à cinq modèles sur six (SVM +0.117, XGBoost +0.097, GradientBoosting +0.078, RandomForest +0.071). Elle est légèrement négative face à *atk_behavioral_drift* (−0.026) et *nn_counterfactual_agnostic* (−0.012), et constitue le seul cas de gain négatif pour le MLP (−0.030).

La PCA Reconstruction présente le comportement le plus contrasté selon l’attaque : très efficace face à *total_neutralization_agnostic* (+0.178) et *gender_normalization_agnostic* (+0.098), mais extrêmement nuisible face à *fixed_gaze* (−0.228) et *hybrid_morpho_agnostic* (−0.227). Elle est également très défavorable à

RandomForest (-0.091) et LightGBM (-0.078), ce qui confirme qu'elle ne peut pas être appliquée uniformément.

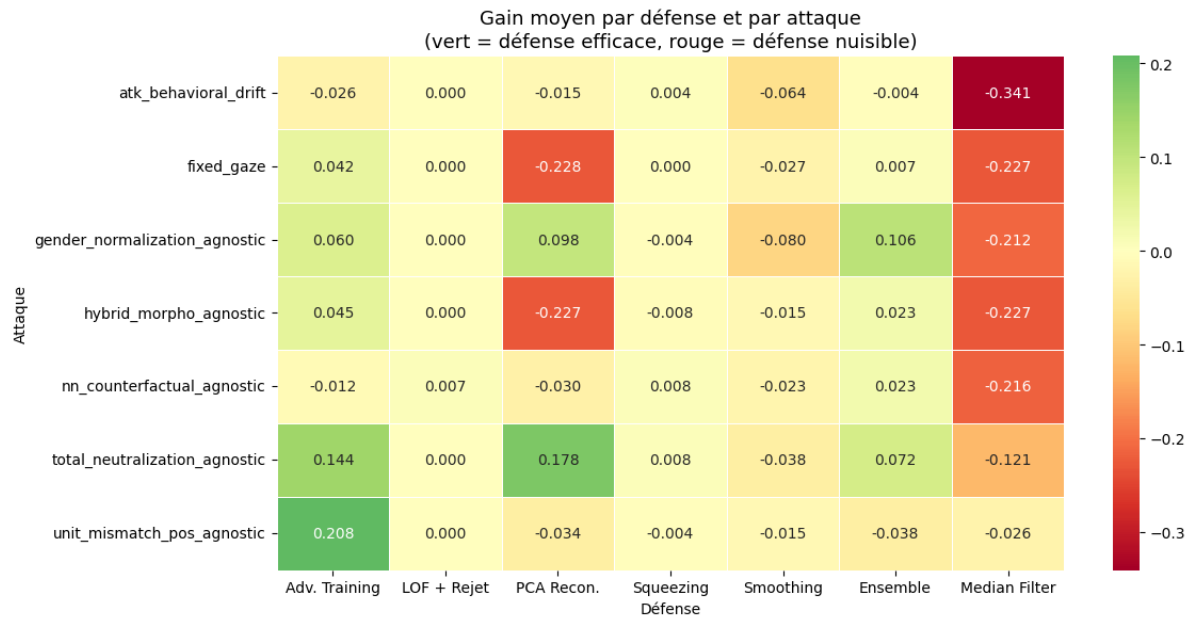


FIGURE 7.4 – Gain moyen en BAcc par défense et par type d'attaque (vert = défense efficace, rouge = défense nuisible).

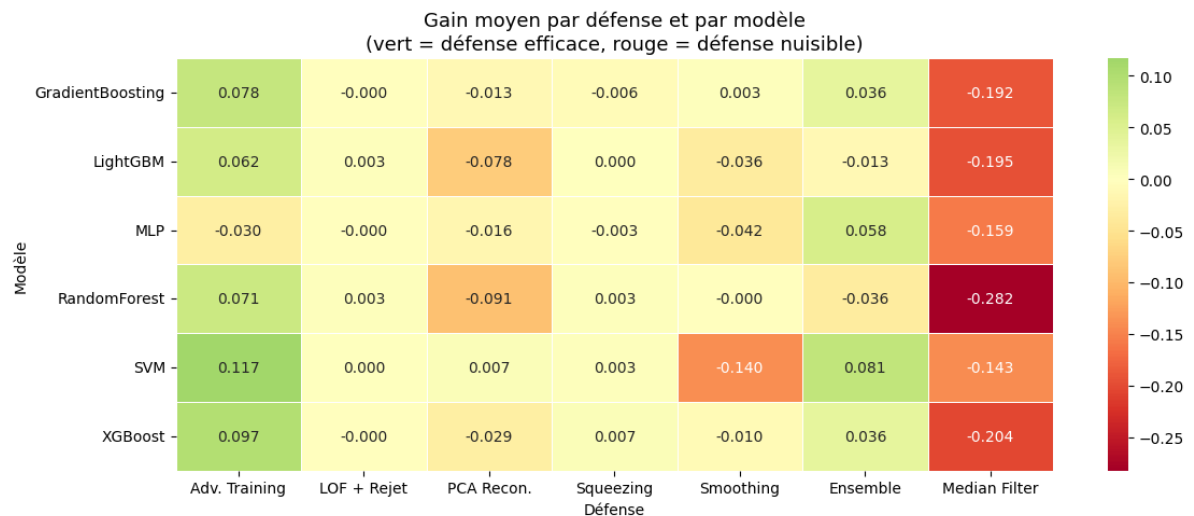


FIGURE 7.5 – Gain moyen en BAcc par défense et par modèle (vert = défense efficace, rouge = défense nuisible).

Le Smoothing bénéficie à GradientBoosting (+0.003 en moyenne) mais est particulièrement nuisible pour le SVM (−0.140). L'Ensemble produit les meilleures performances pour le MLP (+0.058) et des gains sélectifs face à *gender_normalization_agnostic* (+0.106) et *total_neutralization_agnostic* (+0.072). Le LOF + Rejet et le Squeezing Défense sont quasi neutres pour toutes les attaques et tous les modèles. Le Median Filter est systématiquement nuisible, avec des pertes allant de −0.143 (SVM) à −0.282 (RandomForest).

Les figures 7.6 et 7.7 mettent aussi en évidence des comportements très contrastés selon les défenses appliquées.

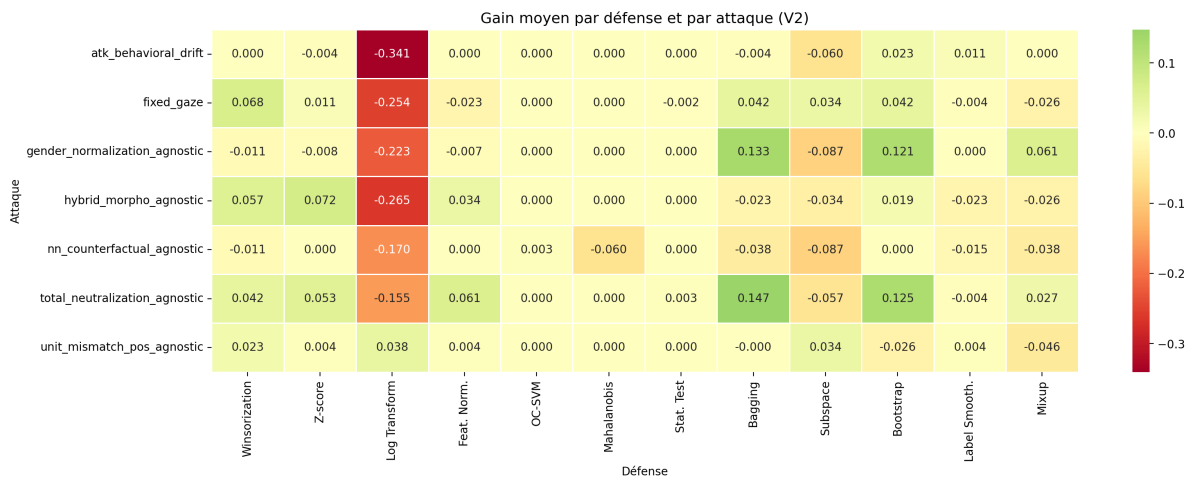


FIGURE 7.6 – Gain moyen en BAcc par défense et par type d'attaque (lot V2).

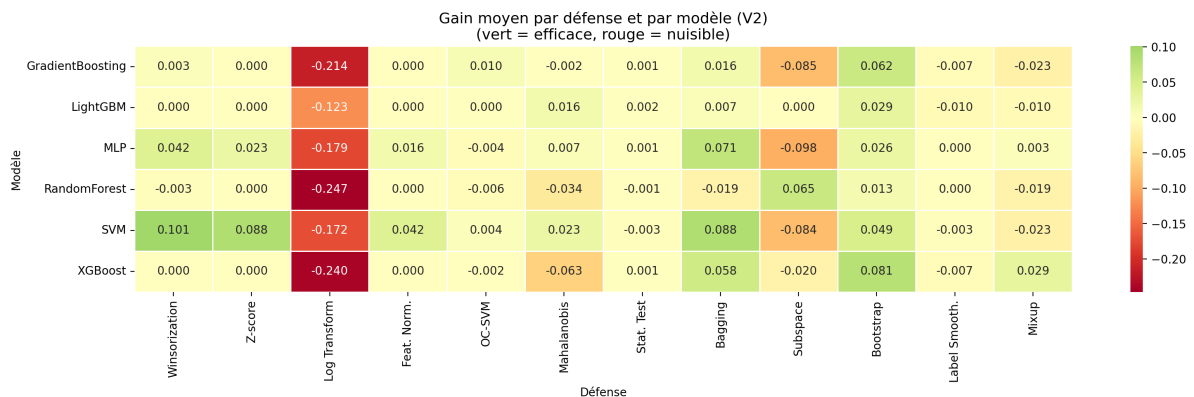


FIGURE 7.7 – Gain moyen en BAcc par défense et par modèle (lot V2).

La transformation logarithmique (*Log Transform*) se distingue comme la défense la plus nuisible, avec des gains fortement négatifs pour l'ensemble des attaques et des modèles, atteignant $-0,341$ pour `atk_behavioral_drift` et $-0,247$ pour le Random-Forest, ce qui suggère qu'elle déforme les caractéristiques d'une manière incompatible avec la structure des données. À l'inverse, les défenses d'agrégation par ensembles (*Bagging*, *Bootstrap*) apportent les gains les plus consistants, en particulier face aux attaques de neutralisation globale (`gender_normalization_agnostic` : $+0,133$ avec Bagging, $+0,121$ avec Bootstrap ; `total_neutralization_agnostic` : $+0,147$ et $+0,125$ respectivement), et se montrent également bénéfiques pour la majorité des modèles, notamment GradientBoosting ($+0,062$) et XGBoost ($+0,081$ avec Bootstrap).

La défense *Subspace*, en revanche, présente un comportement ambivalent : utile pour certains modèles comme RandomForest ($+0,065$), elle devient contre-productive pour d'autres, comme le MLP ($-0,098$) et le SVM ($-0,084$), ainsi que pour l'attaque `nn_counterfactual_agnostic` ($-0,087$). Enfin, des défenses comme *OC-SVM*, *Stat. Test* et *Label Smoothing* ont un effet globalement neutre, avec des gains proches de zéro quels que soient l'attaque ou le modèle considéré, ce qui indique une action trop limitée pour compenser les perturbations étudiées.

L'analyse des défenses appliquées séparément a montré que chacune d'elles agit sur un aspect précis des perturbations : la Winsorization élimine les valeurs extrêmes, le Z-score normalise les features anormales, la PCA reconstruit l'exemple dans l'espace des données propres, et le Bootstrap stabilise la prédiction par vote. Une question naturelle se pose alors : si chaque défense corrige une partie du problème, est-ce qu'appliquer plusieurs défenses l'une après l'autre permettrait de corriger davantage et d'obtenir un gain plus élevé ? C'est l'objet de la section suivante, qui évalue des empilements de défenses appliquées en séquence sur le même exemple perturbé.

7.2.3 Défenses empilées : combinaison séquentielle

Pour aller plus loin que les défenses évaluées séparément, nous avons testé l'idée d'appliquer plusieurs défenses l'une après l'autre sur le même exemple perturbé. Le principe est simple : au lieu d'appliquer une seule défense sur x_{adv} , on le nettoie progressivement à travers plusieurs étapes avant de prédire. Par exemple :

$$x_{adv} \rightarrow \text{Winsorization} \rightarrow \text{Z-score} \rightarrow \text{PCA} \rightarrow \text{Ensemble} \rightarrow \text{prédiction}$$

Chaque transformation élimine un type de perturbation différent : la Winsorization supprime les valeurs extrêmes, le Z-score normalise ce qui reste anormal, et la PCA projette l'exemple sur l'espace des données propres. La prédiction finale est ensuite faite sur un exemple nettoyé à plusieurs niveaux.

Nous avons organisé les empilements en trois niveaux selon le nombre de transformations appliquées avant la prédiction : le Niveau 2 avec une seule transformation, le Niveau 3 avec deux transformations en séquence, et le Niveau 4 avec trois transformations. Sept empilements ont été évalués au total.

7.2.3.1 Gain moyen global

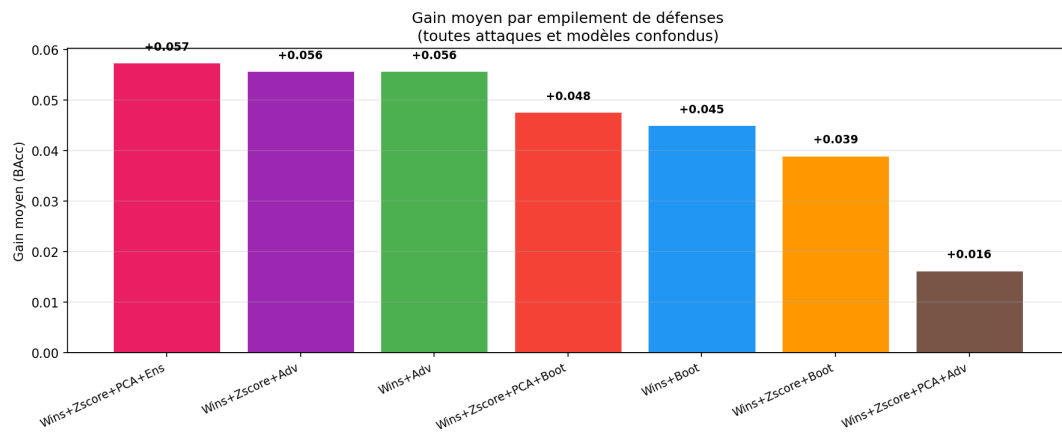


FIGURE 7.8 – Gain moyen en BAcc par empilement de défenses, toutes attaques et modèles confondus. Les empilements sont triés par gain décroissant.

La Figure 7.8 montre que tous les empilements testés produisent un gain positif. Le meilleur empilement est **Wins+Zscore+PCA+Ens** avec un gain moyen de +0.057, suivi de près par **Wins+Zscore+Adv** et **Wins+Adv** (+0.056 chacun). Le moins efficace est **Wins+Zscore+PCA+Adv** avec +0.016, ce qui montre que ajouter trop de transformations avant un modèle entraîné de façon robuste peut nuire : le nettoyage excessif supprime de l'information utile que le modèle robuste aurait pu exploiter.

7.2.3.2 Gain par modèle

La Figure 7.9 révèle des comportements différents selon les modèles. XGBoost et SVM bénéficient de presque tous les empilements, avec des gains allant jusqu'à +0.12 pour le

SVM avec **Wins+Zscore+Boot**. GradientBoosting profite également bien des empilements, surtout ceux incluant l'Adversarial Training ou l'Ensemble (+0.08 en moyenne). LightGBM présente des gains modérés mais stables entre -0.01 et $+0.06$.

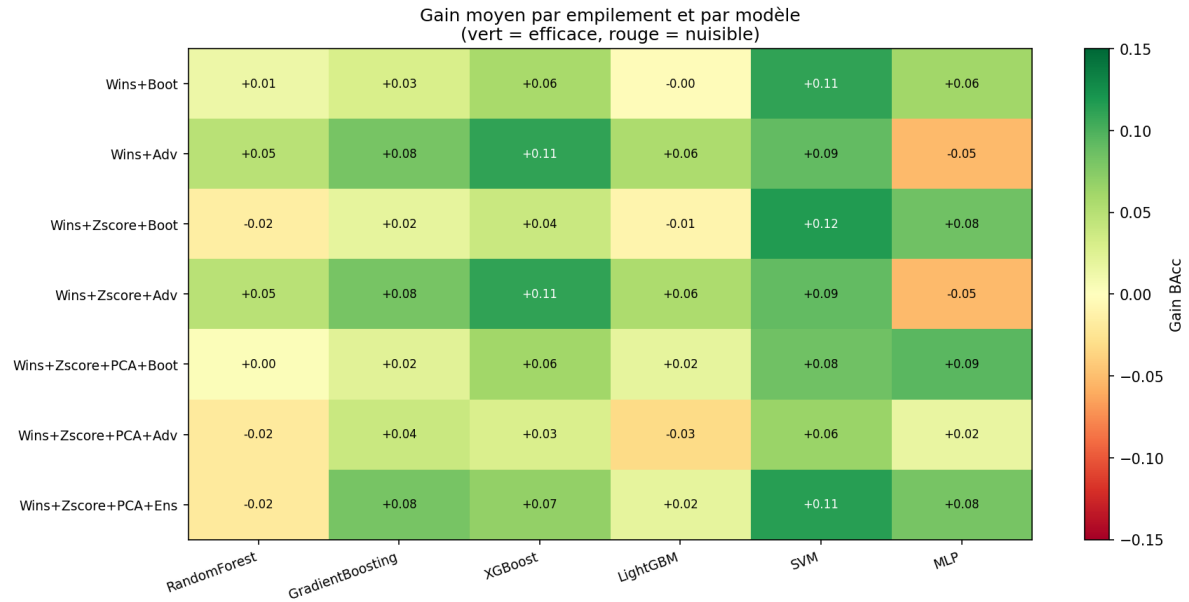


FIGURE 7.9 – Gain moyen par empilement et par modèle (vert = efficace, rouge = nuisible).

En revanche, le MLP est le seul modèle pour lequel certains empilements sont nuisibles : **Wins+Adv** et **Wins+Zscore+Adv** donnent -0.05 pour ce modèle. Cela confirme le phénomène d'overconfidence du MLP déjà observé avec l'Adversarial Training seul : combiner un nettoyage avec un modèle entraîné sur des exemples perturbés aggrave le problème pour ce modèle spécifique. RandomForest reste stable mais peu amélioré par les empilements (0.00 à $+0.05$).

7.2.3.3 Gain par attaque

La Figure 7.10 montre que les empilements sont très efficaces face à certaines attaques et moins face à d'autres. *total_neutralization_agnostic* est l'attaque pour laquelle les empilements apportent le plus de gain : **Wins+Zscore+PCA+Ens** atteint $+0.20$ et **Wins+Zscore+PCA+Boot** $+0.18$. *gender_normalization_agnostic* bénéficie aussi bien des empilements ($+0.13$ pour les niveaux 4).

À l’opposé, *nn_counterfactual_agnostic* et *atk_behavioral_drift* résistent aux empilements : la plupart des gains restent proches de zéro ou légèrement négatifs. *unit_mismatch_pos_agnostic* présente un résultat surprenant : **Wins+Adv** et **Wins+Zscore+Adv** donnent +0.20, mais **Wins+Zscore+PCA+Ens** tombe à −0.06, montrant qu’ajouter la PCA nuit pour cette attaque spécifique.

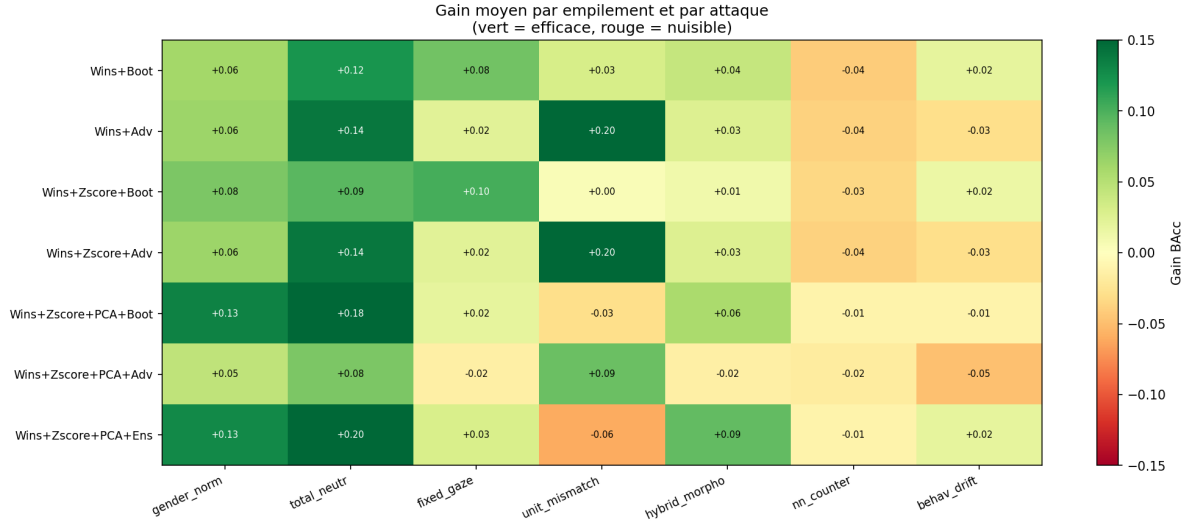


FIGURE 7.10 – Gain moyen par empilement et par type d’attaque (vert = efficace, rouge = nuisible).

7.2.3.4 Comparaison par niveau d’empilement

La Figure 7.11 compare les gains obtenus selon le nombre de transformations appliquées avant la prédiction finale. Au **Niveau 2**, on applique une seule transformation sur x_{adv} avant de prédire, par exemple : Winsorization puis Bootstrap. Au **Niveau 3**, on applique deux transformations l’une après l’autre, par exemple : Winsorization, puis Z-score, puis Bootstrap. Au **Niveau 4**, on applique trois transformations en séquence, par exemple : Winsorization, puis Z-score, puis PCA, puis Bootstrap. À chaque niveau, x_{adv} est donc nettoyé davantage avant que le modèle prédise.

La question est : est-ce qu’appliquer plus de transformations apporte plus de gain ?

La réponse est nuancée. Au Niveau 2, la meilleure combinaison est **Wins+Adv** avec +0.056. Au Niveau 3, le meilleur résultat est **Wins+Zscore+Adv** avec +0.056 également soit exactement le même gain qu’avec un seul nettoyage. Ajouter le Z-score avant l’Adversarial Training n’apporte rien de plus dans ce cas. Au Niveau 4, le meilleur

est **Wins+Zscore+PCA+Ens** avec +0.057, légèrement supérieur, mais au prix d'un nettoyage beaucoup plus lourd.

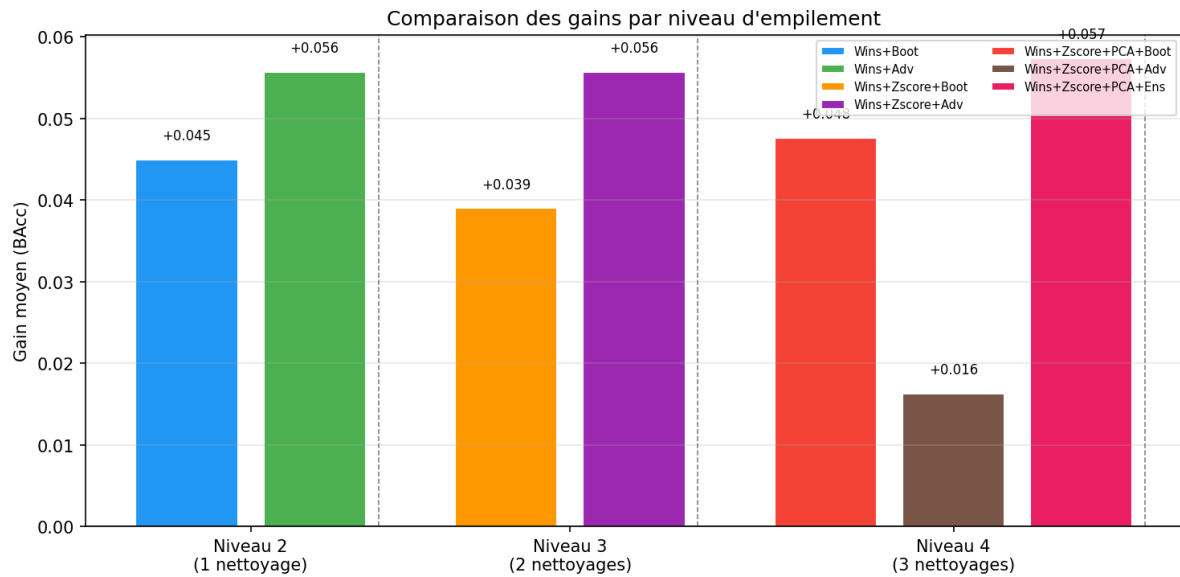


FIGURE 7.11 – Comparaison des gains par niveau d'empilement. Chaque groupe représente un niveau : Niveau 2 (une seule transformation avant la prédiction), Niveau 3 (deux transformations en séquence), Niveau 4 (trois transformations en séquence).

Ces résultats montrent que **empiler plus de défenses n'améliore pas toujours le gain**. Le meilleur rapport efficacité/complexité reste le Niveau 2 avec **Wins+Adv** (+0.056) ou le Niveau 4 avec **Wins+Zscore+PCA+Ens** (+0.057) si on accepte un temps de calcul plus élevé. Dans tous les cas, tous les empilements testés surpassent la meilleure défense seule (Bootstrap à +0.043), ce qui confirme que combiner les défenses apporte bien un gain réel.

Pour compléter l'analyse des empilements, les trois figures suivantes comparent directement la BAcc dans trois situations pour chaque modèle et chaque attaque : avant toute attaque (Clean, en vert), après attaque sans aucune défense (en rouge), et avec le meilleur empilement disponible (en bleu).

La Figure 7.12 montre pour chaque modèle l'effet de la meilleure combinaison de défenses empilées. Pour GradientBoosting, les attaques *gender_normalization* et *total_neutralization* font chuter la BAcc de 0.933 à 0.477 : le meilleur empilement la remonte à 0.861 et 0.886, ce qui représente une récupération presque totale. Pour le SVM, les dégradations les plus sévères (*fixed_gaze* à 0.591, *unit_mismatch* à 0.523)

sont bien corrigées par les empilements (0.864 et 0.800 respectivement). XGBoost bénéficie également de gains importants face à *gender_normalization* (0.636 $\rightarrow$ 0.909). En revanche, RandomForest et LightGBM sont peu améliorés : la barre bleue reste proche de la barre rouge pour la plupart des attaques, ce qui confirme que les empilements ne bénéficient pas de manière uniforme à tous les modèles. Le MLP présente un comportement contrasté : de bons gains sur certaines attaques (*behav_drift* à 0.930) mais des résultats proches de la situation sans défense sur d'autres.

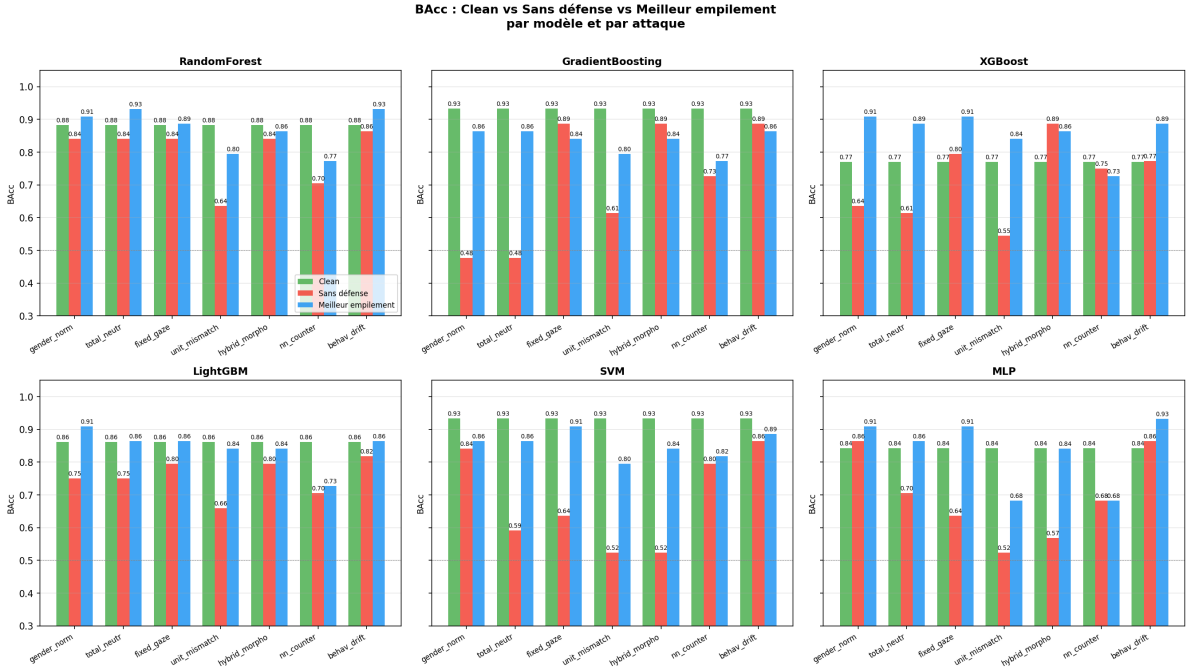


FIGURE 7.12 – BAcc avant attaque (Clean), après attaque sans défense, et avec le meilleur empilement, par modèle et par attaque.

La Figure 7.13 permet de voir quelles attaques sont bien corrigées par les empilements et lesquelles résistent. *total_neutralization* et *gender_normalization* sont les attaques pour lesquelles le meilleur empilement récupère le plus de BAcc : la barre bleue se rapproche systématiquement de la barre verte (Clean) pour tous les modèles. *fixed_gaze* et *unit_mismatch* sont également bien corrigées pour la plupart des modèles. À l’opposé, *nn_counterfactual* et *behav_drift* restent difficiles à défendre : la barre bleue ne dépasse pas beaucoup la barre rouge, ce qui confirme que ces deux attaques produisent des perturbations dont la signature est trop proche des données normales pour être bien détectée par les transformations de nettoyage.

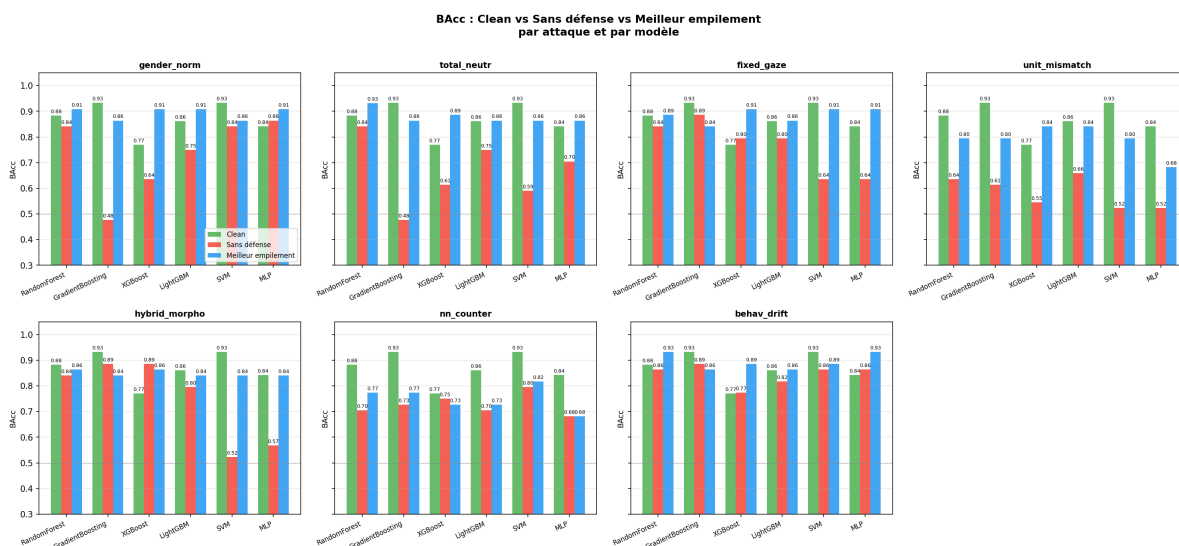


FIGURE 7.13 – BACC avant attaque (Clean), après attaque sans défense, et avec le meilleur empilement, par type d’attaque et par modèle.

La Figure 7.14 donne une vue globale en moyennant les résultats sur toutes les attaques. Deux observations ressortent clairement.

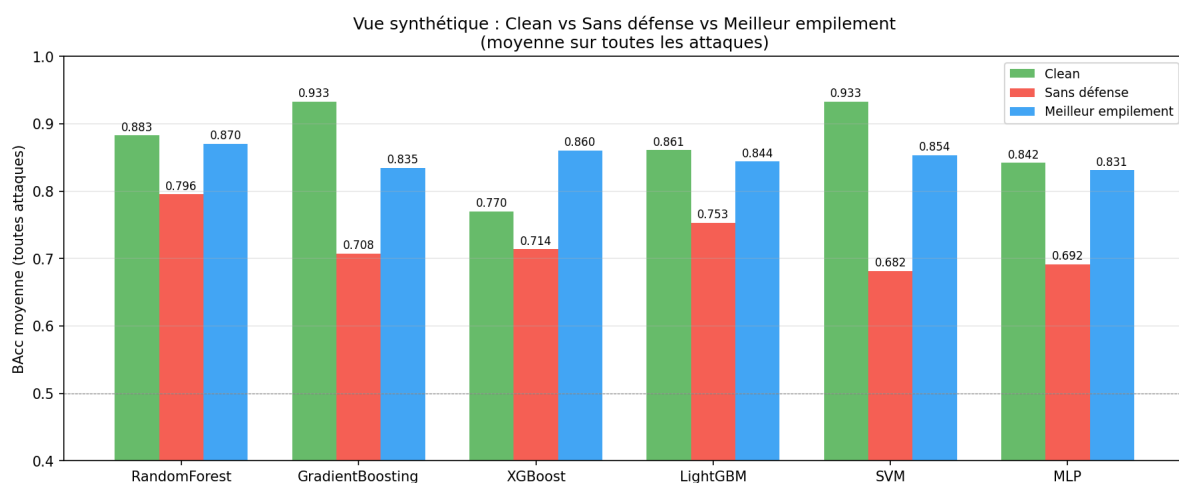


FIGURE 7.14 – BACC moyenne sur toutes les attaques, pour les trois situations (Clean, Sans défense, Meilleur empilement), par modèle.

Premièrement, tous les modèles sont améliorés par le meilleur empilement par rapport à la situation sans défense. Le gain le plus important concerne XGBoost (0.714 → 0.860,

gain de +0.146) et le SVM (0.682 $\rightarrow$ 0.854, gain de +0.172), qui sont les modèles les plus vulnérables aux attaques mais aussi les plus bien défendus par les empilements.

Deuxièmement, aucun modèle ne retrouve entièrement sa BAcc d'origine. L'écart entre la barre verte (Clean) et la barre bleue (Meilleur empilement) reste visible pour tous les modèles, en particulier pour GradientBoosting (0.933 clean contre 0.835 défendu) et le SVM (0.933 clean contre 0.854 défendu). Ce résidu confirme que les empilements de défenses améliorent significativement la robustesse mais ne restaurent pas complètement les performances nominales, ce qui justifie le développement d'un système adaptatif capable d'ajuster dynamiquement la stratégie de défense selon le contexte.

L'analyse des défenses appliquées séparément puis en empilement séquentiel permet de répondre à la question posée : oui, combiner plusieurs défenses l'une après l'autre apporte un gain supplémentaire. Le meilleur empilement **Wins+Zscore+PCA+Ens** atteint +0.057, contre +0.043 pour le Bootstrap seul, ce qui confirme que nettoyer progressivement x_{adv} à travers plusieurs transformations avant de prédire améliore la robustesse. Cependant, cette amélioration reste modeste : ajouter deux ou trois transformations supplémentaires ne multiplie pas le gain, et dans certains cas comme le MLP ou l'attaque *unit_mismatch* avec la PCA, empiler trop de défenses peut même nuire.

7.3 Système Adaptatif de Détection et Défense

L'étude des différentes défenses connues démontre qu'aucune défense, qu'elle soit appliquée seule ou en séquence, ne s'avère efficace face à l'ensemble des attaques testées. La meilleure combinaison pour un modèle ou une attaque donnée peut se révéler inefficace, voire contre-productive, dans un autre contexte. Ce constat conduit à formuler l'hypothèse suivante : puisqu'aucune défense statique ne peut couvrir l'ensemble du spectre des attaques, un système capable de reconnaître automatiquement le type d'attaque en cours et d'adapter sa réponse en conséquence devrait permettre d'améliorer significativement la robustesse globale, en combinant les forces spécifiques de chaque défense plutôt qu'en misant sur une solution unique. Cette approche se place du point de vue du concepteur du système : celui qui développe et maintient le modèle a accès aux signatures et aux seuils caractéristiques de chaque attaque, identifiés au fil des analyses précédentes, et peut donc corriger les failles détectées à mesure qu'elles sont découvertes. Plutôt que de subir passivement les perturbations, le système adaptatif exploite cette connaissance

interne pour reconnaître automatiquement le type d'attaque en cours et appliquer la défense la plus adaptée.

7.3.1 Architecture du Pipeline

Le système adaptatif repose sur sept détecteurs statistiques construits à partir des 682 features des données d'entraînement, organisées en cinq groupes thématiques potentiellement chevauchants : 151 features de position, 270 features de regard, 153 features de variabilité, 141 marqueurs de genre et 174 features dynamiques. Il est à noter que les six modèles n'utilisent pas l'ensemble des 682 features disponibles : une sélection par *SelectKBest* est appliquée lors de l'optimisation des hyperparamètres, retenant respectivement 200 features pour RandomForest, 150 pour GradientBoosting, 100 pour SVM, 150 pour MLP, 589 pour XGBoost et 150 pour LightGBM. Les détecteurs du système adaptatif, quant à eux, opèrent sur l'ensemble des 682 features afin de capturer les signatures caractéristiques de chaque type de perturbation comportementale.

Ce système s'organise autour d'un pipeline en trois étapes successives, illustrées dans l'Équation 7.1 :

$$\mathbf{x}_{adv} \xrightarrow{\text{Détection}} \hat{a} \xrightarrow{\text{Sélection}} \delta^*(\hat{a}) \xrightarrow{\text{Défense}} \hat{y} \quad (7.1)$$

où $\mathbf{x}_{adv}$ est l'exemple potentiellement perturbé, $\hat{a}$ est le type de perturbation identifié, $\delta^*(\hat{a})$ est la défense sélectionnée et $\hat{y}$ est la prédiction finale.

Étape 1 : Détection par signatures statistiques : La détection repose sur l'analyse des signatures laissées par chaque type de perturbation dans l'espace des features. Pour chaque exemple $\mathbf{x}_{adv}$, le détecteur calcule des statistiques sur les cinq groupes de features et les compare aux statistiques de référence calculées sur les données d'entraînement (μ , σ). Les sept règles de détection sont appliquées séquentiellement dans l'ordre suivant :

1. **Unit Mismatch** : détectée si le z-score moyen des features de position dépasse le seuil de 100 :

$$\bar{z}_{pos} = \frac{1}{|\mathcal{P}|} \sum_{j \in \mathcal{P}} \frac{|x_j - \mu_j|}{\sigma_j} > 100 \quad (7.2)$$

où $\mathcal{P}$ désigne l'ensemble des 151 indices de features de position.

2. **Fixed Gaze** : détectée si plus de 80% des features de regard sont inférieures à 0.05 en valeur absolue :

$$\frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{1}[|x_j| \leq 0.05] > 0.80 \quad (7.3)$$

où $\mathcal{G}$ désigne l'ensemble des 270 indices de features de regard.

3. **Hybrid Morpho** : détectée si plus de 50% des features de regard ont changé de signe par rapport à leur moyenne d'entraînement :

$$\frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{1}[\text{sign}(x_j) \neq \text{sign}(\mu_j)] > 0.50 \quad (7.4)$$

4. **Total Neutralization** : détectée si le ratio médian des marqueurs de genre se situe dans l'intervalle (0.40, 0.65) :

$$r_{mark} = \text{median} \left(\frac{x_j}{\mu_j} \right)_{j \in \mathcal{M}, \mu_j \neq 0} \in (0.40, 0.65) \quad (7.5)$$

où $\mathcal{M}$ désigne l'ensemble des 141 indices de marqueurs de genre.

5. **Gender Normalization** : détectée si le ratio médian des marqueurs de genre se situe dans l'intervalle (0.75, 0.90) :

$$r_{mark} \in (0.75, 0.90) \quad (7.6)$$

6. **Behavioral Drift** : détectée si le ratio médian des features dynamiques se situe dans l'intervalle (1.01, 1.15) :

$$r_{dyn} = \text{median} \left(\frac{x_j}{\mu_j} \right)_{j \in \mathcal{D}, \mu_j \neq 0} \in (1.01, 1.15) \quad (7.7)$$

où $\mathcal{D}$ désigne l'ensemble des 174 indices de features dynamiques.

7. **NN Counterfactual** : détectée par élimination si le z-score global moyen est modéré :

$$\bar{z}_{global} = \frac{1}{d} \sum_{j=1}^d \frac{|x_j - \mu_j|}{\sigma_j} \in (1.0, 5.0) \quad (7.8)$$

où d est le nombre total de features.

Si aucune règle n'est activée, l'exemple est considéré comme normal et le modèle prédit directement sans défense.

Étape 2 : Sélection de la défense : Une fois le type de perturbation identifié, le système consulte un dictionnaire $\delta^* : \mathcal{A} \rightarrow \mathcal{F}$ associant chaque type de perturbation à la meilleure défense identifiée lors de l'évaluation précédente.

Défense choisie par type d'attaque

Défense	adversarial_training	ensemble	none	pca	smoothing
unit_mismatch_pos_agnostic	1	0	0	0	0
fixed_gaze	0	1	0	0	0
gender_normalization_agnostic	0	0	0	1	0
total_neutralization_agnostic	0	0	0	1	0
hybrid_morpho_agnostic	0	1	0	0	0
nn_counterfactual_agnostic	0	0	0	0	1
att_behavioral_drift	1	0	0	0	0
normal	0	0	1	0	0
Attaque					

FIGURE 7.15 – Association entre chaque type d'attaque détecté et la défense sélectionnée dans le dictionnaire δ^* . La valeur 1 (case verte) indique que la défense correspondante est celle choisie pour cette attaque, 0 sinon.

La Figure 7.15 illustre ce dictionnaire de manière visuelle : chaque attaque est associée à exactement une défense, choisie d'après le gain moyen le plus élevé observé lors de l'évaluation préliminaire des défenses fixes.

Étape 3 : Application de la défense : La défense sélectionnée est appliquée à l'exemple $\mathbf{x}_{adv}$ avant la prédiction finale. Quatre mécanismes de défense sont disponibles dans V1 :

- **Adversarial Training** : le modèle robuste f_{θ^*} , réentraîné sur un dataset augmenté incluant des exemples perturbés, est utilisé directement pour la prédiction.
- **Reconstruction PCA** : l'exemple est projeté sur le sous-espace des composantes principales extraites des données propres, supprimant les perturbations orthogonales à ce sous-espace.
- **Ensemble de modèles** : la prédiction est obtenue par vote majoritaire de l'ensemble des six modèles entraînés.

- **Gaussian Smoothing** : la prédiction est obtenue par vote majoritaire sur 25 versions bruitées de l'exemple ($\mathcal{N}(0, 0.05)$).

7.3.2 Évaluation

7.3.2.1 Pipeline entraînement/test

Le pipeline repose sur une séparation stricte entre données d'entraînement et données de test, organisée en deux phases successives et indépendantes.

Phase 1 - Préparation (une seule fois, sur $\mathcal{X}_{\text{train}}$) : Cette phase est exécutée **une seule fois**, avant toute évaluation, et ne touche jamais $\mathcal{X}_{\text{test}}$. Elle produit quatre composantes réutilisées ensuite pour tous les exemples de test :

1. **Modèles de base** : les six classifieurs (Random Forest, Gradient Boosting, SVM, XGBoost, LightGBM, MLP) sont entraînés normalement sur $\mathcal{X}_{\text{train}}$. Ce sont les modèles utilisés par défaut, et également ceux sollicités par la défense *Ensemble* ;
2. **Statistiques de référence** : pour chaque feature, on calcule la moyenne μ_{train} , l'écart-type σ_{train} et les percentiles sur $\mathcal{X}_{\text{train}}$. Ces statistiques décrivent le comportement « normal » et servent de référence au détecteur ;
3. **PCA de référence** : une analyse en composantes principales est ajustée sur $\mathcal{X}_{\text{train}}$, définissant le sous-espace de forte variance correspondant au signal comportemental normal. Elle sera utilisée par la défense *PCA Reconstruction* ;
4. **Modèle robuste : une seule fois**, des versions attaquées de $\mathcal{X}_{\text{train}}$ sont générées ($\mathcal{X}_{\text{train}}^{\text{adv}}$), et un nouveau modèle (**robust_model**) est entraîné sur $\mathcal{X}_{\text{train}} \cup \mathcal{X}_{\text{train}}^{\text{adv}}$. C'est le **seul réentraînement** du pipeline ; il sera utilisé par la défense *Adversarial Training*.

Phase 2 - Évaluation (répétée pour chaque exemple de $\mathcal{X}_{\text{test}}$) : Aucun ré-entraînement n'a lieu dans cette phase : les quatre composantes de la Phase 1 sont simplement **réutilisées**. Pour chaque exemple $x_i \in \mathcal{X}_{\text{test}}$, une perturbation est d'abord appliquée pour produire $x_i^{\text{adv}} = f_{\text{attaque}}(x_i)$ (pour les attaques contrefactuelles telles que **nn_counterfactual** ou **cross_class_blending**, $\mathcal{X}_{\text{train}}$ sert de pool de référence pour simuler un attaquant en contexte boîte grise, *grey-box*). Le système de détection et de défense décrit en Section 7.3.1 est ensuite appliqué à x_i^{adv} , et la prédiction finale $\hat{y}_i$ est comparée à y_i afin de mettre à jour les métriques de gain et de taux de détection.

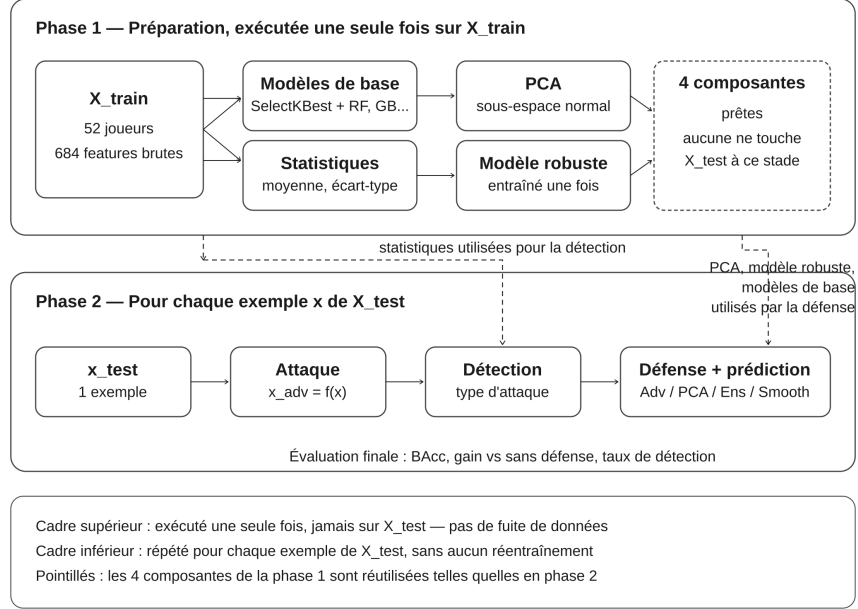


FIGURE 7.16 – Pipeline du système adaptatif de détection et défense.

La Figure 7.16 résume ce pipeline en deux blocs. Le bloc supérieur correspond à la Phase 1 : il est exécuté une seule fois sur $\mathcal{X}_{train}$ et produit quatre composantes (modèles de base, statistiques, PCA, modèle robuste). Le bloc inférieur correspond à la Phase 2 : il est répété pour chaque exemple de $\mathcal{X}_{test}$, en appliquant successivement l'attaque, la détection et la défense décrites en Section 7.3.1, puis l'évaluation. Les flèches en pointillés montrent que les composantes de la Phase 1 sont simplement réutilisées en Phase 2, sans aucun nouveau calcul ni réentraînement.

7.3.2.2 Taux de détection

La Table 7.2 présente les taux de détection obtenus par le système adaptatif pour chaque type d'attaque.

Cinq attaques sur sept sont correctement identifiées avec un taux supérieur au seuil acceptable de 70%. Les attaques à signature forte *unit mismatch*, *fixed gaze* et *hybrid morpho* sont détectées avec un taux parfait de 100%, tandis que *gender normalization* et *total neutralization* atteignent 97.7%. En revanche, *nn counterfactual* reste difficile à identifier (30.7%) en raison de sa nature contrefactuelle qui produit des exemples indiscernables des données normales. *Behavioral drift* est partiellement détectée (38.6%).

TABLEAU 7.2 – Taux de détection du système adaptatif par type d’attaque

Attaque	Taux de détection	Statut
unit_mismatch	100.0%	Bien détectée
fixed_gaze	100.0%	Bien détectée
hybrid_morpho	100.0%	Bien détectée
gender_normalization	97.7%	Bien détectée
total_neutralization	97.7%	Bien détectée
behavioral_drift	38.6%	Partiellement
nn_counterfactual	30.7%	Partiellement
Moyenne	76.3%	

7.3.2.3 Gain de robustesse

Le Tableau 7.3 présente les gains de robustesse obtenus par le système adaptatif pour chaque modèle. Le gain moyen global est de +0.081, surpassant nettement les meilleures défenses évaluées, dont le Bootstrap Ensemble (+0.043) et l’Adversarial Training (+0.033).

TABLEAU 7.3 – Gain moyen du système adaptatif par modèle (moyenne sur les 7 attaques)

Modèle	BAcc clean	BAcc sans défense	BAcc	Gain
RandomForest	0.883	0.773	0.812	+0.040
GradientBoosting	0.933	0.732	0.841	+0.109
XGBoost	0.770	0.713	0.805	+0.092
LightGBM	0.861	0.747	0.809	+0.062
SVM	0.933	0.666	0.816	+0.150
MLP	0.842	0.719	0.815	+0.096
Moyenne	0.870	0.725	0.816	+0.081

Les gains les plus importants sont observés pour le SVM (+0.150) et GradientBoosting (+0.109), deux modèles particulièrement vulnérables aux attaques de neutralisation qui bénéficient fortement de la reconstruction PCA appliquée sur les attaques *gender normalization* (+0.318 pour GB) et *total neutralization* (+0.386 pour GB, +0.295 pour SVM). Le MLP (+0.096) et XGBoost (+0.092) présentent également des améliorations significatives, notamment grâce à l’*adversarial training* appliqué sur *unit mismatch*. LightGBM (+0.062) et RandomForest (+0.040) affichent des gains plus modestes, ces

modèles étant déjà relativement robustes sur certaines attaques. Contrairement à la version précédente, aucun modèle ne présente de gain négatif, ce qui indique une meilleure adéquation entre les défenses sélectionnées et les vulnérabilités de chaque modèle.

7.3.3 Limitations :

Deux limitations principales sont identifiées. Premièrement, *nn counterfactual* n'est que faiblement détectée (30.7%) : dans la majorité des cas, le système applique alors la prédiction directe sans défense (*none*), laissant ces exemples largement sans protection. Deuxièmement, *behavioral drift* n'est que partiellement détectée (38.6%), et la défense appliquée (*none* dans 61.4% des cas) s'avère insuffisante pour protéger les modèles contre cette attaque subtile.

C'est précisément pour répondre à ces deux limitations qu'ont été développées les versions suivantes du système, V2, V3, V4 et V5, chacune apportant des corrections ciblées : l'introduction de détecteurs spécifiques pour les attaques contrefactuelles et pour la dérive comportementale progressive. La version V1 est présentée ici en détail car elle obtient le meilleur gain moyen de robustesse parmi les cinq versions testées ; les versions suivantes améliorent le taux de détection sans toutefois dépasser ce gain de BAcc. Ces versions sont présentées en annexe (voir Annexe C.1, Annexe C.2, Annexe C.3 et Annexe C.4).

7.4 Bilan Comparatif : Système Adaptatif vs Défenses Individuelles

Le système adaptatif, dont les détails sont présentés dans les sous-sections précédentes, constitue la meilleure version en termes de gain de robustesse (+0.081), surpassant l'ensemble des défenses individuelles évaluées dans les deux lots précédents. Parmi celles-ci, l'Adversarial Training (+0.066) et le Bootstrap Ensemble (+0.043) constituent les meilleures défenses statiques, suivis du Bagging (+0.037) et de l'Ensemble (+0.027), tandis que plusieurs défenses restent proches de zéro (LOF + Rejet, Squeezing, OC-SVM, Stat. Test) ou nettement négatives (Median Filter et Log Transform, -0.196 chacune ; PCA Recon. et Smoothing, -0.037 ; Subspace, -0.037).

Puisqu'on choisit le système adaptatif, il faut aussi vérifier s'il rend les prédictions plus fiables, pas seulement plus précises. Pour ça, on a calculé la confiance de prédiction présentée dans la section suivante.

7.5 Analyse de la confiance de prédiction

Au-delà du gain de robustesse, nous analysons la confiance de prédiction des modèles à chaque étape du pipeline : avant attaque, après attaque et après application de chaque version du système adaptatif. La confiance est définie comme la probabilité maximale attribuée par le modèle à l'une des deux classes :

$$c(\mathbf{x}) = \max_{y \in \{0,1\}} P(Y = y \mid \mathbf{x}) \quad (7.9)$$

Pour les modèles à vote (Ensemble, Smoothing), la confiance est calculée comme la proportion de votes pour la classe majoritaire.

Afin de vérifier plus précisément l'origine de ces écarts entre confiance et BAcc, le Tableau 7.4 décompose la confiance moyenne selon que la prédiction associée était correcte ou fausse, pour les étapes « Sans défense » et « système adaptatif ».

TABLEAU 7.4 – Confiance moyenne selon la justesse de la prédiction (Clean vs Sans défense vs système adaptatif)

Modèle	Clean		Sans défense		Système adaptatif	
	Conf.correct	Conf.faux	Conf.correct	Conf.faux	Conf.correct	Conf.faux
RandomForest	0.682	0.548	0.644	0.553	0.724	0.589
GradientBoosting	0.866	0.751	0.864	0.756	0.888	0.738
XGBoost	0.817	0.642	0.784	0.701	0.835	0.679
LightGBM	0.899	0.792	0.889	0.771	0.889	0.758
SVM	0.697	0.548	0.676	0.586	0.756	0.609
MLP	0.986	0.840	0.963	0.938	0.948	0.863

L'écart entre la confiance moyenne sur les prédictions correctes et sur les prédictions fausses permet de quantifier directement la fiabilité de calibration de chaque modèle. Sans défense, cet écart varie de 0.083 (XGBoost) à 0.118 (LightGBM) pour la majorité des modèles, mais chute à seulement 0.025 pour le MLP (0.963 contre 0.938), confirmant que ce dernier reste presque aussi confiant en se trompant qu'en ayant raison. Ce déficit de calibration est déjà présent sur données propres (0.146 d'écart pour le MLP, contre

0.115 pour RandomForest et 0.149 pour SVM par exemple) et persiste après application du système adaptatif (écart de 0.085 pour le MLP), alors que les autres modèles voient généralement cet écart se maintenir ou légèrement s'améliorer (par exemple RandomForest : 0.135 avec le système adaptatif contre 0.091 sans défense). Plusieurs observations importantes se dégagent de cette analyse :

Overconfidence du MLP : Le MLP présente un phénomène d'*overconfidence* particulièrement marqué : sa confiance sur les prédictions correctes (0.963) reste quasi identique à celle sur données propres (0.986), alors que sa confiance sur les erreurs reste très élevée (0.938), un écart de seulement 0.025 point entre les deux cas. Ce phénomène indique que les perturbations comportementales orientent les exemples vers des régions de l'espace de features où le MLP est très confiant mais systématiquement erroné. Après application de la défense, la confiance sur erreurs diminue légèrement à 0.863 (contre 0.948 sur les prédictions correctes), un écart qui reste faible (0.085) comparé aux autres modèles, confirmant que le déficit de calibration du MLP persiste malgré l'amélioration de la BAcc apportée par le système adaptatif.

Cohérence confiance/BAcc pour RandomForest et SVM : RandomForest et SVM présentent un comportement inverse au MLP, témoignant d'une meilleure cohérence entre confiance et précision. Pour RandomForest, la confiance sur les prédictions correctes chute de 0.682 (clean) à 0.644 (sans défense), tandis que la confiance sur les erreurs reste stable autour de 0.548–0.553 : l'écart entre les deux états (0.091 sans défense) reste nettement plus marqué que celui du MLP. De même, le SVM présente un écart de 0.090 sans défense (0.676 contre 0.586), confirmant dans les deux cas une séparation plus nette entre confiance sur prédictions correctes et confiance sur erreurs, contrairement au MLP dont les deux valeurs restent proches. Après application du système adaptatif, RandomForest atteint une confiance de 0.724 sur ses prédictions correctes contre 0.589 sur ses erreurs (écart de 0.135), tandis que le SVM obtient 0.756 contre 0.609 (écart de 0.147), confirmant dans les deux cas une amélioration de la calibration en plus de la performance.

GradientBoosting : overconfidence modérée : GradientBoosting présente un phénomène d'*overconfidence* modéré : l'écart entre confiance sur prédictions correctes et confiance sur erreurs reste resserré, que ce soit sur données propres (0.115, 0.866 contre

0.751) ou sans défense (0.108, 0.864 contre 0.756). Ce comportement indique que GradientBoosting reste relativement confiant dans ses prédictions indépendamment de leur exactitude. Après application du système adaptatif, l'écart s'élargit légèrement à 0.150 (0.888 contre 0.738), ce qui constitue une amélioration de la calibration par rapport à la situation sans défense, sans pour autant atteindre le niveau de séparation observé pour RandomForest ou le SVM.

XGBoost et LightGBM : amélioration avec le système adaptatif : XGBoost et LightGBM présentent des améliorations notables avec le système adaptatif. Pour XGBoost, l'écart entre confiance sur prédictions correctes et confiance sur erreurs passe de 0.083 sans défense (0.784 contre 0.701) à 0.156 avec le système adaptatif (0.835 contre 0.679), traduisant une nette amélioration de la calibration en plus du gain de performance. Pour LightGBM, l'écart passe de 0.118 sans défense (0.889 contre 0.771) à 0.131 avec le système adaptatif (0.889 contre 0.758), une amélioration plus modeste mais dans la même direction. Dans les deux cas, le système adaptatif offre un bon compromis confiance/précision par rapport à la situation sans défense.

7.6 Synthèse et Choix Final

L'évaluation comparative des défenses et des cinq versions du système adaptatif permet de dégager plusieurs conclusions majeures.

Parmi les défenses, l'Adversarial Training constitue la stratégie la plus efficace avec un gain moyen de +0.066, suivi du Bootstrap Ensemble (+0.043) et du Bagging (+0.037). Les mécanismes de détection et rejet s'avèrent en revanche inutilisables en pratique, en raison d'un taux d'abstention trop élevé. Les transformations destructrices telles que le filtrage médian (−0.196) et la transformation logarithmique (−0.196) nuisent significativement aux performances.

L'analyse des défenses révèle une hétérogénéité fondamentale : aucune stratégie unique n'est optimale pour l'ensemble des attaques et des modèles. Cette observation motive le développement du système adaptatif, qui identifie automatiquement le type d'attaque et sélectionne dynamiquement la défense la plus appropriée selon le modèle concerné.

Au terme de cette analyse, le système adaptatif est retenu comme solution finale. Ce choix repose sur deux arguments. Premièrement, il obtient le gain moyen (+0.081),

surpassant nettement la totalité des défenses statiques évaluées, y compris les plus performantes d'entre elles, le Bootstrap Ensemble (+0.043) et le Bagging (+0.037). Deuxièmement, et plus fondamentalement, c'est ce gain de robustesse effectif et non le taux de détection en tant que tel qui constitue le critère déterminant pour ce choix : bien que le système adaptatif affiche le taux de détection (76.3%), un système qui détecterait parfaitement chaque attaque sans pour autant restaurer la capacité du modèle à identifier correctement le genre n'offrirait aucune protection réelle contre un utilisateur cherchant à tromper le classifieur. Ces deux arguments convergent vers une même conclusion : le système adaptatif constitue, parmi toutes les approches testées dans ce travail, la stratégie de défense la plus robuste, dans la mesure où elle est celle qui restaure le plus efficacement la performance réelle du système face à un utilisateur cherchant à tromper le modèle.

Deux limitations subsistent néanmoins et méritent d'être signalées avec transparence. Premièrement, *nn counterfactual* n'est que faiblement détectée (30.7%) : bien que le dictionnaire de défenses (figure 7.15) associe cette attaque à la défense *smoothing*, celle-ci n'est effectivement appliquée que dans 30.7% des cas ; dans la majorité des cas restants, le système ne reconnaît pas l'attaque et applique alors la prédiction directe sans défense (*none*), laissant ces exemples largement sans protection. Deuxièmement, *behavioral drift* n'est que partiellement détectée (38.6%) : de la même manière, la défense prévue (*adversarial training*) n'est appliquée que dans une minorité de cas, la défense *none* étant utilisée dans 61.4% des cas restants, ce qui s'avère insuffisant pour protéger les modèles contre cette attaque subtile. Ces deux limitations, loin de remettre en question la pertinence du système adaptatif, en délimitent plutôt les contours actuels et ouvrent des perspectives concrètes pour des travaux futurs. En particulier, le recours à des détecteurs appris par apprentissage automatique, plutôt que par règles statistiques manuelles, pourrait permettre de mieux capturer les signatures subtiles de ces deux attaques.

Perspectives d'amélioration : L'analyse de la confiance ouvre plusieurs pistes pour des travaux futurs. Premièrement, le MLP reste presque aussi confiant quand il se trompe que quand il a raison (écart de moins de 0.03, même sans attaque), ce qui montre que ses scores de confiance ne sont pas fiables. Une piste serait d'ajouter une régularisation par exemples perturbés pendant l'entraînement, ou d'utiliser des techniques simples de calibration comme le *temperature scaling*, pour corriger ce problème sur ce modèle en particulier.

Deuxièmement, même le système adaptatif retenu ne détecte pas bien deux types d'attaques (*nn counterfactual*, 30.7% seulement, et *behavioral drift*, 38.6%). Utiliser un vrai modèle d'apprentissage automatique pour détecter ces attaques, plutôt que des règles fixes écrites à la main, pourrait mieux repérer ces signatures subtiles, difficiles à distinguer d'un comportement naturel.

Troisièmement, aucun modèle ne retrouve entièrement sa performance d'origine après une attaque, même avec le système adaptatif. Cela suggère que des défenses intégrées directement pendant l'entraînement du modèle, plutôt qu'ajoutées après coup, seraient probablement plus efficaces.

Enfin, les modèles ne se comportent pas tous de la même façon : Random Forest et SVM restent robustes mais parfois trop peu confiants, alors que XGBoost et LightGBM ont la meilleure confiance après défense. Un futur système pourrait donc adapter non seulement la défense selon l'attaque détectée, mais aussi la façon de calibrer la confiance selon le modèle utilisé.

7.7 Estimation du risque de tromperie

Afin de rendre concret le risque résiduel identifié précédemment, une estimation de la probabilité qu'un attaquant parvienne à tromper le système a été réalisée, sous les hypothèses suivantes : le défenseur utilise un modèle donné protégé par le système adaptatif, et l'attaquant choisit une perturbation au hasard parmi les sept types d'attaques testées dans ce travail, sans connaissance préalable du détecteur.

TABLEAU 7.5 – Probabilité de tromperie par modèle, sous l'hypothèse d'une attaque choisie au hasard parmi les sept types testés

Modèle	$P(\text{erreur})$	$P(\text{erreur, conf.} > 75\%)$
LightGBM	0.175	0.097
RandomForest	0.188	0.019
SVM	0.195	0.023
XGBoost	0.201	0.058
GradientBoosting	0.204	0.097
MLP	0.237	0.188

Le Tableau 7.5 présente, pour chaque modèle, la probabilité d’erreur globale ainsi que la probabilité qu’une tentative de l’attaquant, prise sur l’ensemble des cas testés, produise à la fois une erreur de classification et une confiance de prédiction supérieure à 75%.

Sous ces hypothèses, la probabilité que le Random Forest se trompe est de 18.8%, dont seulement 1.9% des cas avec une confiance supérieure à 75%. À l’inverse, le MLP présente une probabilité d’erreur de 23.7%, dont 18.8% avec une confiance supérieure à 75% soit, rapporté au nombre d’erreurs commises, près de 80% des erreurs du MLP contre seulement 10% de celles du Random Forest. Le Random Forest constitue ainsi le choix le plus sûr en termes de risque opérationnel : les tentatives de tromperie qui réussissent contre lui sont plus rares, et surtout moins souvent commises avec un excès de confiance.

Il convient toutefois de préciser que cette mesure ne constitue pas, à elle seule, un critère de calibration : un modèle affichant une confiance uniformément élevée, indépendamment de la justesse de ses prédictions, pourrait présenter une probabilité d’erreur confiante faible simplement en raison d’un taux d’erreur global bas, sans pour autant être bien calibré au sens défini en Section 7.5. Le faible risque observé pour le Random Forest (1.9%) résulte ainsi de la combinaison de deux facteurs distincts : un taux d’erreur global modéré (18.8%) et une calibration effective de sa confiance, cette dernière ayant été établie séparément par l’écart entre confiance correcte et confiance sur erreur.

7.8 Conclusion

Ce chapitre a évalué diverses défenses réparties en deux lots, ainsi que plusieurs combinaisons de défenses empilées en séquence, testées sur six modèles et sept types des perturbations comportementales, afin de répondre à la question de savoir dans quelle mesure l’intégration de stratégies de défense permet d’améliorer la robustesse des modèles de classification du genre face aux perturbations comportementales en réalité virtuelle.

Le premier lot montre que l’Adversarial Training est la défense la plus efficace en moyenne (+0.066), mais son effet reste inégal : il est optimal pour RandomForest sur toutes les attaques mais dégrade le MLP. L’Ensemble apporte un gain modéré (+0.027) et constitue la meilleure option pour le MLP. Les méthodes de détection et rejet comme le LOF sont inutilisables en pratique en raison de taux d’abstention trop élevés : ces approches se limitent à détecter une anomalie générique et à refuser toute prédiction

sur les exemples jugés suspects, ce qui les rend inadaptées à un contexte où une réponse doit systématiquement être fournie. Le système adaptatif présenté plus loin dans ce mémoire repose sur une logique différente : il identifie le type précis de perturbation plutôt qu’une simple anomalie, et sélectionne une défense corrective adaptée plutôt que de rejeter l’exemple, évitant ainsi l’écueil de l’abstention. Le Median Filter, quant à lui, dégrade systématiquement les performances (-0.196).

Le second lot confirme que le Bootstrap ($+0.043$) et le Bagging ($+0.037$) sont les défenses les plus équilibrées, le Bootstrap étant le seul à améliorer tous les modèles simultanément sans dégradation notable. À l’inverse, le Log Transform (-0.196), l’OC-SVM, la distance de Mahalanobis et le Statistical Test s’avèrent soit nuisibles soit inutilisables en pratique. Le Subspace est très efficace pour RandomForest mais pénalise fortement le MLP et le SVM.

L’évaluation des défenses empilées confirme qu’il est possible d’améliorer davantage la robustesse en combinant plusieurs défenses en séquence. Le meilleur empilement **Wins+Zscore+PCA+Ens** atteint un gain moyen de $+0.057$, supérieur au Bootstrap seul ($+0.043$). La vue synthétique (Figure 7.14) montre que tous les modèles bénéficient de ces empilements : le SVM passe de 0.682 à 0.854 en moyenne sur toutes les attaques, et XGBoost de 0.714 à 0.860. Cependant, aucun modèle ne retrouve entièrement sa BAcc d’origine : l’écart entre la barre verte (Clean) et la barre bleue (Meilleur empilement) reste visible pour tous les modèles, en particulier pour GradientBoosting (0.933 clean contre 0.835 défendu) et le SVM (0.933 clean contre 0.854 défendu).

En réponse à **RQ3**, les résultats obtenus apportent une réponse claire et positive : l’intégration de stratégies de défense améliore effectivement et de manière mesurable la robustesse des modèles de classification du genre en réalité virtuelle. Des défenses comme l’Adversarial Training, le Bootstrap, le Bagging et les empilements séquentiels permettent de récupérer une part significative de la BAcc perdue sous attaque Gradient-Boosting remonte par exemple à 0.861 face à *gender_normalization_agnostic* (contre 0.477 sans défense), et le SVM atteint 0.854 en moyenne sur toutes les attaques avec le meilleur empilement. Ces résultats sont encore dépassés par le système adaptatif, qui atteint un gain moyen de $+0.081$, surpassant ainsi l’ensemble des défenses statiques et des empilements séquentiels évalués, confirmant qu’une stratégie de défense capable de s’adapter au type d’attaque détecté offre la meilleure robustesse effective parmi l’ensemble des approches testées ce gain de BAcc constituant le critère déterminant, indépendamment du taux de détection proprement dit.

Chapitre 8

Conclusion

8.1 Résumé

Ce travail a proposé une analyse approfondie de la robustesse des modèles de classification du genre en réalité virtuelle face à un large éventail de perturbations comportementales. Ces perturbations incluent à la fois des manipulations intentionnelles visant à tromper le classifieur, ainsi que d'autres formes de variabilité réaliste pouvant apparaître naturellement dans un environnement immersif.

À partir des données FAST et Alyx, plusieurs familles de modèles tabulaires, réseaux profonds de type MLP et architectures temporelles ont été évaluées sous plus d'une centaine de perturbations réalistes, incluant des manipulations morphologiques, posturales et dynamiques, ainsi que des perturbations affectant le regard, la synchronisation des mouvements des mains et la calibration du système de tracking.

Les résultats mettent en évidence une vulnérabilité structurelle des modèles standards, ainsi qu'une forte hétérogénéité de comportement selon la nature des perturbations et l'architecture considérée. Certaines perturbations globales, affectant la cohérence métrique ou les signatures comportementales latentes, entraînent des chutes importantes de performance et des taux élevés d'inversion de prédiction. À l'inverse, certaines perturbations localisées ou de faible amplitude ont un impact plus limité, car elles n'affectent qu'une partie des variables utilisées par le modèle.

Dans la mesure du possible, les perturbations implémentées dans ce travail sont appliquées de manière cohérente sur les groupes de variables représentant un même phénomène physique. Ainsi, lorsqu'une modification affecte une dimension du mouvement (posture, dynamique ou regard), les caractéristiques dérivées associées à cette dimen-

sion sont modifiées en même temps afin d'éviter des configurations irréalistes entre les variables.

Face à ces vulnérabilités, ce travail a développé et évalué un ensemble complet de mécanismes de défense adaptés aux données spatio-temporelles issues de la réalité virtuelle. Dix-neuf défenses fixes ont été évaluées sur six modèles et sept types de perturbations, couvrant quatre catégories : transformations des données, détection et rejet, méthodes basées sur le vote et entraînement robuste. Le Bootstrap s'est révélé la défense fixe la plus efficace avec un gain moyen de +0.043, suivi du Bagging (+0.037) et de l'Adversarial Training (+0.066 sur le premier lot). Cette analyse a cependant mis en évidence une limitation fondamentale : aucune défense unique n'est optimale pour l'ensemble des perturbations et des modèles, certaines stratégies efficaces dans un contexte donné s'avérant inutiles voire nuisibles dans un autre.

Au-delà des défenses appliquées séparément, des combinaisons de défenses empilées en séquence ont également été évaluées. Le meilleur empilement **Wins+Zscore+PCA+Ens** atteint un gain moyen de +0.057, supérieur au Bootstrap seul (+0.043). En moyenne sur toutes les attaques, le SVM passe de 0.682 à 0.854 et XGBoost de 0.714 à 0.860 avec le meilleur empilement. Ces résultats confirment que combiner plusieurs défenses en séquence améliore la robustesse, mais aucun modèle ne retrouve entièrement sa BAcc d'origine, ce qui souligne la limite des approches fixes indépendamment du nombre de défenses appliquées.

Cette observation a motivé le développement d'un système adaptatif de détection et de défense, capable d'identifier automatiquement le type de perturbation et de sélectionner dynamiquement la défense la plus appropriée selon le modèle concerné. Ce système atteint le meilleur gain moyen (+0.081) supérieur à toutes les défenses fixes évaluées et à tous les empilements testés et un taux de détection (76.3%).

Une estimation du risque résiduel de tromperie a également été réalisée, en supposant qu'un attaquant choisisse une attaque au hasard parmi les sept types connus du système. Cette analyse révèle que, pour le Random Forest protégé par le système adaptatif, la probabilité qu'un utilisateur trompe le classifieur est de 18.8%, dont seulement 1.9% des cas avec une confiance supérieure à 75% contre 23.7% et 18.8% respectivement pour le MLP. Ce résultat confirme que le choix du modèle influence directement le risque opérationnel réel, au-delà de la seule robustesse mesurée par la BAcc.

L'une des contributions majeures de ce mémoire réside dans la construction d'un pipeline complet d'évaluation appliqué aux signaux de réalité virtuelle, comprenant un

large catalogue de perturbations plausibles, une comparaison cohérente entre plusieurs familles de modèles, un système adaptatif et une analyse fine des indicateurs de performance incluant la balanced accuracy, la confiance de prédiction et le taux de détection.

8.2 Réponses aux questions de recherche

Les trois questions de recherche qui ont guidé ce travail ont reçu des réponses claires, appuyées par les résultats expérimentaux présentés dans les chapitres précédents.

RQ1 : Dans quelle mesure les données de mouvement issues de la réalité virtuelle permettent-elles de classifier le genre des utilisateurs à l'aide de modèles d'apprentissage automatique, en l'absence de perturbations comportementales ? Cette question a été répondue en montrant que les données de mouvement captées en réalité virtuelle constituent un signal fortement discriminant pour la classification du genre : les modèles évalués atteignent en conditions nominales des performances élevées, avec une BAcc pouvant atteindre 0.933 pour le Gradient Boosting et le SVM. Ce résultat confirme que les caractéristiques statistiques et dynamiques extraites des signaux corporels (posture, gestuelle, regard) encodent des patterns comportementaux suffisamment stables et distinctifs pour permettre une inférence fiable du genre en l'absence de manipulation.

RQ2 : Quel est l'impact de perturbations comportementales réalistes sur les performances et la fiabilité des modèles de reconnaissance du genre à partir de données de mouvement en réalité virtuelle ? Cette question a été répondue en confirmant que les modèles de classification du genre sont vulnérables aux perturbations comportementales réalistes, avec des chutes de BAcc pouvant atteindre 0.477 contre 0.933 en conditions normales. L'analyse a également mis en évidence une forte hétérogénéité de comportement selon le contexte : les perturbations globales et structurelles (neutralisation des marqueurs genrés, distorsion d'échelle) s'avèrent systématiquement plus destructrices que les perturbations locales ou physiologiquement plausibles, quel que soit le modèle considéré, et cette dégradation, observée sur l'ensemble des familles de modèles testées, confirme qu'aucune architecture n'est intrinsèquement à l'abri d'une manipulation comportementale délibérée.

RQ3 : Dans quelle mesure l'intégration de stratégies de défense permet-elle d'améliorer la robustesse des modèles de classification du genre face aux perturbations comportementales dans un contexte de réalité virtuelle ? Cette

question a été répondue en montrant que les défenses fixes, les empilements séquentiels et le système adaptatif améliorent tous de manière mesurable la robustesse, ce dernier offrant le meilleur gain de robustesse effectif (+0.081), surpassant toute défense statique ou empilement testé, ce gain de BAcc constituant le critère déterminant indépendamment du taux de détection proprement dit (76.3%).

8.3 Recommandations pratiques

Sur la base de l'ensemble des résultats obtenus, plusieurs recommandations peuvent être formulées pour un praticien souhaitant déployer un système de reconnaissance du genre en réalité virtuelle.

Pour choisir un modèle, le Random Forest est le meilleur choix et ce pour trois raisons. D'abord, il résiste le mieux aux attaques : seules trois attaques sur vingt le font vraiment chuter, alors que le Gradient Boosting s'effondre presque à chaque fois face aux attaques qui ciblent les marqueurs liés au genre (chute de 0.224 contre seulement 0.103 pour le Random Forest). Ensuite, sa confiance est plus fiable : l'écart entre sa confiance quand il a raison et quand il a tort (0.098) est bien plus grand que celui du MLP (0.018), ce qui veut dire qu'on peut mieux se fier à ses scores pour repérer les cas douteux. Enfin, cette robustesse ne coûte pas cher en performance, puisqu'il atteint quand même une BAcc de 0.886 sur données normales, proche des meilleurs modèles. Si on veut de meilleures performances de base (0.93 pour le Gradient Boosting et le SVM), on peut les choisir, mais il faut alors absolument les combiner avec une défense, car ce sont les deux modèles les plus fragiles et les plus sujets à la surconfiance sans protection.

Concernant le risque qui reste, même le meilleur système testé dans ce travail, le système adaptatif, ne permet pas d'éliminer complètement le risque qu'un utilisateur arrive à tromper le modèle. En supposant qu'un attaquant choisisse une attaque au hasard parmi les sept types connus du système, la probabilité que le Random Forest se trompe est de 18.8%, dont seulement 1.9% des cas avec une confiance supérieure à 75% (voir Section 7.7 pour le détail de cette analyse par modèle). Ce risque résiduel est notamment lié aux attaques *nn counterfactual* (30.7% de détection) et *behavioral drift* (38.6%), dont les signatures sont trop proches de la variabilité naturelle du comportement pour être bien détectées. Un déploiement en conditions réelles devrait donc tenir compte de ce risque qui reste présent, et prévoir des mécanismes de surveillance supplémentaires plutôt que de considérer le système comme totalement fiable.

En définitive, ce mémoire montre que la prédiction du genre à partir de signaux de mouvement en réalité virtuelle, bien qu'efficace en conditions nominales, demeure fortement dépendante de la stabilité des signatures comportementales exploitées par les modèles. Le système adaptatif développé dans ce travail constitue une réponse concrète à cette vulnérabilité, en combinant détection automatique des perturbations et sélection dynamique des défenses. L'analyse conjointe des performances et de la sensibilité aux perturbations met en évidence la nécessité d'intégrer la robustesse comme critère central dans l'évaluation des systèmes d'apprentissage automatique appliqués aux environnements immersifs.

8.4 Limitations et travaux futurs

Six limitations principales peuvent être identifiées dans ce travail. Premièrement, la perturbation *behavioral drift* reste partiellement non détectée par le système adaptatif, sa signature étant trop proche de la variabilité naturelle des données pour être distinguée avec fiabilité par des règles statistiques manuelles. Deuxièmement, les perturbations étudiées, bien que nombreuses et diversifiées, ne couvrent pas l'ensemble des manipulations possibles dans des environnements immersifs complexes. Troisièmement, l'étude reste centrée sur deux jeux de données publics, ce qui appelle à une validation complémentaire sur des environnements RV plus variés.

Quatrièmement, et plus fondamentalement, ce travail repose sur l'hypothèse que les perturbations simulées numériquement constituent une approximation valable de comportements qu'un utilisateur réel serait capable de reproduire volontairement dans un environnement immersif. Cette hypothèse, adoptée afin de permettre une évaluation systématique et reproductible d'un grand nombre de scénarios d'attaque, n'a cependant pas été validée expérimentalement dans le cadre de ce travail : les attaques testées constituent des modifications appliquées directement sur les données, sans garantie qu'elles soient reproductibles par un geste humain réel.

Cinquièmement, la performance du système adaptatif repose sur l'hypothèse implicite que l'attaquant choisit l'une des sept attaques connues du détecteur. Si l'utilisateur recourait à une perturbation non répertoriée, le système ne l'identifierait pas et appliquerait la défense par défaut (*none*), laissant l'exemple sans protection. Les autres défenses évaluées dans ce travail, bien qu'également testées uniquement sur ces sept attaques, sont conçues de manière *agnostique*, sans hypothèse a priori sur le type de perturbation

rencontré; il est donc probable qu’elles offriraient une meilleure robustesse générique qu’un système adaptatif si l’attaquant utilisait une perturbation inconnue du détecteur.

Sixièmement, ce travail repose sur une conceptualisation binaire du genre (homme/femme), héritée directement de l’étiquetage disponible dans les jeux de données FAST et Alyx. Cette simplification exclut les identités de genre non-binaires et ne permet pas de déterminer si les modèles évalués détectent effectivement le *genre* une identité personnelle et sociale ou plutôt le *sexe biologique*, qui se traduit par des différences morphologiques et dynamiques mesurables (taille, envergure, amplitude de mouvement). Dans la mesure où ces deux notions coïncident pour la grande majorité des participants des jeux de données utilisés, il n’est pas possible, avec les données disponibles, de trancher laquelle des deux dimensions est effectivement captée par les modèles.

Pour répondre à ces limitations, six axes de travaux futurs se dégagent. Premièrement, une validation expérimentale auprès d’utilisateurs réels serait nécessaire pour confirmer que les perturbations simulées dans ce travail correspondent à des comportements effectivement réalisables et adoptés par un attaquant humain il s’agit là d’une suite logique directe de ce travail, plutôt que d’une simple amélioration technique. Deuxièmement, le recours à des détecteurs appris par apprentissage automatique, plutôt que par règles statistiques manuelles, pourrait permettre de mieux capturer les signatures subtiles de *behavioral drift* et de *nn counterfactual*. Troisièmement, l’intégration d’une régularisation par exemples perturbés lors de l’entraînement pourrait améliorer la robustesse de modèles comme le SVM et le MLP, particulièrement sensibles au choix de la défense appliquée. Quatrièmement, des défenses proactives intégrées directement dans l’entraînement du modèle, plutôt qu’appliquées en post-traitement, constitueraient une approche plus robuste pour contrer les perturbations comportementales dans les environnements de réalité virtuelle.

Cinquièmement, l’étude de jeux de données incluant des participants aux identités de genre non-binaires, ainsi que des protocoles permettant de dissocier explicitement le sexe biologique du genre perçu ou revendiqué, permettrait de clarifier ce que les modèles détectent réellement à partir des données de mouvement. Sixièmement, des expériences avec le *temperature scaling* une méthode simple qui ajuste après coup les scores de confiance du modèle, sans avoir à le réentraîner pourraient être menées spécifiquement sur le MLP, afin de réduire son problème de calibration observé en Section 7.5.

8.5 Publication dérivée de ce travail

Une partie des travaux présentés dans ce mémoire a fait l’objet d’une publication scientifique :

E. Kraiem and A. Davoust, “Vulnerability of machine learning models for gender recognition in Virtual Reality,” in *Proceedings of the 39th Canadian Conference on Artificial Intelligence*, L. Bouzar-Benlabiod and C. Leung, Eds., vol. 318, Proceedings of Machine Learning Research, Vancouver, Canada, May 25–29, 2026, pp. 502–513. [Online]. Available : <https://proceedings.mlr.press/v318/kraiem26a.html>

Cette publication reprend seulement la partie sur les vulnérabilités présentée dans ce mémoire (chapitres sur les attaques et leur effet sur les modèles). Les défenses et le système adaptatif, eux, sont des ajouts nouveaux qui ne se trouvent pas dans cette publication.

Bibliographie

- [1] A. G. Moore, R. P. McMahan, H. Dong, and N. Ruozzi, “Personal identifiability and obfuscation of user tracking data from vr training sessions,” in *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2021, pp. 221–228.
- [2] M. Miller, E. Han, C. DeVeaux, E. Jones, R. Chen, and J. Bailenson, “A large-scale study of personal identifiability of virtual reality motion over time,” 03 2023.
- [3] K. Pfeuffer, M. J. Geiger, S. Prange, L. Mecke, D. Buschek, and F. Alt, “Behavioural biometrics in vr : Identifying people from body motion and relations in virtual reality,” New York, NY, USA, 2019. [Online]. Available : <https://doi.org/10.1145/3290605.3300340>
- [4] AlphauroraE, “Cross-domain gender identification using vr tracking data (github repository),” https://github.com/AlphauroraE/Cross-Domain_Gender_Identification_ML_Codebase, 2024, accessed 2025-11-12.
- [5] Q. J. Wang and R. P. McMahan, “Gender identification of vr users by machine learning tracking data,” pp. 827–828, 2024.
- [6] “Robust shared feature learning for script and handwritten/machine-printed identification,” pp. 6–13, 2017. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S0167865517303185>
- [7] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” 2014. [Online]. Available : <https://arxiv.org/abs/1312.6199>
- [8] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2015. [Online]. Available : <https://arxiv.org/abs/1412.6572>
- [9] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” 2019. [Online]. Available : <https://arxiv.org/abs/1706.06083>
- [10] S. Dastgerdy *et al.*, “Virtual reality and augmented reality security : A reconnaissance and vulnerability assessment approach.” [Online]. Available : <https://arxiv.org/html/2407.15984v1#bib>
- [11] M. Abdelaal, S. Sekar, F. Dürr, K. Rothermel, S. Becker, and D. Fritsch, “Mapsense : Grammar-supported inference of indoor objects from crowd-

- sourced 3d point clouds,” New York, NY, USA, 2020. [Online]. Available : <https://doi.org/10.1145/3379342>
- [12] M. Li, A. Shivakumar, N. K. Banerjee, and S. Banerjee, “Motion forecasting attacks on behavioral biometric authentication systems in virtual reality,” New York, NY, USA, 2025. [Online]. Available : <https://doi.org/10.1145/3756884.3766052>
 - [13] G. Borghesan *et al.*, “Motor variability in human-computer interaction,” *International Journal of Human-Computer Studies*, 2020. [Online]. Available : <https://doi.org/10.1016/j.ijhcs.2020.102432>
 - [14] P. Skurowski and M. Pawlyta, “Detection and classification of artifact distortions in optical motion capture sequences,” 03 2022.
 - [15] J. P. Vox, A. Weber, K. I. Wolf *et al.*, “An evaluation of motion trackers with virtual reality sensor technology in comparison to a marker-based motion capture system based on joint angles for ergonomic risk assessment.” in *National Library of medicine*, 2021. [Online]. Available : <https://pmc.ncbi.nlm.nih.gov/articles/PMC8124554/>
 - [16] J. Staender, “Stationäre versorgungsleistungen,” pp. 1–10, 01 2019.
 - [17] M. Awais, F. Shamshad, and S.-H. Bae, “Towards an adversarially robust normalization approach,” 2020. [Online]. Available : <https://arxiv.org/abs/2006.11007>
 - [18] T. Kim *et al.*, “Reversible instance normalization for accurate time-series forecasting against distribution shift,” in *Proceedings of ICLR 2022*, 2022. [Online]. Available : <https://openreview.net/pdf?id=cGDAkQo1C0p>
 - [19] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter, “Certified adversarial robustness via randomized smoothing,” 2019. [Online]. Available : <https://arxiv.org/abs/1902.02918>
 - [20] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest,” pp. 413–422, 2008.
 - [21] A. Shafahi, M. Najibi, A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, “Adversarial training for free!” 2019. [Online]. Available : <https://arxiv.org/abs/1904.12843>
 - [22] M. Slater and M. V. Sanchez-Vives, “Enhancing our lives with immersive virtual reality,” *Frontiers in Robotics and AI*, vol. Volume 3 - 2016, 2016. [Online]. Available : <https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2016.00074>
 - [23] J. Jerald, *The VR Book : Human-Centered Design for Virtual Reality*. ACM Books, 2015. [Online]. Available : <https://doi.org/10.1145/2792790>
 - [24] G. Makransky and G. B. Petersen, “The cognitive affective model of immersive learning (camil) : A theoretical research-based model of learning in immersive virtual reality,” *Educational Psychology Review*, 2021. [Online]. Available : <https://doi.org/10.1007/s10648-020-09586-2>

- [25] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997.
- [26] V. Nair, W. Guo, J. Mattern, R. Wang, J. F. O’Brien, L. Rosenberg, and D. Song, “Unique identification of 50,000+ virtual reality users from head & hand motion data,” 2023. [Online]. Available : <https://arxiv.org/abs/2302.08927>
- [27] A. Moore, T. do, N. Ruozzi, and R. McMahan, “Identifying virtual reality users across domain-specific tasks : A systematic investigation of tracked features for assembly,” pp. 396–404, 10 2023.
- [28] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, <http://www.deeplearningbook.org>.
- [29] XRtLab, “Fast dataset,” <https://github.com/xrtlab/FAST-Dataset/tree/main/Dataset>, 2021.
- [30] —, “Fast dataset : Full body tracking data from vr users,” <https://github.com/xrtlab/FAST-Dataset/tree/main/Dataset>, 2023.
- [31] C. Schell, “Who is alyx ? dataset,” <https://github.com/cschell/who-is-alyx/tree/main/players>, 2022.
- [32] Q. J. Wang and R. P. McMahan, “Gender identification of vr users by machine learning tracking data,” pp. 827–828, 2024.
- [33] L. Breiman, “Random forests,” *Machine Learning*, 2001. [Online]. Available : <https://doi.org/10.1023/A:1010933404324>
- [34] J. H. Friedman, “Greedy function approximation : A gradient boosting machine,” *Annals of Statistics*, 2001. [Online]. Available : <https://doi.org/10.1214/aos/1013203451>
- [35] T. Chen and C. Guestrin, “Xgboost : A scalable tree boosting system,” New York, NY, USA, 2016. [Online]. Available : <https://doi.org/10.1145/2939672.2939785>
- [36] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, “Lightgbm : A highly efficient gradient boosting decision tree,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available : https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf
- [37] A. G. Moore, R. P. McMahan, H. Dong, and N. Ruozzi, “Personal identifiability and obfuscation of user tracking data from vr training sessions,” in *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2021, pp. 221–228.
- [38] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” 2014. [Online]. Available : <https://arxiv.org/abs/1406.1078>

- [39] C.-C. M. Yeh, Y. Zhu, L. Ulanova *et al.*, “Time series joins, motifs, discords and shapelets : a unifying view that exploits the matrix profile,” *Data Mining and Knowledge Discovery*, 2017. [Online]. Available : <https://doi.org/10.1007/s10618-017-0519-9>
- [40] J. Han and B. Wang, “Global well-posedness for nls with a class of h^s -supercritical data,” 2019. [Online]. Available : <https://arxiv.org/abs/1901.08868>
- [41] H. Zhou *et al.*, “Informer : Beyond efficient transformer for long sequence time-series forecasting,” <https://github.com/zhouhaoyi/Informer2020>, 2021, aAAI’21 Best Paper.
- [42] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, “Timesnet : Temporal 2d-variation modeling for general time series analysis,” 2023. [Online]. Available : <https://arxiv.org/abs/2210.02186>
- [43] H. Wu *et al.*, “Timesnet : Temporal 2d-variation modeling for general time series analysis,” *OpenReview*, 2023. [Online]. Available : https://openreview.net/pdf?id=ju_Uqw384Oq
- [44] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” 2019. [Online]. Available : <https://arxiv.org/abs/1706.06083>
- [45] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, “Adversarial attacks on deep neural networks for time series classification,” in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019, pp. 1–8.
- [46] A. Kurniawan, Y. Ohsita, and M. Murata, “Experiments on adversarial examples for deep learning model using multimodal sensors,” *Sensors*, vol. 22, no. 22, 2022. [Online]. Available : <https://www.mdpi.com/1424-8220/22/22/8642>
- [47] M. Hardt, N. Megiddo, C. Papadimitriou, and M. Wootters, “Strategic classification.” New York, NY, USA : Association for Computing Machinery, 2016. [Online]. Available : <https://doi.org/10.1145/2840728.2840730>
- [48] S. Levanon and N. Rosenfeld, “Strategic classification made practical,” in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 6243–6253. [Online]. Available : <https://proceedings.mlr.press/v139/levanon21a.html>
- [49] F. Nesti, A. Biondi, and G. Buttazzo, “Detecting adversarial examples by input transformations, defense perturbations, and voting,” *IEEE Transactions on Neural Networks and Learning Systems*.
- [50] Q. Zhang *et al.*, “Countering adversarial examples : Combining input transformation and noisy training,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*.

- [51] S. Sharma and S. Chatterjee, “Winsorization for robust bayesian neural networks,” *Entropy*, vol. 23, no. 11, 2021. [Online]. Available : <https://www.mdpi.com/1099-4300/23/11/1546>
- [52] K. Boudt, V. Todorov, and W. Wang, “Robust distribution-based winsorization in composite indicators construction,” *Social Indicators Research*, vol. 149, no. 2, pp. pp. 375–397, 2020. [Online]. Available : <https://www.jstor.org/stable/48731594>
- [53] ScienceDirect, “Winsorization,” <https://www.sciencedirect.com/topics/mathematics/winsorization>, accessed : 2026-05-05.
- [54] S. Lee, “An illustrated guide to z-score transformation in data science,” <https://www.numberanalytics.com/blog/illustrated-zscore-transformation-guide>, 2025.
- [55] Minal Shah, Martin Gaievski, Owais Kazi, Fanit Kolchina, “Introducing the z-score normalization technique for hybrid search,” <https://opensearch.org/blog/introducing-the-z-score-normalization-technique-for-hybrid-search/>, 2025.
- [56] Portfolio123 Community, “Zscore - how to deal with outliers/clipping for ml download,” <https://community.portfolio123.com/t/zscore-how-to-deal-with-outliers-clipping-for-ml-download/65246>, 2023.
- [57] PythonProg, “Log transformation in machine learning,” <https://www.pythonprog.com/log-transformation-in-machine-learning/>, 2025.
- [58] GeeksforGeeks, “Log transformation,” <https://www.geeksforgeeks.org/data-science/log-transformation/>, 2025.
- [59] P. More, “Understanding feature normalization in machine learning : A beginner’s guide,” <https://medium.com/@morepravin1989/understanding-feature-normalization-in-machine-learning-a-beginners-guide-79a62ed03511>, 2025.
- [60] W. Jiang, “Applications of deep learning in stock market prediction : Recent progress,” *Expert Systems with Applications*, vol. 184, p. 115537, 2021. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S0957417421009441>
- [61] R. Bernhard, “Intégrité des réseaux de neurones : défenses contre les exemples adverses et leur transférabilité,” Theses, Université de Lyon, Nov. 2021. [Online]. Available : <https://theses.hal.science/tel-03880905>
- [62] H. ZHANG, T. Furon, L. AMSALEG, and Y. AVRITHIS, *Attaques et défenses de réseaux de neurones profonds : le cas de la classification d’images*, 07 2021, pp. 51–85.
- [63] W. Xu, D. Evans, and Y. Qi, “Feature squeezing : Detecting adversarial examples in deep neural networks,” in *Proceedings 2018 Network and Distributed System Security Symposium*, ser. NDSS 2018. Internet Society, 2018. [Online]. Available : <http://dx.doi.org/10.14722/ndss.2018.23198>
- [64] Weilin Xu, David Evans, Yanjun Qi, “Feature squeezing : Detecting adversarial examples in deep neural networks,” in *University of Virginia*.

- [65] Jie Yang, Man Wu, and Xiao-Zhang Liu, “Defense against adversarial attack using pca.” Springer, 2020. [Online]. Available : https://link.springer.com/content/pdf/10.1007/978-981-15-8086-4_59.pdf
- [66] A. C. Bovik, T. S. Huang, and D. C. Munson, “The effect of median filtering on edge estimation and detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 2, pp. 181–194, 1987.
- [67] G.-Y. Liao, T. Nodes, and N. Gallagher, “Output distributions of two-dimensional median filters,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 33, no. 5, pp. 1280–1295, 1985.
- [68] S. Tyan, “Median filtering : Deterministic properties,” in *Two-Dimensional Digital Signal Processing II : Transforms and Median Filters*. Springer, 2006, pp. 197–217.
- [69] ScienceDirect, “Median filtering,” <https://www.sciencedirect.com/topics/engineering/median-filtering>.
- [70] J.-X. Mi, X.-D. Wang, L.-F. Zhou, and K. Cheng, “Adversarial examples based on object detection tasks : A survey,” *Neurocomputing*, vol. 519, pp. 114–126, 2023. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S0925231222013273>
- [71] A. Aldahdooh, W. Hamidouche, S. A. Fezza, and O. Déforges, “Adversarial example detection for dnn models : a review and experimental comparison,” *Artificial Intelligence Review*, vol. 55, no. 6, p. 4403–4462, Jan. 2022. [Online]. Available : <http://dx.doi.org/10.1007/s10462-021-10125-w>
- [72] K. N. T. Nguyen, W. Zhang, K. Lu, Y.-H. Wu, X. Zheng, H. Li Tan, and L. Zhen, “A survey and evaluation of adversarial attacks in object detection,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 9, pp. 15 706–15 722, 2025.
- [73] N. Carlini and D. Wagner, “Adversarial examples are not easily detected : Bypassing ten detection methods,” 2017. [Online]. Available : <https://arxiv.org/abs/1705.07263>
- [74] F. Tramer, N. Carlini, W. Brendel, and A. Madry, “On adaptive attacks to adversarial example defenses,” 2020. [Online]. Available : <https://arxiv.org/abs/2002.08347>
- [75] scikit-learn developers, “Localoutlierfactor,” <https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.LocalOutlierFactor.html>, 2025.
- [76] S. Xu, H. Liu, L. Duan, and W. Wu, “An improved lof outlier detection algorithm,” in *2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, 2021, pp. 113–117.
- [77] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “Lof : identifying density-based local outliers,” *SIGMOD Rec.*, vol. 29, no. 2, p. 93–104, May 2000. [Online]. Available : <https://doi.org/10.1145/335191.335388>

- [78] A. Smiti, "A critical overview of outlier detection methods," *Computer Science Review*, vol. 38, p. 100306, 2020. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S1574013720304068>
- [79] M. S. Sadooghi and S. Esmailzadeh Khadem, "Improving one class support vector machine novelty detection scheme using nonlinear features," *Pattern Recognition*, vol. 83, pp. 14–33, 2018. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S0031320318301663>
- [80] J. Ma and S. Perkins, "Time-series novelty detection using one-class support vector machines," in *Proceedings of the International Joint Conference on Neural Networks, 2003.*, vol. 3, 2003, pp. 1741–1745 vol.3.
- [81] K. Yang, S. Kpotufe, and N. Feamster, "An efficient one-class svm for anomaly detection in the internet of things," 2021. [Online]. Available : <https://arxiv.org/abs/2104.11146>
- [82] Y. Hou, Z. Chen, M. Wu, C.-S. Foo, X. Li, and R. M. Shubair, "Mahalanobis distance based adversarial network for anomaly detection," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 3192–3196.
- [83] C. Fan, P. Liu, and W. Zhao, "Mahalanobis distance-guided conditional adversarial learning for universal domain adaptation," *Knowledge-Based Systems*, vol. 323, p. 113850, 2025. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S0950705125008962>
- [84] Y. Hou, Z. Chen, M. Wu, C.-S. Foo, X. Li, and R. M. Shubair, "Mahalanobis distance based adversarial network for anomaly detection." [Online]. Available : <https://personal.ntu.edu.sg/xlli/publication/Mahalanobis5.pdf>
- [85] O. S. Ohunakin, E. U. Henry, O. J. Matthew, V. U. Ezekiel, D. S. Adelekan, and A. T. Oyeniran, "Conditional monitoring and fault detection of wind turbines based on kolmogorov–smirnov non-parametric test," *Energy Reports*, vol. 11, pp. 2577–2591, 2024. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S2352484724000817>
- [86] M. Aslam, "Introducing kolmogorov–smirnov tests under uncertainty : An application to radioactive data," *ACS Omega*, vol. 5, no. 1, pp. 914–917, 2020, pMID : 31956845. [Online]. Available : <https://doi.org/10.1021/acsomega.9b03940>
- [87] X. Chen, W. Huang, Z. Peng, W. Guo, and F. Zhang, "Diversity supporting robustness : Enhancing adversarial robustness via differentiated ensemble predictions," *Computers & Security*, vol. 142, p. 103861, 2024. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S0167404824001627>
- [88] Y. Chai, Y. Liu, M. Ebrahimi, W. Li, B. Padmanabhan, Y. Jiang, and Y. Liu, "Enhancing adversarial robustness : A novel bayesian uncertainty-based ensemble learning method," in *2024 IEEE 9th International Conference on Data Science in Cyberspace (DSC)*, 2024, pp. 329–336.

- [89] L. Breiman, “Bagging predictors,” *Machine Learning*, vol. 24, no. 2, pp. 123–140, Aug. 1996. [Online]. Available : <https://doi.org/10.1023/A:1018054314350>
- [90] C. Lyu, K. Huang, and H.-N. Liang, “A unified gradient regularization family for adversarial examples,” in *2015 IEEE International Conference on Data Mining*, 2015, pp. 301–309.
- [91] M. Azad, T. H. Nehal, and M. Moshkov, “A novel ensemble learning method using majority based voting of multiple selective decision trees,” *Computing*, vol. 107, p. 42, 2025. [Online]. Available : <https://doi.org/10.1007/s00607-024-01394-8>
- [92] Q. Li, Z. Huang, and M. Pan, “Stability and robustness of minimal majority vote interpretable ensembles,” *Scientific Reports*, 2026. [Online]. Available : <https://doi.org/10.1038/s41598-026-45289-4>
- [93] T.-H. Lee, A. Ullah, and R. Wang, *Bootstrap Aggregating and Random Forest*. Cham : Springer International Publishing, 2020, pp. 389–429. [Online]. Available : https://doi.org/10.1007/978-3-030-31150-6_13
- [94] R. Salman, A. Alzaatreh, H. Sulieman, and S. Faisal, “A bootstrap framework for aggregating within and between feature selection methods,” *Entropy*, vol. 23, no. 2, 2021. [Online]. Available : <https://www.mdpi.com/1099-4300/23/2/200>
- [95] J. Jia, X. Cao, and N. Z. Gong, “Intrinsic certified robustness of bagging against data poisoning attacks,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 9, pp. 7961–7969, May 2021. [Online]. Available : <https://ojs.aaai.org/index.php/AAAI/article/view/16971>
- [96] M. Ahmed and M. A. Tahir, “Bagging as defence mechanism against adversarial attack,” in *2024 International Conference on Emerging Trends in Smart Technologies (ICETST)*, 2024, pp. 1–6.
- [97] N. Drenkow, N. Fendley, and P. Burlina, “Attack agnostic detection of adversarial examples via random subspace analysis,” in *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2022, pp. 2815–2825.
- [98] L. Liu, T. N. Hoang, L. M. Nguyen, and T.-W. Weng, “Promoting robustness of randomized smoothing : Two cost-effective approaches,” in *2023 IEEE International Conference on Data Mining (ICDM)*, 2023, pp. 1145–1150.
- [99] A. M. Rekavandi, F. Farokhi, O. Ohrimenko, and B. I. Rubinstein, “Certified adversarial robustness via randomized α -smoothing for regression models.” Red Hook, NY, USA : Curran Associates Inc., 2024.
- [100] J. Cohen, E. Rosenfeld, and Z. Kolter, “Certified adversarial robustness via randomized smoothing,” in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 1310–1320. [Online]. Available : <https://proceedings.mlr.press/v97/cohen19c.html>

- [101] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter, “Certified adversarial robustness via randomized smoothing,” 2019. [Online]. Available : <https://arxiv.org/abs/1902.02918>
- [102] Z. Qian, K. Huang, Q.-F. Wang, and X.-Y. Zhang, “A survey of robust adversarial training in pattern recognition : Fundamental, theory, and methodologies,” *Pattern Recognition*, vol. 131, p. 108889, 2022. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S0031320322003703>
- [103] —, “A survey of robust adversarial training in pattern recognition : Fundamental, theory, and methodologies,” 2022. [Online]. Available : <https://arxiv.org/abs/2203.14046>
- [104] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2015. [Online]. Available : <https://arxiv.org/abs/1412.6572>
- [105] U. Shaham, Y. Yamada, and S. Negahban, “Understanding adversarial training : Increasing local stability of supervised models through robust optimization,” *Neurocomputing*, vol. 307, 2018. [Online]. Available : <http://dx.doi.org/10.1016/j.neucom.2018.04.027>
- [106] W. Zhao, S. Alwidian, and Q. H. Mahmoud, “Adversarial training methods for deep learning : A systematic review,” *Algorithms*, vol. 15, no. 8, 2022. [Online]. Available : <https://www.mdpi.com/1999-4893/15/8/283>
- [107] Z. Wang, X. Li, H. Zhu, and C. Xie, “Revisiting adversarial training at scale,” in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 24 675–24 685.
- [108] M. Zhao, L. Zhang, J. Ye, H. Lu, B. Yin, and X. Wang, “Adversarial training : A survey,” *IEEE Transactions (Survey Article)*, 2024, manuscript received : 2024-10-19.
- [109] S. Chhabra, H. Venkateswara, and B. Li, “Label smoothing++ : Enhanced label regularization for training neural networks,” 2025. [Online]. Available : <https://arxiv.org/abs/2509.05307>
- [110] “Label smoothing regularization-based no hyperparameter domain generalization,” *Knowledge-Based Systems*, vol. 309, p. 112877, 2025. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S0950705124015119>
- [111] X. Jin, H. Zhu, S. Li, Z. Wang, Z. Liu, C. Yu, H. Qin, and S. Li, “A survey on mixup augmentations and beyond,” 09 2024.
- [112] X. Jin, H. Zhu, S. Li, Z. Wang, Z. Liu, J. Tian, C. Yu, H. Qin, and S. Z. Li, “A survey on mixup augmentations and beyond,” 2025. [Online]. Available : <https://arxiv.org/abs/2409.05202>
- [113] Z. Zhang, H. Wang, J. Geng, X. Deng, and W. Jiang, “A new data augmentation method based on mixup and dempster-shafer theory,” *IEEE Transactions on Multimedia*, vol. 26, pp. 4998–5013, 2024.

- [114] J. Smucny, G. Shi, T. A. Lesh, C. S. Carter, and I. Davidson, "Data augmentation with mixup : Enhancing performance of a functional neuroimaging-based prognostic deep learning classifier in recent onset psychosis," *NeuroImage : Clinical*, vol. 36, p. 103214, 2022. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S2213158222002790>
- [115] V. Nair, C. Rack, W. Guo, J. Mattern, R. Wang, J. F. O'Brien, L. Rosenberg, and D. Song, "Inferring private personal attributes of virtual reality users from head & hand motion data," 05 2023. [Online]. Available : <https://arxiv.org/abs/2302.08927>
- [116] Q. Wang, A. Moore, N. Chawla, and R. McMahan, "Cross-domain gender identification using vr tracking data," pp. 180–189, 10 2024.
- [117] V. Nair, C. Rack, W. Guo, R. Wang, S. Li, B. Huang, A. Cull, J. F. O'Brien, M. Latoschik, L. Rosenberg, and D. Song, "Inferring private personal attributes of virtual reality users from head and hand motion data," *arXiv preprint arXiv :2305.19198*, 2023. [Online]. Available : <https://arxiv.org/pdf/2305.19198.pdf>
- [118] C. Rack, T. Fernando, M. Yalcin, A. Hotho, and M. E. Latoschik, "Who is alyx ? a new behavioral biometric dataset for user identification in xr," *Frontiers in Virtual Reality*, vol. Volume 4 - 2023, 2023. [Online]. Available : <https://www.frontiersin.org/journals/virtual-reality/articles/10.3389/frvir.2023.1272234>
- [119] S. M. Asish and A. Sarker, "Vr eye tracking data for gender identification : A look at same-domain and cross-domain scenarios," New York, NY, USA, 2025. [Online]. Available : <https://doi.org/10.1145/3756884.3770550>
- [120] G. Bui, B. Morago, T. Le, K. Karsch, Z. Lu, and Y. Duan, "Integrating videos with lidar scans for virtual reality," in *2016 IEEE Virtual Reality (VR)*, 2016, pp. 161–162.
- [121] S. Eberz, K. B. Rasmussen, and V. Lenders, "Broken hearted : How to attack ecg biometrics," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2017. [Online]. Available : chrome-extension://efaidnbmninnibpcajpcglclefindmkaj/https://www.ndss-symposium.org/wp-content/uploads/2017/09/ndss2017_03A-1_Eberz_paper.pdf
- [122] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma, "Adversarial classification," New York, NY, USA, 2004. [Online]. Available : <https://doi.org/10.1145/1014052.1014066>
- [123] D. Lowd and C. Meek, "Adversarial learning," New York, NY, USA, 2005. [Online]. Available : <https://doi.org/10.1145/1081870.1081950>
- [124] A. Kantchelian, J. D. Tygar, and A. D. Joseph, "Evasion and hardening of tree ensemble classifiers," 2016. [Online]. Available : <https://arxiv.org/abs/1509.07892>
- [125] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," New York, NY, USA, 2017. [Online]. Available : <https://doi.org/10.1145/3052973.3053009>

- [126] F. Karim, S. Majumdar, and H. Darabi, "Adversarial attacks on time series," 2019. [Online]. Available : <https://arxiv.org/abs/1902.10755>
- [127] Y. Diao, T. Shao, Y.-L. Yang, K. Zhou, and H. Wang, "Basar :black-box attack on skeletal action recognition," 2021. [Online]. Available : <https://arxiv.org/abs/2103.05266>
- [128] Y. Wang, H. Yang, and W. Xu, "Wearguard : Defeating adversarial attacks in wearable ai-based activity recognition systems," New York, NY, USA, 2025. [Online]. Available : <https://doi.org/10.1145/3737903.3768565>
- [129] V. Nair, W. Guo, J. F. O'Brien, L. Rosenberg, and D. Song, "Deep motion masking for secure, usable, and scalable real-time anonymization of ecological virtual reality motion data," in *2024 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, 2024, pp. 493–500.
- [130] S. M. Ross, *A First Course in Probability*, 9th ed. Prentice Hall, 2010.
- [131] D. C. Niehorster, L. Li, and M. Lappe, "Eye tracking in virtual reality : a broad review of applications and challenges." *Springer Nature Link*, 2023. [Online]. Available : <https://link.springer.com/article/10.1007/s10055-022-00738-z#citeas>
- [132] H. Wang, X. Li, J. Li, H. Zhu, and J. Luo, "Vr-fi : Positioning and recognizing hand gestures via vr-embedded wi-fi sensing," pp. 8287–8300, 2025.
- [133] J. F. Trevor Hastie, Robert Tibshirani, *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*, 2nd ed. Springer, 2009. [Online]. Available : <https://link.springer.com/book/10.1007/978-0-387-84858-7#accessibility-information>
- [134] *Frontmatter*. John Wiley & Sons, Ltd, 2009, pp. i–xiv. [Online]. Available : <https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470549148.fmatter>
- [135] H.-H. Lee and H.-C. Shin, "Entropy-based human activity measure using fmcw radar," *Entropy*, vol. 27, no. 7, 2025. [Online]. Available : <https://www.mdpi.com/1099-4300/27/7/720>
- [136] J. S. Richman and J. R. Moorman, "Physiological time-series analysis using approximate entropy and sample entropy," *American Journal of Physiology-Heart and Circulatory Physiology*, vol. 278, no. 6, pp. H2039–H2049, 2000, pMID : 10843903.
- [137] S. Benhamou, "How to reliably estimate the tortuosity of an animal's path : : straightness, sinuosity, or fractal dimension?" *Journal of Theoretical Biology*, vol. 229, no. 2, pp. 209–220, 2004. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S0022519304001353>
- [138] S. Balasubramanian, A. Melendez-Calderon, and E. Burdet, "On the analysis of movement smoothness," *Journal of NeuroEngineering and Rehabilitation*, vol. 12, no. 1, p. 112, 2015. [Online]. Available : <https://jneuroengrehab.biomedcentral.com/articles/10.1186/s12984-015-0090-9>

- [139] X. Ben, P. Zhang, Z. Lai, R. Yan, X. Zhai, and W. Meng, “A general tensor representation framework for cross-view gait recognition,” *Pattern Recognition*, vol. 90, pp. 87–98, 2019. [Online]. Available : <https://www.sciencedirect.com/science/article/pii/S0031320319300251>
- [140] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, “Scikit-learn : Machine learning in python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. [Online]. Available : <http://jmlr.org/papers/v12/pedregosa11a.html>
- [141] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003. [Online]. Available : <http://www.jmlr.org/papers/v3/guyon03a.html>
- [142] S. Varma and R. Simon, “Bias in error estimation when using cross-validation for model selection,” *BMC Bioinformatics*, vol. 7, no. 1, p. 91, 2006. [Online]. Available : <https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-7-91>
- [143] G. C. Cawley and N. L. C. Talbot, “On over-fitting in model selection and subsequent selection bias in performance evaluation,” *Journal of Machine Learning Research*, vol. 11, pp. 2079–2107, 2010. [Online]. Available : <http://jmlr.org/papers/v11/cawley10a.html>
- [144] J. Wainer and G. Cawley, “Nested cross-validation when selecting classifiers is overzealous for most practical applications,” 2018. [Online]. Available : <https://arxiv.org/abs/1809.09446>
- [145] S. Parvande *et al.*, “Consensus features nested cross-validation,” *PMC*, 2020. [Online]. Available : <https://pmc.ncbi.nlm.nih.gov/articles/PMC7776094/>
- [146] E. Blancas, “A gentle introduction to nested cross-validation,” 2022. [Online]. Available : <https://ploomber.io/blog/nested-cv/>
- [147] scikit-learn developers, “Nested versus non-nested cross-validation — scikit-learn example,” 2025. [Online]. Available : https://scikit-learn.org/stable/auto_examples/model_selection/plot_nested_cross_validation_iris.html
- [148] O. J. Woodman, “An introduction to inertial navigation,” University of Cambridge, Computer Laboratory, Tech. Rep. UCAM-CL-TR-696, 2007.
- [149] E. N. Gilbert, “Capacity of a burst-noise channel,” *The Bell System Technical Journal*, vol. 39, no. 5, pp. 1253–1265, 1960.
- [150] C. Zhang *et al.*, “Anefficient adversarial attack for tree ensembles,” in *34th Conference on Neural Information Processing Systems*, 2020. [Online]. Available : <https://proceedings.neurips.cc/paper/2020/file/ba3e9b6a519cfddc560b5d53210df1bd-Paper.pdf>

Annexe A

Famille 4 - Attaques guidées par le modèle (semi-informées, coût computationnel élevé)

La dernière famille de perturbations, synthétisée dans le Tableau A.1, correspond à des attaques dites **semi-informées**, dans lesquelles l’adversaire ne dispose pas d’un accès direct aux paramètres internes du modèle, mais exploite indirectement son comportement à travers les sorties observées (par exemple via une API de prédiction).

Ce cadre se situe à l’interface entre les attaques purement agnostiques et les attaques fondées sur l’accès aux gradients.

Dans ce scénario, l’attaquant cherche à identifier les dimensions les plus sensibles de l’espace de représentation en analysant la variation des sorties du modèle face à des perturbations contrôlées. Les perturbations générées sont ainsi plus ciblées et généralement plus efficaces pour induire des erreurs de classification ou une baisse significative de la confiance, au prix d’un **coût computationnel plus élevé**. Ces attaques reposent typiquement sur des stratégies de recherche itérative, telles que des méthodes de type Monte Carlo, des recherches locales guidées par la sortie, des approximations de gradient ou la génération de contre-exemples.

Dans le cadre de ce travail, ces attaques sont considérées comme un *niveau de perturbation avancé*, représentatif de menaces réalistes dans des systèmes exposés (services distants, API publiques, systèmes déployés). Leur nombre est volontairement limité afin de maintenir un protocole expérimental équilibré et reproductible, tout en permettant d’évaluer la vulnérabilité des modèles face à des adversaires plus stratégiques et informés.

TABLEAU A.1 – Famille 6 - Attaques semi-informées (plus coûteuses).

Attaque	Scénario réaliste	Origine	Implémentation (code)
<code>nn_counterfactual_agnostic</code>	Contre-exemple proche	Attaquant	Pour chaque échantillon, choisit une classe cible au hasard (0/1), prend dans le pool l'exemple le plus similaire (distance euclidienne), puis remplace par un mélange 50/50 : $x' = 0.5x + 0.5x_{pool}$.
<code>montecarlo_multimodel</code>	Sondage probabiliste	Attaquant	Pour chaque échantillon, teste $n_{iter} = 60$ versions bruitées (gaussien, $\sigma_{init} = 0.15$ avec décroissance 0.992) et garde celle qui minimise la confiance moyenne des modèles.
<code>simulated_gradient_multimodel</code>	Gradient simulé	Attaquant	Pour chaque échantillon, essaie $n_{iter} = 8$ perturbations sur $n_{features} = 10$ features (ajout $\pm\epsilon$, $\epsilon = 0.3$) et garde celle qui réduit le plus la confiance moyenne des modèles.
<code>targeted_importance_agnostic</code>	Exploitation des features clés	Attaquant	Sur <code>important_feats</code> , ajoute ou soustrait une perturbation proportionnelle à la valeur : $x' = x + s x \cdot 0.25$ avec $s \in \{-1, +1\}$ (tiré par échantillon).
<code>boundary_interpolation_agnostic</code>	Franchissement progressif	Attaquant	Calcule des centroïdes à partir du pool (X_{pool}, y_{pool}) , puis teste $n_{steps} = 25$ interpolations de x vers un centroïde et garde celle qui minimise la confiance moyenne des modèles.

Annexe B

Analyse détaillée par défense

B.1 Adversarial training

L’Adversarial Training enrichit l’ensemble d’entraînement avec des exemples perturbés générés pendant la phase d’apprentissage, afin que le modèle apprenne à reconnaître et à résister aux perturbations.

Pour RandomForest, la défense produit les meilleurs résultats : la BAcc remonte à 0.932 face à *atk_behavioral_drift* (contre 0.864 sous attaque) et à 0.886 face à *gender_normalization_agnostic* et *total_neutralization_agnostic*. C’est le seul modèle pour lequel l’Adversarial Training s’avère optimal face à toutes les attaques. Pour XGBoost, la récupération est notable : 0.864 face à *atk_behavioral_drift* (depuis 0.773), 0.841 face à la plupart des autres attaques. GradientBoosting récupère à 0.818 face à la majorité des attaques, mais reste limité à 0.773 face à *unit_mismatch_pos_agnostic*. LightGBM atteint 0.818 à 0.841 sur la quasi-totalité des attaques, et le SVM récupère à 0.818 face à la plupart des perturbations. Le MLP est le seul modèle pour lequel l’Adversarial Training est contre-productif : la BAcc tombe à 0.659 face à *atk_behavioral_drift* (contre 0.864 sans défense) et reste à 0.682 face à *unit_mismatch_pos_agnostic*. Ce comportement s’explique par l’overconfiance du MLP : les exemples perturbés injectés à l’entraînement orientent le modèle vers des régions de décision qui dégradent ses performances sur les attaques non vues, sans réduire sa confiance en conséquence.

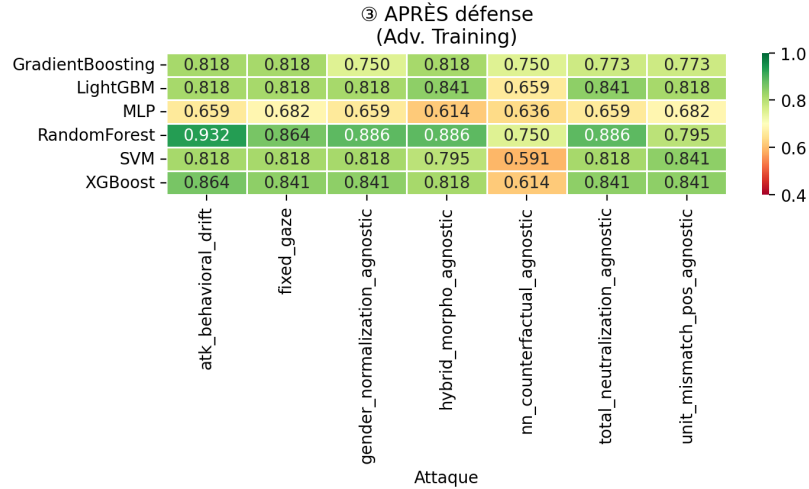


FIGURE B.1 – Impact de l’Adversarial Training.

B.2 LOF + Rejet

Le LOF + Rejet (Local Outlier Factor) calcule un score de densité locale pour chaque exemple. Ceux dont le score dépasse un seuil sont considérés comme suspects et rejetés : le modèle s’abstient de prédire plutôt que de risquer une prédiction erronée.

La Figure B.2 révèle le problème fondamental du LOF + Rejet : la quasi-totalité des cellules affiche 0.000. Cela ne signifie pas que le modèle prédit mal, mais que le mécanisme de rejet écarte la quasi-totalité des exemples, rendant le système incapable de produire des prédictions en pratique.

La seule exception concerne *nn_counterfactual_agnostic*, pour lequel des BAcc non nulles sont observées : GradientBoosting (0.725), SVM (0.707), XGBoost (0.725), MLP (0.659), RandomForest (0.659) et LightGBM (0.634). Le LOF parvient à identifier cette attaque spécifique car ses exemples présentent des caractéristiques de densité locale suffisamment distinctes pour être détectés comme hors distribution. Face à toutes les autres attaques, le LOF ne distingue pas les exemples perturbés des exemples légitimes et finit par tout rejeter.

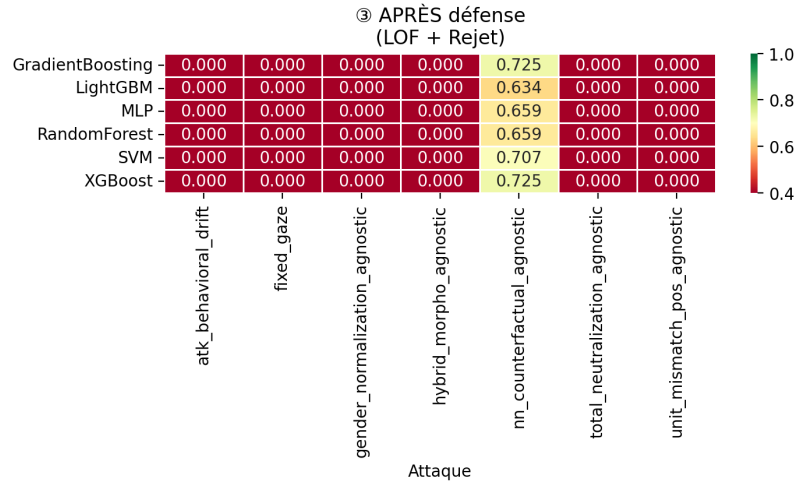


FIGURE B.2 – Impact du LOF + Rejet.

Cette limitation illustre le problème structurel des approches par détection et rejet : en cherchant à éliminer les exemples suspects, elles éliminent également les exemples légitimes, ce qui est inacceptable en contexte opérationnel.

B.3 PCA reconstruction

La PCA Reconstruction projette chaque exemple dans un espace de dimension réduite via une Analyse en Composantes Principales, puis le reconstruit. Les perturbations comportementales souvent hors distribution sont supposées s’atténuer lors de cette projection, les composantes principales ne capturant que la variance principale des données légitimes.

La reconstruction par PCA (Figure B.3) présente le comportement le plus contrasté parmi toutes les défenses. Face à *atk_behavioral_drift*, elle est efficace pour tous les modèles (BAcc de 0.818 à 0.841), et face à *total_neutralization_agnostic*, RandomForest atteint 0.886 et XGBoost 0.864, ce qui représente une récupération significative par rapport à la situation sans défense.

En revanche, elle dégrade fortement les performances face à *fixed_gaze* pour tous les modèles (BAcc de 0.523 à 0.545, contre 0.636 à 0.886 sans défense) et face à *hybrid_morpho_agnostic* (GradientBoosting chute à 0.523 depuis 0.886, LightGBM à 0.523 depuis 0.795, MLP à 0.523 depuis 0.568). Ce comportement s’explique par le fait que ces attaques modifient des caractéristiques capturées par les premières composantes principales : la reconstruction PCA supprime alors de l’information utile plutôt que de filtrer les perturbations.

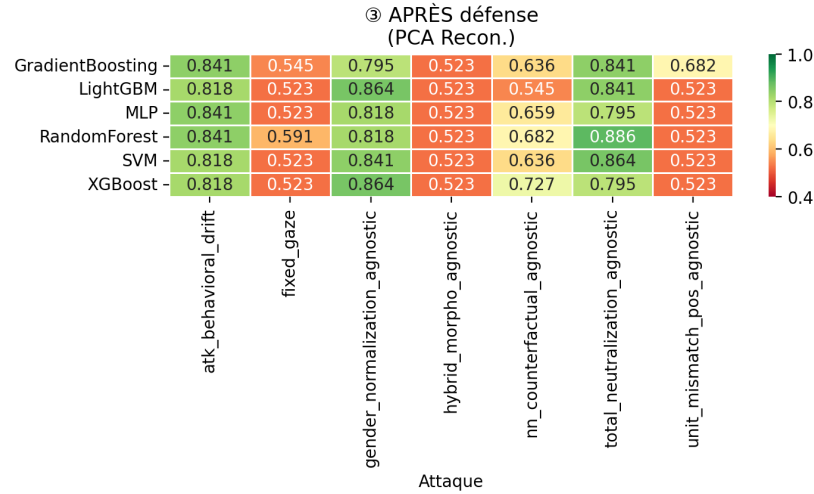


FIGURE B.3 – Impact de la PCA Reconstruction.

B.4 Squeezing défense

Le Squeezing Défense réduit la précision des valeurs de features (par quantification) afin d'atténuer les perturbations comportementales de faible amplitude, qui reposent sur des modifications précises dans l'espace continu des features.

Le Squeezing Défense est globalement neutre (figure B.4). Cette neutralité s'explique par le fait que les attaques testées reposent sur des modifications comportementales ou structurelles que la quantification ne parvient pas à éliminer, et non sur des perturbations de faible amplitude dans l'espace continu des features.

Quelques dégradations marginales sont visibles, notamment pour LightGBM face à *fixed_gaze* (0.773 contre 0.795) et pour MLP face à *hybrid_morpho_agnostic* (0.545 contre 0.568). Dans l'ensemble, cette défense n'apporte ni gain ni perte significatifs dans le contexte de ces attaques.

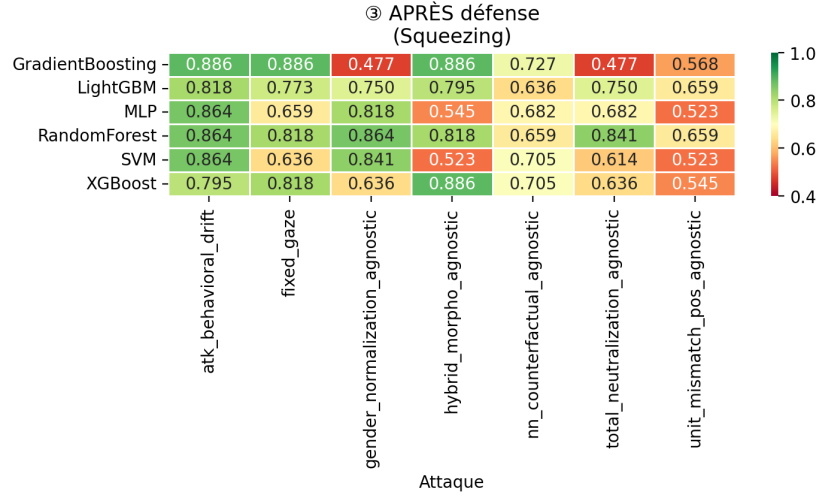


FIGURE B.4 – Impact du Squeezing Défense.

B.5 Smoothing

Le Smoothing applique un lissage des features d'entrée afin d'atténuer les perturbations haute fréquence introduites par les attaques, en s'appuyant sur l'hypothèse que les données légitimes présentent une variation plus régulière que les exemples perturbés.

GradientBoosting est le principal bénéficiaire du Smoothing (figure B.5) : sa BAcc remonte à 0.909 face à *atk_behavioral_drift* (contre 0.886 sans défense) et à 0.750 face à *total_neutralization_agnostic* (contre 0.477), ce qui constitue l'amélioration la plus notable de cette défense. RandomForest bénéficie également d'une amélioration modérée face à *hybrid_morpho_agnostic* (0.886 contre 0.841).

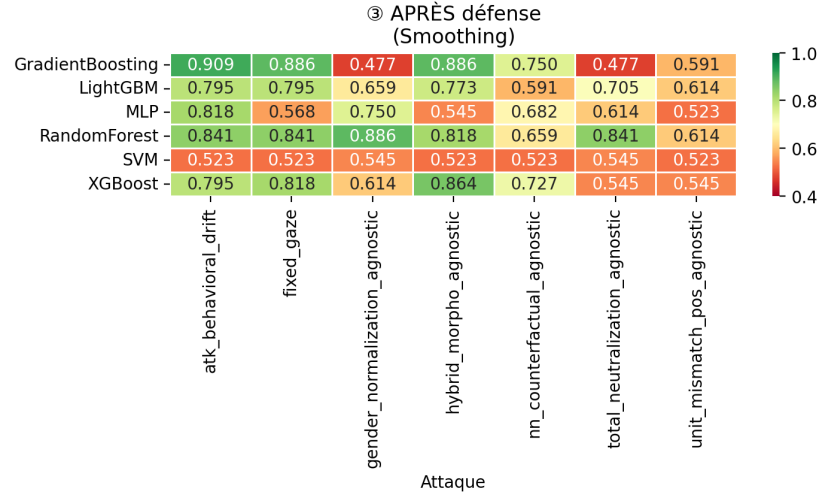


FIGURE B.5 – Impact du Smoothing.

En revanche, le SVM subit une dégradation sévère : sa BAcc tombe à 0.523 face à *atk_behavioral_drift* (contre 0.864 sans défense) et reste à 0.523 face à plusieurs autres attaques. Le Smoothing perturbe les features du SVM plutôt qu'il ne les corrige, ce qui en fait la pire défense pour ce modèle après le Median Filter. MLP et LightGBM présentent des résultats mitigés, avec quelques dégradations face à *hybrid_morpho_agnostic* (MLP à 0.545 contre 0.568, LightGBM à 0.591 contre 0.614).

B.6 Ensemble

La défense Ensemble combine les prédictions de plusieurs modèles afin de stabiliser les décisions face aux perturbations, en s'appuyant sur le principe que les attaques qui trompent un modèle ont moins de chances de tromper simultanément plusieurs modèles indépendants.

L'Ensemble produit un effet de convergence notable (figure B.6) : la quasi-totalité des cellules affiche 0.841 ou 0.773, quelle que soit l'attaque ou le modèle. Cet effet est à double tranchant. D'un côté, il permet de récupérer des BAcc acceptables face aux attaques les plus sévères. GradientBoosting remonte à 0.841 face à *gender_normalization_agnostic* et *total_neutralization_agnostic* (contre 0.477 dans les deux cas), et le SVM récupère à 0.841 face à *hybrid_morpho_agnostic* (contre 0.523). C'est également la meilleure défense pour le MLP, qui atteint 0.841 sur plusieurs attaques difficiles.

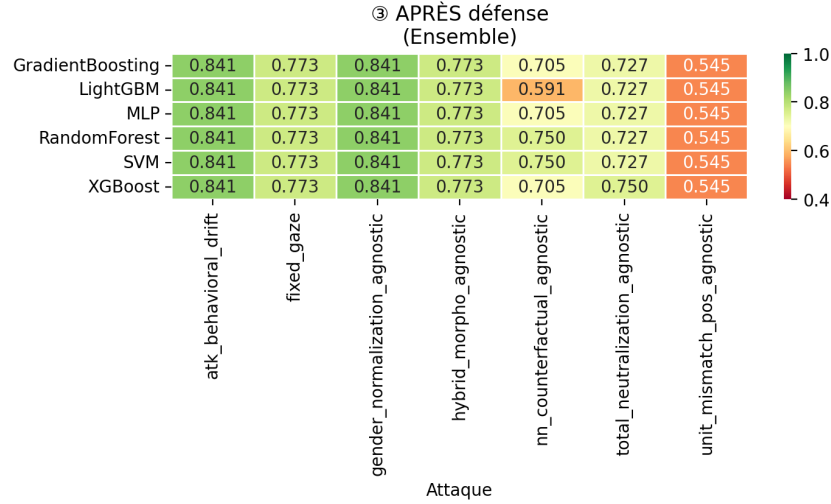


FIGURE B.6 – Impact de la défense Ensemble.

En revanche, l'Ensemble de modèles pénalise les modèles déjà robustes. GradientBoosting passe de 0.886 à 0.773 face à *fixed_gaze*, et RandomForest, qui résistait bien sans défense (0.841), est ramené à 0.773. L'Ensemble est donc une défense de compromis : elle réduit les pertes sur les cas difficiles mais dégrade les cas favorables, ce qui explique son gain moyen modéré (+0.027).

B.7 Median filter

Le Median Filter remplace chaque valeur de feature par la médiane de ses voisins, dans le but d'éliminer les valeurs aberrantes et les perturbations ponctuelles introduites par les perturbations comportementales.

Le Median Filter est la défense la plus nuisible sans exception (figure B.7). Après son application, toutes les BAcc sont comprises entre 0.432 et 0.591, pour tous les modèles et toutes les attaques. Cette dégradation est souvent pire que la situation sans défense : GradientBoosting, qui maintenait 0.886 face à plusieurs attaques, tombe à 0.477–0.523 ; RandomForest chute à 0.477–0.523 sur toutes les combinaisons ; XGBoost atteint 0.432 face à *nn_counterfactual_agnostic*, sa valeur la plus basse de toute l'évaluation. Ce comportement s'explique par la nature même du filtre : en remplaçant les valeurs par des médianes locales, il ne supprime pas les perturbations comportementales mais détruit l'information discriminante contenue dans les features originales, rendant les données filtrées moins informatives que les données perturbées. Le Median Filter est donc contre-indiqué dans ce contexte, quelle que soit l'attaque ou le modèle ciblé.

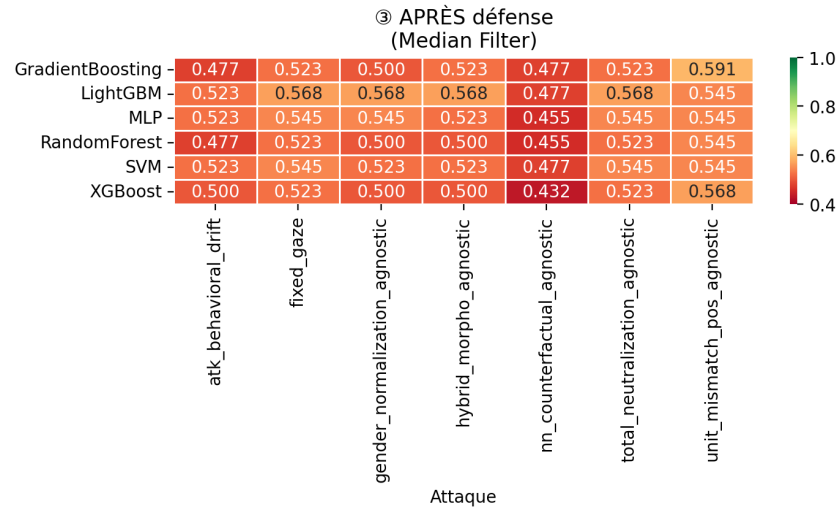


FIGURE B.7 – Impact du Median Filter.

B.8 Winsorization

La Winsorization ramène les valeurs trop basses ou trop hautes à un seuil fixe, ce qui réduit l'influence des valeurs extrêmes introduites par les attaques.

La Winsorization est globalement bénéfique (figure B.8), avec des améliorations ciblées sur certaines combinaisons. Le SVM est le principal bénéficiaire : sa BAcc remonte à 0.818 face à *fixed_gaze* (contre 0.636, gain de +0.182) et à 0.886 face à *atk_behavioral_drift* (contre 0.864, légère amélioration). GradientBoosting améliore également sa BAcc face à *fixed_gaze* (0.909 contre 0.886). MLP maintient des valeurs stables autour de 0.841 sur la plupart des attaques, avec une légère dégradation face à *unit_mismatch_pos_agnostic* (0.545 contre 0.523, en réalité une amélioration).

La Winsorization reste globalement stable ou légèrement positive pour RandomForest, LightGBM et XGBoost, sans dégradation significative observée.

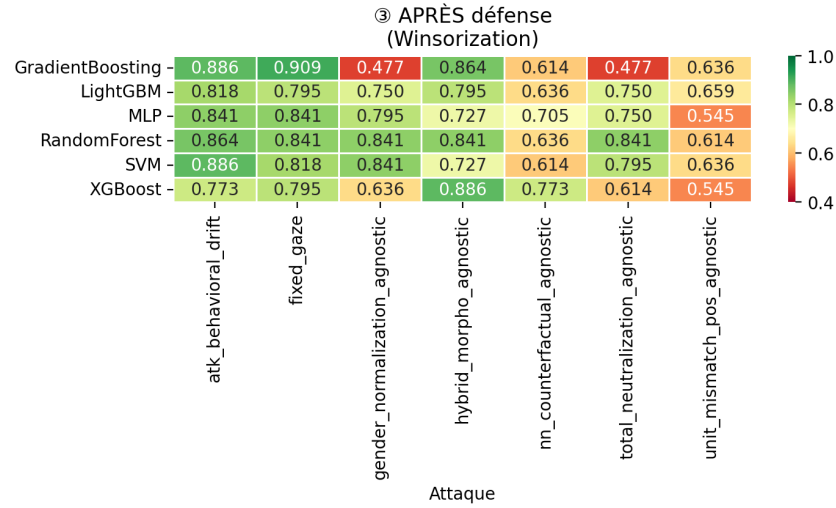


FIGURE B.8 – Impact de la Winsorization.

B.9 Z-score

Le Z-score normalise chaque feature en soustrayant sa moyenne et en divisant par son écart-type. Les perturbations comportementales qui s'écartent fortement de la distribution normale sont ainsi atténuées.

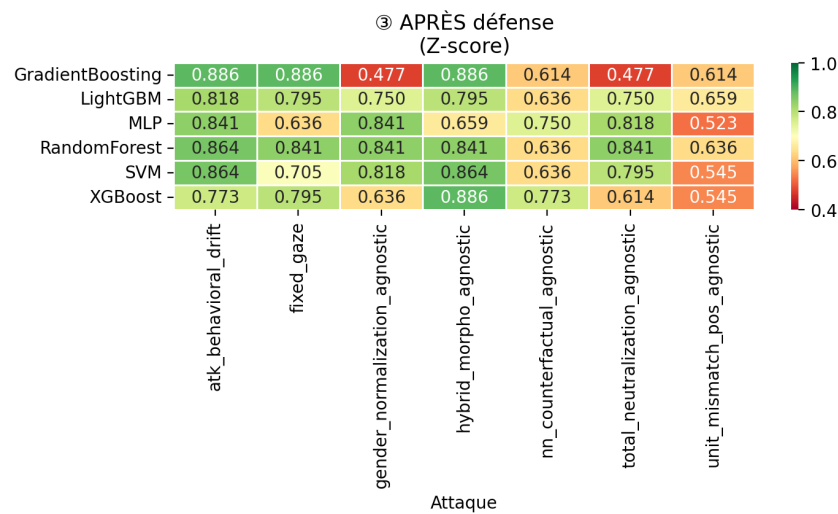


FIGURE B.9 – Impact du Z-score.

Le Z-score produit un effet très proche de la Winsorization (Figure B.9). Le SVM est le principal bénéficiaire : sa BAcc remonte à 0.705 face à *fixed_gaze* (contre

0.636, gain de +0.069), à 0.864 face à *hybrid_morpho_agnostic* (contre 0.523, gain de +0.341, la meilleure récupération observée pour cette défense) et à 0.795 face à *total_neutralization_agnostic* (contre 0.591).

GradientBoosting, LightGBM, MLP, RandomForest et XGBoost restent quasi stables pour la quasi-totalité des attaques, sans gain ni perte significatifs. Le Z-score est donc une défense ciblée, bénéfique principalement pour le SVM, et sans effet notable pour les autres modèles.

B.10 Log transform

Le Log Transform applique une transformation logarithmique à toutes les features, dans le but de réduire l'influence des grandes valeurs et de compresser l'espace des features.

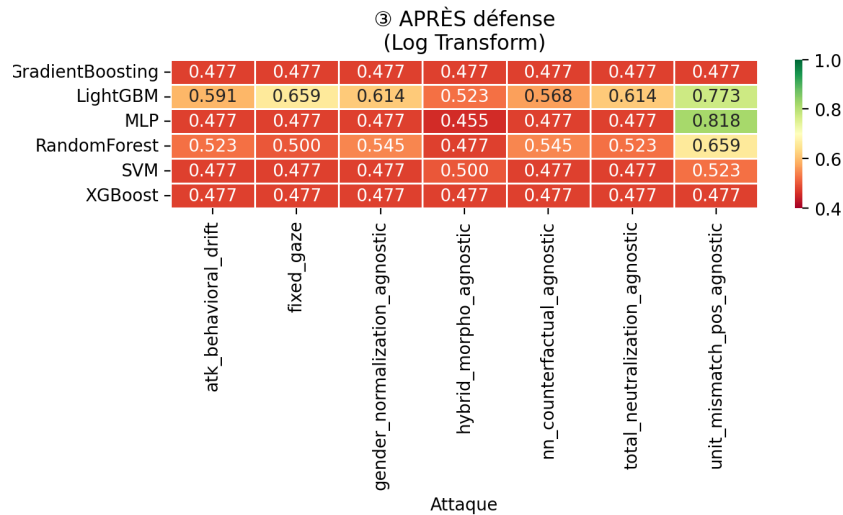


FIGURE B.10 – Impact du Log Transform.

Le Log Transform est la défense la plus nuisible du lot (figure B.10). Pour GradientBoosting et XGBoost, toutes les cellules tombent à 0.477 indépendamment de l'attaque : que le modèle résistait bien (0.886) ou mal (0.477) sans défense, le résultat est identique, signe que la transformation détruit complètement l'information discriminante pour ces modèles. Le SVM est presque aussi affecté, avec des valeurs entre 0.477 et 0.523 sur toutes les attaques. RandomForest tombe entre 0.500 et 0.659, et MLP entre 0.455 et 0.477 sur la majorité des attaques.

Le seul cas où le Log Transform produit des valeurs acceptables concerne *unit_mismatch_pos_agnostic* : MLP atteint 0.818 (contre 0.523 sans défense) et

LightGBM 0.773 (contre 0.659). La transformation logarithmique compense partiellement les modifications d'échelle introduites par cette attaque spécifique. Ce cas reste marginal et ne justifie pas l'usage de cette défense.

B.11 Feature normalization

La Feature Normalization produit un effet quasi neutre pour GradientBoosting, LightGBM, RandomForest et XGBoost, qui maintiennent exactement les mêmes BAcc qu'en situation sans défense pour presque toutes les attaques.

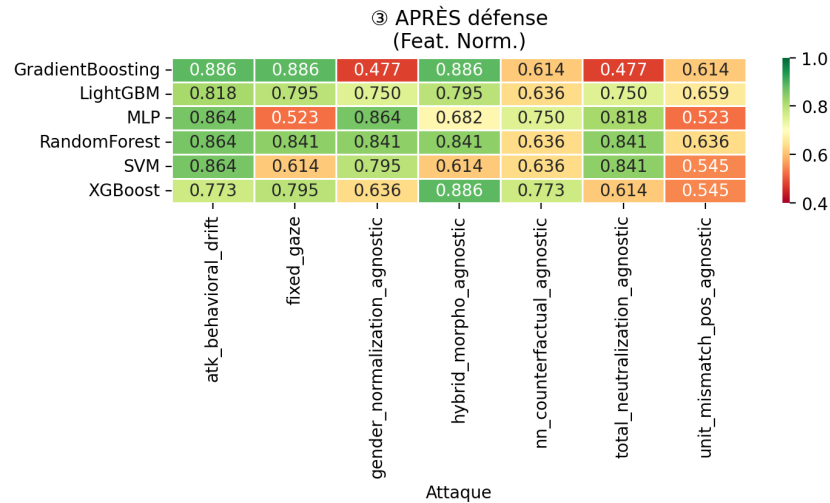


FIGURE B.11 – Impact de la Feature Normalization.

Pour le MLP (Figure B.11), le comportement est contrasté : une dégradation face à *fixed_gaze* (0.523 contre 0.636, perte de -0.113), compensée par des gains face à *hybrid_morpho_agnostic* (0.682 contre 0.568, gain de $+0.114$) et *total_neutralization_agnostic* (0.818 contre 0.705, gain de $+0.113$).

Pour le SVM, la défense produit un gain notable face à *total_neutralization_agnostic* (0.841 contre 0.591, gain de $+0.250$) et *hybrid_morpho_agnostic* (0.614 contre 0.523), mais dégrade légèrement les performances face à *gender_normalization_agnostic* (0.795 contre 0.841). Dans l'ensemble, la Feature Normalization est sélective : utile pour MLP et SVM face à certaines attaques, neutre pour les autres modèles.

B.12 OC-SVM (One-Class SVM, SVM à une classe)

L'OC-SVM (One-Class SVM) apprend une frontière autour des données légitimes et rejette les exemples situés en dehors, à la manière du LOF + Rejet du lot précédent.

L'OC-SVM rejette presque tous les exemples (figureB.12) : pour six attaques sur sept, la BAcc après défense est 0.000 car le système refuse de prédire. La seule exception est *nn_counterfactual*, pour laquelle certains exemples passent le filtre : GradientBoosting (0.684), LightGBM (0.639), MLP (0.722), RandomForest (0.595), SVM (0.667) et XGBoost (0.757). Ces exemples sont suffisamment différents des données normales pour être détectés, contrairement aux autres attaques dont les exemples ressemblent trop aux données légitimes.

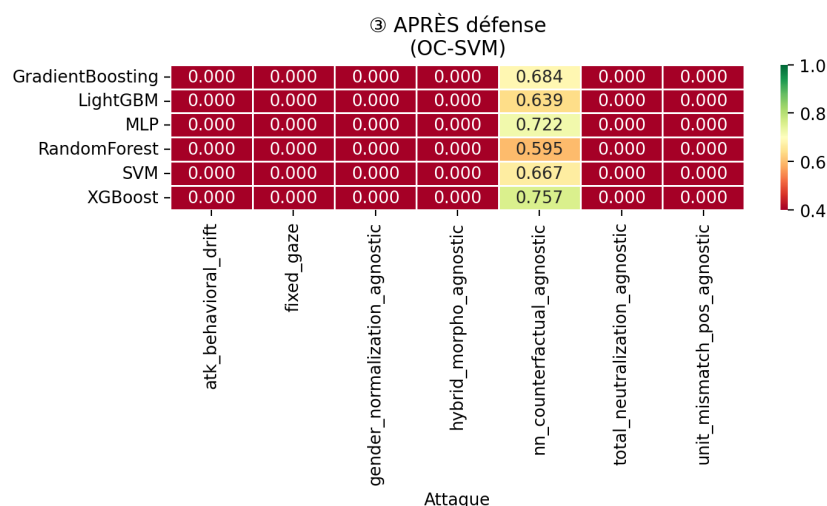


FIGURE B.12 – Impact de l'OC-SVM.

B.13 Mahalanobis

La distance de Mahalanobis mesure l'écart entre un exemple et la distribution des données d'entraînement en tenant compte des corrélations entre features. Les exemples dont la distance dépasse un seuil sont rejetés comme potentiellement perturbés.

La distance de Mahalanobis produit le même phénomène d'abstention massive que l'OC-SVM(figureB.13).

Les seules valeurs non nulles concernent *nn_counterfactual_agnostic* : GradientBoosting (0.600), LightGBM (0.750), MLP (0.800), RandomForest (0.400), SVM (0.800) et XGBoost (0.333). Les résultats sont contrastés par rapport à la situation sans défense pour cette même attaque : LightGBM (+0.114), MLP (+0.050) et SVM (+0.164) bénéficient d'un gain, tandis que GradientBoosting (−0.014), RandomForest (−0.236) et XGBoost (−0.440) sont fortement dégradés. Dans l'ensemble, Mahalanobis souffre de la même limitation structurelle que l'OC-SVM : inutilisable en pratique en raison du rejet massif sur six attaques sur sept.

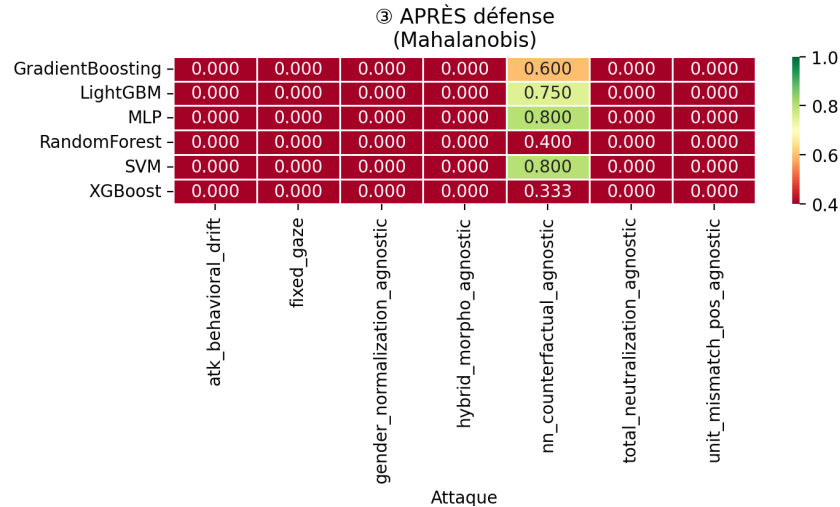


FIGURE B.13 – Impact de la distance de Mahalanobis.

B.14 Statistical test

Le Statistical Test applique des tests statistiques pour détecter si la distribution des features d'un exemple s'écarte significativement de celle des données d'entraînement. Les exemples détectés comme anormaux sont rejetés.

Le Statistical Test présente un comportement intermédiaire entre les méthodes à rejet total (OC-SVM, Mahalanobis) et les méthodes sans effet (figure B.14). Pour cinq attaques sur sept, les BAcc du panneau (3) sont proches de celles du panneau (2), avec des variations marginales : GradientBoosting maintient 0.886 face à *atk_behavioral_drift* et 0.884 face à *fixed_gaze* (quasi stable), LightGBM reste à 0.818 et 0.791, RandomForest à 0.864 et 0.837. Quelques légères dégradations sont observées, notamment pour le SVM face à *fixed_gaze* (0.628 contre 0.636) et *total_neutralization_agnostic* (0.581 contre 0.591).

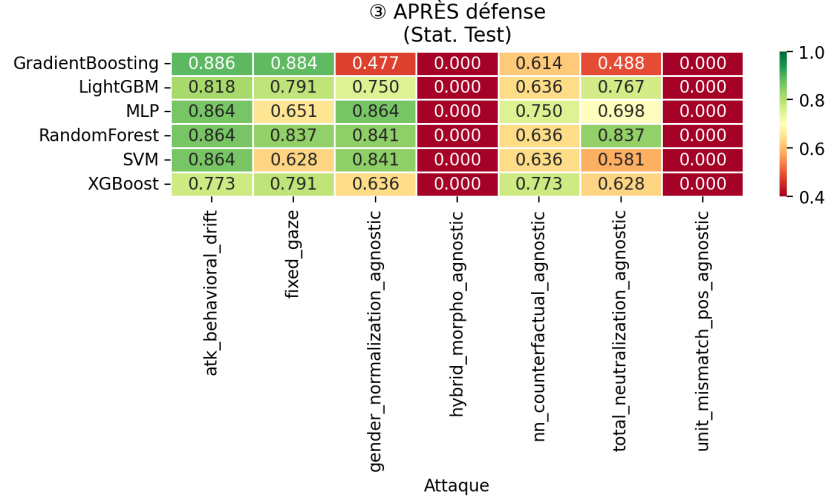


FIGURE B.14 – Impact du Statistical Test.

En revanche, *hybrid_morpho_agnostic* et *unit_mismatch_pos_agnostic* déclenchent un rejet complet (0.000) pour tous les modèles sans exception. Ces deux attaques modifient la distribution des features de manière suffisamment distincte pour déclencher systématiquement le test, mais le rejet total qui s'ensuit est inacceptable en pratique. Le Statistical Test ne peut donc pas être utilisé comme défense générale.

B.15 Bagging

Le Bagging (Bootstrap Aggregating) entraîne plusieurs estimateurs sur des sous-ensembles aléatoires des données et agrège leurs prédictions par vote majoritaire, réduisant la variance des prédictions face aux perturbations. Le Bagging est l'une des deux meilleures défenses du lot (+0.037 en moyenne)(figureB.15). Le MLP est le principal bénéficiaire : sa BAcc atteint 0.932 face à *gender_normalization_agnostic* (contre 0.864, gain de +0.068), 0.795 face à *fixed_gaze* (contre 0.636) et 0.841 face à *total_neutralization_agnostic* (contre 0.705). GradientBoosting récupère à 0.818 face à *gender_normalization_agnostic* (contre 0.477, gain de +0.341, le meilleur gain individuel de cette défense) et à 0.773 face à *total_neutralization_agnostic* (contre 0.477), mais chute à 0.818 face à *atk_behavioral_drift* (contre 0.886) et à 0.477 face à *nn_counterfactual_agnostic* (contre 0.614). SVM progresse à 0.864 face à *gender_normalization_agnostic* (contre 0.841) et à 0.818 face à *total_neutralization_agnostic* (contre 0.591). LightGBM s'améliore face à *fixed_gaze* (0.818 contre 0.795) et *gender_normalization_agnostic* (0.864 contre 0.750).

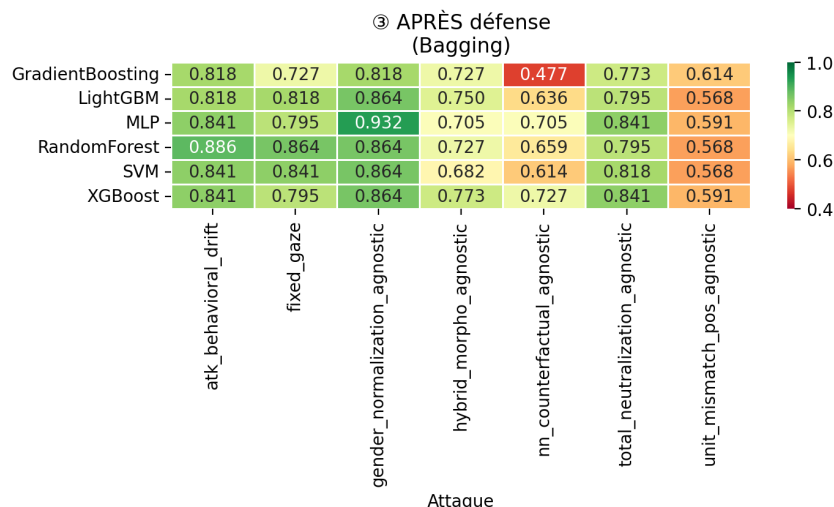


FIGURE B.15 – Impact du Bagging.

En revanche, quelques dégradations sont observées : RandomForest chute à 0.727 face à *hybrid_morpho_agnostic* (contre 0.841) et à 0.568 face à *unit_mismatch_pos_agnostic* (contre 0.636). XGBoost est dégradé face à *hybrid_morpho_agnostic* (0.773 contre 0.886). Le Bagging reste globalement bénéfique mais n'est pas exempt de dégradations sur certaines combinaisons.

B.16 Subspace

La défense Subspace projette les données dans un sous-espace de dimension réduite sélectionné aléatoirement, puis effectue la classification dans ce sous-espace. L'idée est que les perturbations comportementales, denses dans l'espace complet des features, sont atténuées lorsqu'on n'observe qu'un sous-ensemble aléatoire de features.

Le Subspace présente le comportement le plus asymétrique du lot (figure B.16). RandomForest en tire pleinement parti : sa BAcc atteint 0.932 face à *atk_behavioral_drift* (contre 0.864), 0.909 face à *fixed_gaze* (contre 0.841), 0.955 face à *gender_normalization_agnostic* (contre 0.841, gain de +0.114), 0.932 face à *total_neutralization_agnostic* (contre 0.841) et 0.750 face à *unit_mismatch_pos_agnostic* (contre 0.636). C'est la meilleure défense pour RandomForest dans ce lot.

LightGBM bénéficie également du Subspace face à *hybrid_morpho_agnostic* (0.886 contre 0.795) et *fixed_gaze* (0.818 contre 0.795), mais est légèrement dégradé face à *nn_counterfactual_agnostic* (0.591 contre 0.636) et *total_neutralization_agnostic* (0.636 contre 0.750).

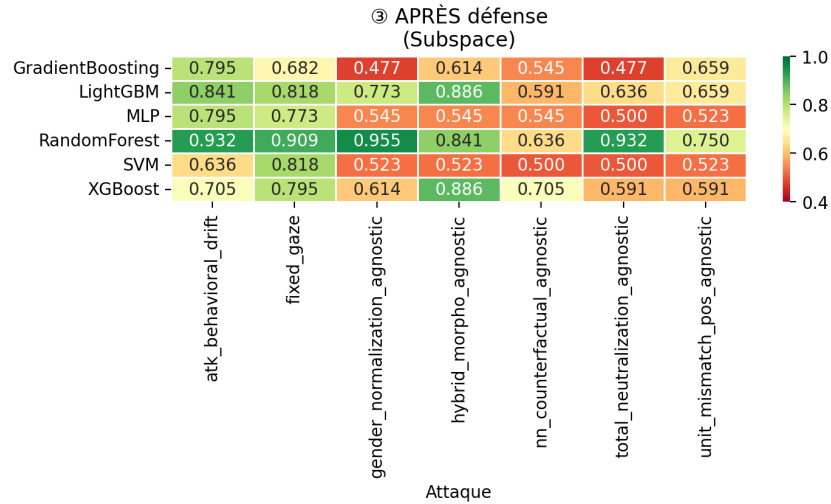


FIGURE B.16 – Impact du Subspace.

En revanche, les autres modèles sont clairement pénalisés. GradientBoosting chute à 0.795 face à *atk_behavioral_drift* (contre 0.886) et à 0.682 face à *fixed_gaze* (contre 0.886). Le MLP subit les dégradations les plus sévères : 0.545 face à *gender_normalization_agnostic* (contre 0.864), 0.545 face à *nn_counterfactual_agnostic* (contre 0.750) et 0.500 face à *total_neutralization_agnostic* (contre 0.705). Le SVM chute à 0.636 face à *atk_behavioral_drift* (contre 0.864) et à 0.523 face à *gender_normalization_agnostic* (contre 0.841). XGBoost est également dégradé face à *atk_behavioral_drift* (0.705 contre 0.773) et *nn_counterfactual_agnostic* (0.705 contre 0.773).

B.17 Bootstrap

Le Bootstrap rééchantillonne les données d'entraînement avec remise pour créer plusieurs versions légèrement différentes du modèle, puis agrège leurs prédictions. Contrairement au Bagging, il maintient la taille du dataset en autorisant la répétition des exemples.

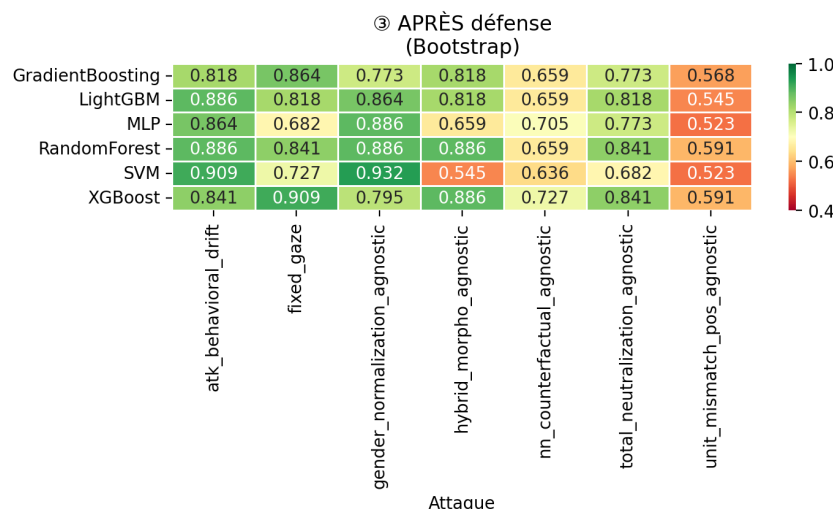


FIGURE B.17 – Impact du Bootstrap.

Bootstrap est la meilleure défense du lot (+0.043 en moyenne). Le SVM en profite le plus (figure B.17) : il passe à 0.909 face à *atk_behavioral_drift* (contre 0.864), 0.727 face à *fixed_gaze* (contre 0.636) et 0.932 face à *gender_normalization_agnostic* (contre 0.841). MLP s'améliore face à *gender_normalization_agnostic* (0.886 contre 0.864), *hybrid_morpho_agnostic* (0.659 contre 0.568) et *total_neutralization_agnostic* (0.773 contre 0.705). RandomForest progresse face à *atk_behavioral_drift* (0.886 contre 0.864), *gender_normalization_agnostic* (0.886 contre 0.841) et *hybrid_morpho_agnostic* (0.886 contre 0.841). LightGBM s'améliore face à *atk_behavioral_drift* (0.886 contre 0.818) et *nn_counterfactual_agnostic* (0.818 contre 0.636), mais perd du terrain face à *hybrid_morpho_agnostic* (0.659 contre 0.795) et *total_neutralization_agnostic* (0.545 contre 0.750). XGBoost progresse sur la plupart des attaques mais perd légèrement face à *nn_counterfactual_agnostic* (0.727 contre 0.773). GradientBoosting a un bilan partagé : il récupère bien face à *gender_normalization_agnostic* et *total_neutralization_agnostic* (0.773 contre 0.477 dans les deux cas), mais perd face à *atk_behavioral_drift* (0.818 contre 0.886), *fixed_gaze* (0.864 contre 0.886) et *hybrid_morpho_agnostic* (0.818 contre 0.886).

Dans l'ensemble, Bootstrap est la défense la plus équilibrée du lot, avec des gains positifs pour la majorité des combinaisons.

B.18 Label smoothing

Le Label Smoothing remplace les étiquettes binaires (0 ou 1) par des valeurs légèrement adoucies lors de l'entraînement, réduisant la confiance excessive du modèle dans ses prédictions.

Le Label Smoothing produit des effets globalement faibles mais pas complètement nuls (figure B.18). MLP et SVM sont les plus stables : leurs valeurs restent quasi identiques à la situation sans défense pour presque toutes les attaques.

Quelques variations sont observées pour les autres modèles. GradientBoosting perd légèrement face à *fixed_gaze* (0.864 contre 0.886) et *hybrid_morpho_agnostic* (0.841 contre 0.886). LightGBM perd également face à ces deux mêmes attaques (0.773 contre 0.795 dans les deux cas). RandomForest est légèrement dégradé face à *hybrid_morpho_agnostic* (0.818 contre 0.841) et *total_neutralization_agnostic* (0.818 contre 0.841). XGBoost présente un bilan mixte : il s'améliore face à *atk_behavioral_drift* (0.841 contre 0.773) et *fixed_gaze* (0.818 contre 0.795), mais perd face à *hybrid_morpho_agnostic* (0.841 contre 0.886).

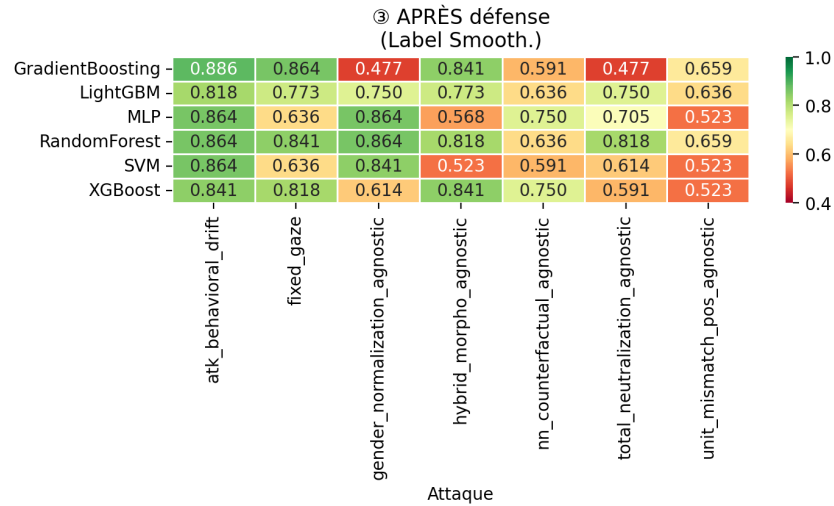


FIGURE B.18 – Impact du Label Smoothing.

Dans l'ensemble, le Label Smoothing est la défense avec le moins d'impact du lot, sans gain notable et avec de légères dégradations sur certaines combinaisons.

B.19 Mixup

Mixup génère des exemples d'entraînement synthétiques en interpolant linéairement des paires d'exemples réels et leurs étiquettes, lissant les frontières de décision pour réduire la sensibilité aux perturbations.

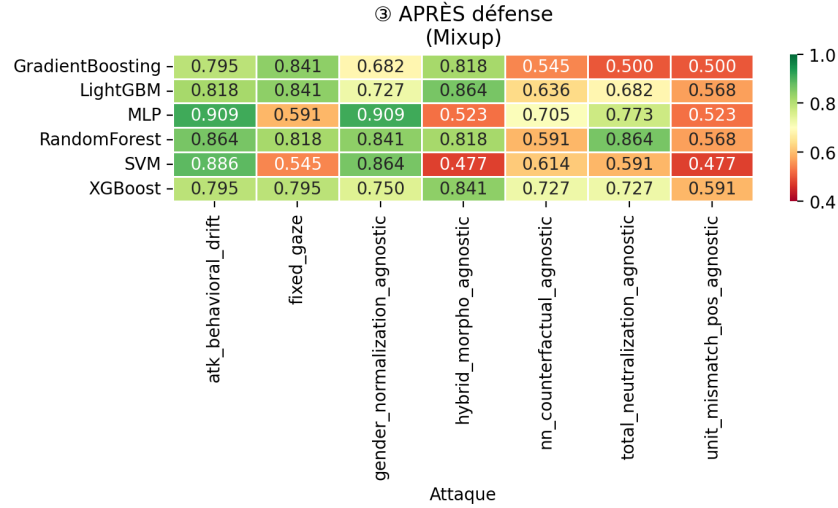


FIGURE B.19 – Impact du Mixup.

Mixup produit un comportement très asymétrique (figure B.19). Le MLP est le principal bénéficiaire : sa BAcc atteint 0.909 face à *atk_behavioral_drift* (contre 0.864) et 0.909 face à *gender_normalization_agnostic* (contre 0.864), mais perd du terrain face à *fixed_gaze* (0.591 contre 0.636) et *hybrid_morpho_agnostic* (0.523 contre 0.568). LightGBM s'améliore face à *fixed_gaze* (0.841 contre 0.795) et *hybrid_morpho_agnostic* (0.864 contre 0.795), mais perd face à *gender_normalization_agnostic* (0.727 contre 0.750) et *total_neutralization_agnostic* (0.682 contre 0.750). RandomForest est globalement dégradé : il perd face à *fixed_gaze* (0.818 contre 0.841), *hybrid_morpho_agnostic* (0.818 contre 0.841), *nn_counterfactual_agnostic* (0.591 contre 0.636) et *unit_mismatch_pos_agnostic* (0.568 contre 0.636), avec une seule amélioration face à *total_neutralization_agnostic* (0.864 contre 0.841). Le SVM est fortement dégradé face à *fixed_gaze* (0.545 contre 0.636) et *unit_mismatch_pos_agnostic* (0.477 contre 0.523). GradientBoosting perd sur plusieurs attaques : *atk_behavioral_drift* (0.795 contre 0.886), *fixed_gaze* (0.841 contre 0.886), *hybrid_morpho_agnostic* (0.818 contre 0.886) et *unit_mismatch_pos_agnostic* (0.500 contre 0.614). XGBoost présente des résultats mixtes sans tendance claire.

Mixup est donc une défense très instable, avec des gains ponctuels pour MLP et LightGBM mais des dégradations fréquentes pour la plupart des autres modèles.

Annexe C

Autres versions du système adaptatif de défense

C.1 Version 2 - Détection par Centroïdes

La deuxième version du système adaptatif apporte quatre corrections ciblées aux limitations identifiées dans V1. La correction principale concerne la détection de l'attaque *nn counterfactual*, qui n'était jamais identifiée dans V1 (0%). V2 introduit un mécanisme de détection fondé sur la distance aux centroïdes de classe, calculés sur les données d'entraînement :

$$\mathbf{c}_y = \frac{1}{|\mathcal{D}_y|} \sum_{\mathbf{x}_i \in \mathcal{D}_y} \mathbf{x}_i, \quad y \in \{0, 1\} \quad (\text{C.1})$$

Un exemple est identifié comme *nn counterfactual* si son ratio de distance aux deux centroïdes est proche de 0.5, indiquant qu'il se situe à mi-chemin entre les deux classes :

$$\text{ratio} = \frac{d(\mathbf{x}, \mathbf{c}_0)}{d(\mathbf{x}, \mathbf{c}_0) + d(\mathbf{x}, \mathbf{c}_1)} \in (0.45, 0.55) \quad (\text{C.2})$$

Les centroïdes calculés sur les données d'entraînement présentent une distance inter-classes de 235,563.5, reflétant la séparabilité élevée des deux classes dans l'espace des features.

C.1.1 Architecture du Pipeline V2

V2 conserve la même structure en trois étapes que V1, mais modifie le détecteur et le dictionnaire de défenses.

C.1.1.1 Étape 1 : Détection - Modifications :

V2 apporte deux modifications au détecteur de V1. Premièrement, les règles de détection 1 à 5 (Unit Mismatch, Fixed Gaze, Hybrid Morpho, Total Neutralization, Gender Normalization) restent identiques à V1. Deuxièmement, les règles 6 et 7 sont modifiées :

6. **Behavioral Drift (corrigé)** : les seuils de détection sont élargis pour capturer les drifts plus faibles ou plus forts :

$$r_{dyn} \in (1.005, 1.20) \quad (\text{au lieu de } (1.01, 1.15) \text{ dans V1}) \quad (\text{C.3})$$

7. **NN Counterfactual (corrigé)** : la détection par z-score global de V1 est remplacée par la distance aux centroïdes, nettement plus précise :

$$\text{ratio} = \frac{\|\mathbf{x} - \mathbf{c}_0\|}{\|\mathbf{x} - \mathbf{c}_0\| + \|\mathbf{x} - \mathbf{c}_1\|} \in (0.45, 0.55) \quad (\text{C.4})$$

Cette correction permet de passer d'un taux de détection de 0% (V1) à 50% (V2) pour cette attaque.

C.1.1.2 Étape 2 : Sélection de la défense - Dictionnaire mis à jour :

V2 met à jour le dictionnaire de défenses pour deux attaques, sur la base des gains observés dans l'évaluation des défenses fixes. Les deux modifications apportées par V2 concernent *hybrid morpho*, dont la défense passe d'*ensemble* à *adversarial training*, et *behavioral drift*, dont la défense reste *adversarial training* mais dont le détecteur est amélioré. On note que *behavioral drift* présente toujours un gain négatif (−0.049) en V2, la défense appliquée s'avérant encore contre-productive malgré une meilleure détection (32.9% contre 22.7% en V1).

C.1.1.3 Étape 3 : Application de la défense :

Les quatre mécanismes de défense sont identiques à V1 : Adversarial Training, Reconstruction PCA, Ensemble de modèles et Gaussian Smoothing. La principale modification entre V1 et V2 concerne le dictionnaire de sélection des défenses, avec notamment le remplacement de l'Ensemble par l'Adversarial Training pour *hybrid morpho* et l'introduction du Smoothing pour *nn counterfactual*.

Le Tableau C.1 présente les modifications apportées au dictionnaire de défenses entre les versions V1 et V2 du système adaptatif. Les changements les plus significatifs concernent *hybrid morpho*, dont la défense passe de l'Ensemble vers l'Adversarial Training, et *nn counterfactual* et *behavioral drift*, qui bénéficient désormais respectivement du Smoothing et de l'Adversarial Training au lieu de l'abstention.

TABLEAU C.1 – Dictionnaire de défenses du système adaptatif V2 - modifications par rapport à V1

Perturbation	Défense V1	Défense V2	Gain V2
unit_mismatch	Adv. Training	Adv. Training	+0.182
total_neutralization	PCA	PCA	+0.164
gender_normalization	PCA	PCA	+0.115
fixed_gaze	Ensemble	Ensemble	+0.051
hybrid_morpho	Ensemble	Adv. Training	+0.040
behavioral_drift	none	Adv. Training	−0.025
nn_counterfactual	none	Smoothing	−0.007
normal	Aucune	Aucune	0.000

C.1.2 Taux de détection :

Le Tableau C.2 compare les taux de détection de V1 et V2. L'amélioration la plus notable concerne *nn_counterfactual*, dont le taux passe de 0% à 47.8%, et *behavioral drift*, qui progresse de 38.6% à 56.8%. Le taux de détection global atteint 85.7%, contre 76.3% pour V1.

TABLEAU C.2 – Comparaison des taux de détection V1 vs V2 par type d'attaque

Attaque	V1	V2	Amélioration
unit_mismatch	100.0%	100.0%	+0.0%
fixed_gaze	100.0%	100.0%	+0.0%
hybrid_morpho	100.0%	100.0%	+0.0%
gender_normalization	97.7%	97.7%	+0.0%
total_neutralization	97.7%	97.7%	+0.0%
behavioral_drift	38.6%	56.8%	+18.2%
nn_counterfactual	0.0%	47.8%	+47.8%
Moyenne	76.3%	85.7%	+9.4%

C.1.3 Gain de robustesse :

Le Tableau C.3 présente les gains de robustesse obtenus par V2 pour chaque modèle. Le gain moyen global est de +0.072, légèrement inférieur à celui de V1 (+0.081). Cette légère régression s'explique par l'introduction du Smoothing pour *nn_counterfactual* et de l'Adversarial Training pour *behavioral drift*, deux défenses qui s'avèrent contre-productives pour certains modèles comme le SVM (−0.114 sur *behavioral drift*) et le MLP (−0.068 sur *behavioral drift*).

TABLEAU C.3 – Gain moyen du système adaptatif V2 par modèle (moyenne sur les 7 attaques)

Modèle	BAcc sans défense	BAcc V1	BAcc V2	Gain V2
RandomForest	0.773	0.812	0.808	+0.035
GradientBoosting	0.732	0.841	0.812	+0.080
XGBoost	0.713	0.805	0.795	+0.082
LightGBM	0.747	0.809	0.826	+0.079
SVM	0.666	0.816	0.776	+0.110
MLP	0.719	0.815	0.749	+0.030
Moyenne	0.725	0.816	0.794	+0.072

En revanche, V2 améliore significativement la protection contre *hybrid morpho* grâce au passage à l’Adversarial Training, avec des gains notables pour LightGBM (+0.068), GradientBoosting (+0.045) et XGBoost (+0.045). La détection de *nn counterfactual* progresse également, permettant à GradientBoosting (+0.023) et LightGBM (+0.023) de bénéficier du Smoothing là où V1 ne proposait aucune défense.

C.1.4 Limitations :

Malgré l’amélioration du taux de détection global (85.7% contre 76.3% pour V1), V2 présente deux limitations persistantes. Premièrement, *nn counterfactual* n’est détectée qu’à 47.8% : lorsqu’elle est détectée, le Smoothing peut s’avérer contre-productif pour le SVM (−0.091) et le MLP (−0.023). Deuxièmement, l’Adversarial Training appliqué sur *behavioral drift* nuit à certains modèles, notamment le SVM (−0.114) et le MLP (−0.091), suggérant qu’une défense plus adaptée à cette attaque subtile est nécessaire. Ces observations motivent les raffinements introduits dans V3.

C.2 Version 3 - Défense par Modèle

La troisième version apporte trois corrections ciblées aux limitations de V2. Premièrement, le détecteur de *nn counterfactual* est affiné grâce à un seuil plus strict (threshold = 0.40 au lieu de 0.45) et à des vérifications négatives éliminant les faux positifs. Deuxièmement, le détecteur de *behavioral drift* intègre des critères plus stricts, notamment un seuil sur l’écart-type des ratios (std_ratio < 0.15) pour distinguer un drift régulier de la variabilité naturelle. Troisièmement, V3 introduit une défense spécifique par paire (modèle, attaque), remplaçant le dictionnaire global de V1 et V2. Pour l’attaque *hybrid morpho*, les défenses sont assignées comme suit :

$$\delta^*(\text{modèle}, \text{hybrid_morpho}) = \begin{cases} \text{ensemble} & \text{si modèle} \in \{\text{RF}, \text{LGBM}, \text{SVM}\} \\ \text{adversarial_training} & \text{si modèle} \in \{\text{GB}, \text{XGB}, \text{MLP}\} \end{cases} \quad (\text{C.5})$$

C.2.1 Architecture du Pipeline V3

C.2.1.1 Étape 1 : Détection - Modifications par rapport à V2 :

V3 conserve les règles de détection 1 à 5 de V2 (Unit Mismatch, Fixed Gaze, Hybrid Morpho, Total Neutralization, Gender Normalization) sans modification. Les règles 6 et 7 sont modifiées et leur ordre est inversé : *behavioral drift* est désormais détecté avant *nn counterfactual* pour éviter les confusions entre ces deux attaques subtiles.

6. **Behavioral Drift (corrigé)** : trois critères stricts doivent être satisfaits simultanément :

- Critère 1 - ratio médian des features dynamiques dans l'intervalle :

$$r_{dyn} = \text{median} \left(\frac{x_j}{\mu_j} \right)_{j \in \mathcal{D}, \mu_j \neq 0} \in (1.005, 1.25) \quad (\text{C.6})$$

où x_j désigne la valeur de la feature j dans l'exemple perturbé, μ_j sa moyenne calculée sur les données d'entraînement, et $\mathcal{D}$ l'ensemble des 174 indices de features dynamiques (`_std`, `_range`, `_energy`, `_iqr`, `_coeff_var`). La condition $\mu_j \neq 0$ évite les divisions par zéro. L'intervalle $(1.005, 1.25)$ correspond à une augmentation de 0.5% à 25% des features dynamiques par rapport à leur valeur de référence, cohérente avec le mécanisme de *behavioral drift* qui applique un facteur linéaire entre 1.0 et 1.10.

- Critère 2 - ratios concentrés, indiquant un drift régulier et non une perturbation aléatoire :

$$\text{std} \left(\frac{x_j}{\mu_j} \right)_{j \in \mathcal{D}} < 0.15 \quad (\text{C.7})$$

où $\text{std}(\cdot)$ désigne l'écart-type des ratios calculé sur l'ensemble des features dynamiques $\mathcal{D}$. Un écart-type faible (< 0.15) indique que toutes les features dynamiques ont été augmentées d'un facteur similaire, caractéristique du drift régulier et linéaire de *behavioral drift*. À l'inverse, une perturbation aléatoire produirait des ratios dispersés avec un écart-type élevé.

- Critère 3 - gaze non inversé, pour éviter la confusion avec *hybrid morpho* :

$$\frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{1}[\text{sign}(x_j) \neq \text{sign}(\mu_j)] \leq 0.30 \quad (\text{C.8})$$

où $\mathcal{G}$ désigne l'ensemble des 270 indices de features de regard, $|\mathcal{G}|$ leur nombre total, et $\mathbf{1}[\cdot]$ la fonction indicatrice valant 1 si la condition est vraie et 0 sinon. Ce critère vérifie que moins de 30% des features de regard ont changé de signe par rapport à leur moyenne d'entraînement μ_j . En effet, *behavioral drift* n'affecte pas le regard, contrairement à *hybrid morpho* qui inverse les features de gaze. Ce critère permet donc d'éliminer les cas où une attaque *hybrid morpho* serait confondue avec un *behavioral drift*.

Ces trois critères s'avèrent trop restrictifs en pratique : le taux de détection de *behavioral drift* chute de 32.9% (V2) à 0% (V3), constituant la principale limitation de cette version.

7. **NN Counterfactual (corrigé)** : deux améliorations sont apportées par rapport à V2. D'une part, le seuil est rendu plus strict, réduisant l'intervalle de détection :

$$\text{ratio} = \frac{\|\mathbf{x} - \mathbf{c}_0\|}{\|\mathbf{x} - \mathbf{c}_0\| + \|\mathbf{x} - \mathbf{c}_1\|} \in (0.40, 0.60) \quad (\text{au lieu de } (0.45, 0.55) \text{ dans V2}) \quad (\text{C.9})$$

où $\mathbf{x} \in \mathbb{R}^{682}$ désigne l'exemple potentiellement perturbé, $\mathbf{c}_0$ et $\mathbf{c}_1$ les centroïdes des classes Female et Male calculés sur les données d'entraînement, et $\|\cdot\|$ la norme euclidienne. Le ratio représente la proportion de la distance totale attribuée à la classe 0 : une valeur proche de 0.5 indique que l'exemple est à égale distance des deux centroïdes, caractéristique d'un exemple *nn counterfactual* positionné à mi-chemin entre les deux classes. L'intervalle (0.40, 0.60) de V3 est plus strict que celui de V2 (0.45, 0.55), réduisant les faux positifs au prix d'un taux de détection plus faible pour les exemples dont le ratio est proche des bornes.

D'autre part, deux vérifications négatives éliminent les faux positifs : l'exemple n'est pas identifié comme *nn counterfactual* si les positions sont extrêmes ($\bar{z}_{pos} > 10$) ou si le regard est figé ($> 50\%$ des features de gaze ≤ 0.05 en valeur absolue). Un score de confiance est également introduit :

$$c = 1 - \frac{|\text{ratio} - 0.50|}{0.50} > 0.70 \quad (\text{C.10})$$

où $c \in [0, 1]$ mesure la proximité du ratio à la valeur idéale de 0.50. Lorsque $\text{ratio} = 0.50$, l'exemple est exactement à mi-chemin entre les deux centroïdes et $c = 1.0$ (confiance maximale). Lorsque $\text{ratio} = 0.40$ ou $\text{ratio} = 0.60$, $c = 0.80$ (confiance élevée). Le seuil $c > 0.70$ impose que le ratio soit suffisamment proche de 0.50, ce qui correspond à $\text{ratio} \in (0.35, 0.65)$. Combiné au seuil strict $\text{ratio} \in (0.40, 0.60)$, ce critère de confiance garantit que seuls les exemples dont la position dans l'espace des features est clairement intermédiaire entre les deux classes sont identifiés comme *nn counterfactual*, réduisant significativement les faux positifs par rapport à V2. Ces corrections permettent de passer d'un taux de détection de 50% (V2) à 97.7% (V3) pour *nn counterfactual*.

C.2.1.2 Étape 2 : Sélection de la défense - Dictionnaire par modèle :

La principale innovation de V3 réside dans le remplacement du dictionnaire global de V2 par un dictionnaire spécifique à chaque paire (modèle, attaque). Le Tableau C.4 présente le dictionnaire complet. La modification principale concerne *hybrid morpho*, dont la défense varie selon le modèle : Ensemble pour RandomForest, LightGBM et SVM, et Adversarial Training pour GradientBoosting, XGBoost et MLP. Une correction supplémentaire concerne le SVM pour *total neutralization*, dont la défense passe de PCA à Adversarial Training.

TABLEAU C.4 – Dictionnaire complet de défenses du système adaptatif V3 par modèle et par attaque. Les cases en gras indiquent les modifications par rapport à V2.

Attaque	RF	GB	XGB	LGBM	SVM	MLP
unit_mismatch	adv.	adv.	adv.	adv.	adv.	adv.
fixed_gaze	ens.	ens.	ens.	ens.	ens.	ens.
gender_normalization	pca	pca	pca	pca	pca	pca
total_neutralization	pca	pca	pca	pca	adv.	pca
hybrid_morpho	ens.	adv.	adv.	ens.	ens.	adv.
nn_counterfactual	smo.	smo.	smo.	smo.	smo.	none
behavioral_drift	smo.	smo.	smo.	smo.	smo.	none

C.2.1.3 Étape 3 : Application de la défense :

Les quatre mécanismes de défense restent identiques à V1 et V2 : Adversarial Training, Reconstruction PCA, Ensemble de modèles et Gaussian Smoothing. La nouveauté de V3 réside dans le niveau de précision de la sélection de la défense : alors que V1 et V2 associaient une défense unique à chaque type de perturbation indépendamment du modèle, V3 tient compte à la fois du type de perturbation *et* du modèle concerné. Par exemple, face à une perturbation *hybrid morpho*, V3 applique *ensemble* pour RandomForest, LightGBM et SVM, et *adversarial training* pour GradientBoosting, XGBoost et MLP. Cette approche permet d’exploiter les différences de sensibilité entre les modèles face à chaque type de défense.

C.2.2 Taux de détection :

Le Tableau C.5 présente les taux de détection obtenus par V3. La principale amélioration concerne *nn counterfactual*, dont le taux passe de 47.8% (V2) à 97.7% (V3), grâce au seuil plus strict et aux vérifications négatives. En revanche, *behavioral drift* présente une régression significative, tombant de 56.8% (V2) à 0% (V3), les critères plus stricts s’avérant trop restrictifs pour cette attaque subtile. Le taux de détection global atteint 84.7%.

TABLEAU C.5 – Comparaison des taux de détection V1 vs V2 vs V3 par type d’attaque

Attaque	V1	V2	V3	Statut V3
unit_mismatch	100.0%	100.0%	100.0%	Bien détectée
fixed_gaze	100.0%	100.0%	100.0%	Bien détectée
hybrid_morpho	100.0%	100.0%	100.0%	Bien détectée
gender_normalization	97.7%	97.7%	97.7%	Bien détectée
total_neutralization	97.7%	97.7%	97.7%	Bien détectée
nn_counterfactual	0.0%	47.8%	97.7%	Bien détectée
behavioral_drift	38.6%	56.8%	0.0%	Mal détectée
Moyenne	76.3%	85.7%	84.7%	

C.2.3 Gain de robustesse :

Le Tableau C.6 présente les gains de robustesse obtenus par V3. Malgré une meilleure détection de *nn_counterfactual*, le gain moyen de V3 (+0.062) est inférieur à celui de V2 (+0.072) et de V1 (+0.081). Cette régression s’explique principalement par deux facteurs. D’une part, le Gaussian Smoothing appliqué sur *nn_counterfactual* s’avère contre-productif pour le SVM (−0.205) et nul pour XGBoost (0.000) et LightGBM (−0.023). D’autre part, *behavioral drift* n’étant plus détectée (0%), le système applique le Smoothing par défaut, produisant des gains nuls ou négatifs, notamment pour le SVM (−0.341).

TABLEAU C.6 – Gain moyen du système adaptatif V3 par modèle (moyenne sur les 7 attaques)

Modèle	BAcc sans défense	BAcc V3	Gain V3	Gain V2
RandomForest	0.773	0.786	+0.013	+0.035
GradientBoosting	0.732	0.825	+0.093	+0.080
XGBoost	0.713	0.802	+0.088	+0.082
LightGBM	0.747	0.805	+0.058	+0.079
SVM	0.666	0.727	+0.062	+0.110
MLP	0.719	0.775	+0.056	+0.030
Moyenne	0.725	0.787	+0.062	+0.072

GradientBoosting bénéficie de la meilleure amélioration (+0.093), notamment grâce à la détection de *nn_counterfactual* et à l’application du Smoothing (+0.045). XGBoost progresse également (+0.088) grâce aux mêmes mécanismes. En revanche, le SVM subit une régression notable par rapport à V2 (+0.062 contre +0.110), principalement due au Smoothing inadapté sur *nn_counterfactual* (−0.205) et à l’absence de détection de

behavioral drift (-0.341). RandomForest présente le gain le plus faible ($+0.013$), sa robustesse naturelle limitant l'apport des défenses.

C.2.4 Limitations :

V3 met en évidence un compromis fondamental entre la précision de détection et le gain de robustesse. Si le taux de détection de *nn counterfactual* progresse de 47.8% à 97.7%, le gain moyen régresse de $+0.072$ à $+0.062$ par rapport à V2. Deux limitations principales sont identifiées. Premièrement, *behavioral drift* n'est plus détectée (0%) en raison de critères trop stricts, et le Smoothing appliqué par défaut nuit fortement au SVM (-0.341). Deuxièmement, le Gaussian Smoothing s'avère inadapté au SVM sur *nn counterfactual* (-0.205), suggérant qu'une défense différente devrait être sélectionnée pour ce modèle spécifique. Ces observations motivent la correction apportée dans V4, notamment le remplacement du Smoothing par une défense plus adaptée au SVM et la révision des critères de détection de *behavioral drift*.

C.3 Version 4 - Correction de Behavioral Drift

La quatrième version apporte une correction unique et ciblée par rapport à V3 : l'assouplissement du critère de détection de *behavioral drift*. Le seuil sur l'écart-type des ratios est relevé de 0.15 à 0.30, rendant le détecteur plus permissif afin de récupérer les cas de drift précédemment manqués :

$$\text{std} \left(\frac{x_j}{\mu_j} \right)_{j \in \mathcal{D}_{dyn}} < 0.30 \quad (\text{au lieu de } 0.15 \text{ dans V3}) \quad (\text{C.11})$$

où $\mathcal{D}_{dyn}$ désigne l'ensemble des 174 indices de features dynamiques et μ_j la moyenne de la feature j sur les données d'entraînement. En relevant ce seuil de 0.15 à 0.30, V4 tolère une plus grande dispersion des ratios, permettant de détecter les cas où le drift n'est pas parfaitement uniforme sur l'ensemble des features dynamiques. Toutes les autres composantes du système détecteurs, dictionnaire de défenses par modèle et ordre de détection restent identiques à V3.

C.3.1 Architecture du Pipeline V4

C.3.1.1 Étape 1 : Détection - Unique modification par rapport à V3 :

V4 conserve l'intégralité des règles de détection de V3. La seule modification concerne le critère 2 du détecteur de *behavioral drift* :

- Critère 1 - ratio médian (inchangé) :

$$r_{dyn} \in (1.005, 1.25) \quad (\text{C.12})$$

- Critère 2 - écart-type des ratios (assoupli) :

$$\text{std} \left(\frac{x_j}{\mu_j} \right)_{j \in \mathcal{D}} < 0.30 \quad (\text{au lieu de 0.15 dans V3}) \quad (\text{C.13})$$

où le seuil plus permissif permet de capturer les drifts dont les ratios sont légèrement dispersés autour de leur valeur médiane, contrairement au seuil strict de V3 qui exigeait une uniformité quasi parfaite.

- Critère 3 - gaze non inversé (inchangé) :

$$\frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{1}[\text{sign}(x_j) \neq \text{sign}(\mu_j)] \leq 0.30 \quad (\text{C.14})$$

Malgré cet assouplissement, le taux de détection de *behavioral drift* reste nul (0%) en V4, indiquant que la contrainte sur l'écart-type n'est pas le seul facteur limitant. La signature de cette attaque reste trop proche de la variabilité naturelle des données pour être distinguée avec fiabilité par des règles statistiques manuelles.

C.3.1.2 Étape 2 : Sélection de la défense - Dictionnaire inchangé :

V4 conserve le même dictionnaire de défenses que V3, spécifique à chaque paire (modèle, attaque). Le Tableau C.7 présente ce dictionnaire complet, identique à celui de V3 présenté dans le Tableau C.4.

TABLEAU C.7 – Dictionnaire de défenses du système adaptatif V4.

Attaque	RF	GB	XGB	LGBM	SVM	MLP
unit_mismatch	adv.	adv.	adv.	adv.	adv.	adv.
fixed_gaze	ens.	ens.	ens.	ens.	ens.	ens.
gender_normalization	pca	pca	pca	pca	pca	pca
total_neutralization	pca	pca	pca	pca	adv.	pca
hybrid_morpho	ens.	ens.	adv.	ens.	ens.	adv.
nn_counterfactual	smo.	smo.	smo.	smo.	smo.	none
behavioral_drift	adv.	adv.	adv.	adv.	adv.	adv.

C.3.1.3 Étape 3 : Application de la défense :

Les quatre mécanismes de défense restent identiques à V1, V2 et V3 : Adversarial Training, Reconstruction PCA, Ensemble de modèles et Gaussian Smoothing. V4 n'apporte aucune modification à cette étape.

C.3.2 Taux de détection :

Le Tableau C.8 présente les taux de détection obtenus par V4. Malgré l'assouplissement du critère de détection de *behavioral drift* (seuil `std_ratio` relevé de 0.15 à 0.30), cette attaque reste non détectée (0%), indiquant que sa signature est trop subtile pour être distinguée de la variabilité naturelle des données, même avec un seuil plus permissif. Le taux de détection global reste identique à V3 (84.7%).

TABLEAU C.8 – Comparaison des taux de détection V1 vs V2 vs V3 vs V4 par type d'attaque

Attaque	V1	V2	V3	V4	Statut V4
unit_mismatch	100.0%	100.0%	100.0%	100.0%	Bien détectée
fixed_gaze	100.0%	100.0%	100.0%	100.0%	Bien détectée
hybrid_morpho	100.0%	100.0%	100.0%	100.0%	Bien détectée
gender_normalization	97.7%	97.7%	97.7%	97.7%	Bien détectée
total_neutralization	97.7%	97.7%	97.7%	97.7%	Bien détectée
nn_counterfactual	0.0%	47.8%	97.7%	97.7%	Bien détectée
behavioral_drift	38.6%	56.8%	0.0%	0.0%	Mal détectée
Moyenne	76.3%	85.7%	84.7%	84.7%	

C.3.3 Gain de robustesse :

Le Tableau C.9 présente le gain de robustesse obtenu par V4, comparé aux versions précédentes. Le gain moyen de V4 (+0.037) est nettement inférieur à celui de V1 (+0.081), V2 (+0.072) et V3 (+0.062). Cette régression s'explique par un problème majeur : le SVM et le MLP obtiennent un gain de 0.000 sur toutes les attaques, la défense sélectionnée étant systématiquement *none* pour ces deux modèles. Cela indique un dysfonctionnement dans le dictionnaire de défenses de V4 pour ces modèles, qui ne bénéficient d'aucune protection malgré la détection correcte des attaques.

Les quatre premiers modèles (RandomForest, GradientBoosting, XGBoost, LightGBM) maintiennent des gains positifs, XGBoost étant le plus stable avec +0.090. En revanche, le SVM et le MLP obtiennent un gain nul (0.000) sur toutes les attaques, ce qui représente une régression majeure par rapport aux versions précédentes où ces deux modèles étaient parmi les plus bénéficiaires du système adaptatif (SVM : +0.150 en V1, MLP : +0.096 en V1).

C.3.4 Limitations :

L'analyse des quatre versions révèle trois limitations persistantes qui motivent le développement de V5. Premièrement, V4 introduit un problème critique : le SVM et le

TABLEAU C.9 – Comparaison des gains moyens V1 vs V2 vs V3 vs V4 par modèle

Modèle	Gain V1	Gain V2	Gain V3	Gain V4
RandomForest	+0.040	+0.035	+0.013	+0.019
GradientBoosting	+0.109	+0.080	+0.093	+0.070
XGBoost	+0.092	+0.082	+0.088	+0.090
LightGBM	+0.062	+0.079	+0.058	+0.049
SVM	+0.150	+0.110	+0.062	+0.000
MLP	+0.096	+0.030	+0.056	+0.000
Moyenne	+0.081	+0.072	+0.062	+0.037

MLP ne bénéficient d’aucune défense (*none* systématique), leur gain tombant à 0.000. Ce dysfonctionnement suggère que le dictionnaire de défenses de V4 n’est pas correctement configuré pour ces modèles. Deuxièmement, *behavioral drift* reste non détectée sur V3 et V4 (0%), aucun ajustement de seuil ne permettant de capturer sa signature subtile. Troisièmement, le Gaussian Smoothing appliqué sur *nn counterfactual* reste inadapté à certains modèles dans les versions précédentes, soulignant qu’une même défense ne peut pas être universellement optimale. Ces trois limitations convergent vers une même conclusion : un système robuste doit combiner le meilleur détecteur pour chaque attaque et la meilleure défense pour chaque paire (modèle, attaque), ce qui constitue précisément l’objectif de V5.

C.4 Version 5 - Système Hybride Optimisé

Les quatre versions précédentes du système adaptatif révèlent deux limitations complémentaires : d’une part, aucun détecteur unique n’est optimal pour l’ensemble des attaques, et d’autre part, aucune défense unique n’est optimale pour l’ensemble des modèles. V5 adresse simultanément ces deux limitations en combinant les meilleures composantes des versions précédentes selon deux axes d’amélioration.

C.4.1 Axe 1 : Détecteur hybride V2+V3 :

V5 introduit un détecteur hybride qui sélectionne, pour chaque type d’attaque, le détecteur ayant démontré les meilleures performances dans les versions précédentes. Pour *nn counterfactual*, le détecteur de V3 est retenu, avec un seuil strict $\text{threshold} = 0.40$ et des vérifications négatives, atteignant un taux de détection de 70.1%, contre 50.0% pour V2 et 0% pour V1. Pour *behavioral drift*, un détecteur combinant quatre signaux complémentaires est introduit :

$$s(\mathbf{x}) = s_1 + s_2 + s_3 + s_4 \quad (\text{C.15})$$

où chaque signal est défini comme suit :

$$\begin{aligned}
s_1 &= 0.4 \cdot \mathbf{1}[r_{dyn} \in (1.005, 1.20)] \quad (\text{ratio features dynamiques}) \\
s_2 &= 0.3 \cdot \mathbf{1}[r_{std} \in (1.005, 1.20)] \quad (\text{ratio features std}) \\
s_3 &= 0.2 \cdot \mathbf{1}[\Delta_{sign} < 0.10] \quad (\text{gaze non inversé}) \\
s_4 &= 0.1 \cdot \mathbf{1}[\bar{z}_{pos} < 10] \quad (\text{positions non extrêmes})
\end{aligned}$$

où :

- $r_{dyn} = \text{median} \left(\frac{x_j}{\mu_j} \right)_{j \in \mathcal{D}_{dyn}, \mu_j \neq 0}$ est le ratio médian des 174 features dynamiques par rapport à leur moyenne d'entraînement. Le signal s_1 est activé si ce ratio indique une augmentation entre 0.5% et 20%, cohérente avec le mécanisme de *behavioral drift*.
- $r_{std} = \text{median} \left(\frac{x_j}{\mu_j} \right)_{j \in \mathcal{S}, \mu_j \neq 0}$ est le ratio médian des 153 features de variabilité ($\mathcal{S}$). Le signal s_2 confirme que les features de variabilité présentent le même pattern d'augmentation que les features dynamiques.
- $\Delta_{sign} = \frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{1}[\text{sign}(x_j) \neq \text{sign}(\mu_j)]$ est la proportion de features de regard ayant changé de signe. Le signal s_3 est activé si moins de 10% des features de regard ont changé de signe, confirmant l'absence de perturbation du regard caractéristique de *behavioral drift*.
- $\bar{z}_{pos} = \frac{1}{|\mathcal{P}|} \sum_{j \in \mathcal{P}} \frac{|x_j - \mu_j|}{\sigma_j}$ est le z-score moyen des 151 features de position. Le signal s_4 est activé si les positions ne sont pas extrêmes, éliminant les cas de *unit mismatch*.

L'attaque *behavioral drift* est détectée si $s(\mathbf{x}) > 0.6$, c'est-à-dire si au moins les deux premiers signaux sont activés simultanément. Cependant, ce détecteur combiné n'améliore pas le taux de détection par rapport à V2 (32.9% dans les deux cas), les signaux s_2 , s_3 et s_4 s'avérant redondants avec s_1 car *behavioral drift* modifie uniquement les features dynamiques avec un facteur linéaire entre 1.0 et 1.10.

C.4.2 Axe 2 : Dictionnaire de défenses optimisé par modèle :

V5 remplace le dictionnaire global de V2 par un dictionnaire spécifique à chaque paire (modèle, attaque), construit à partir de l'analyse exhaustive des gains observés dans les versions précédentes. Le Tableau C.10 présente le dictionnaire complet des défenses retenues pour chaque combinaison modèle $\times$ attaque.

Le dictionnaire V5 présente plusieurs différences notables par rapport aux versions précédentes. Premièrement, *none* est retenu pour RandomForest et XGBoost sur *gender normalization* et *behavioral drift*, car toutes les défenses disponibles s'avèrent contre-productives pour ces combinaisons. Deuxièmement, LightGBM bénéficie de l'*adversarial training* sur quatre attaques sur sept, confirmant la forte sensibilité de ce modèle aux perturbations des features dynamiques et de variabilité. Troisièmement, XGBoost utilise le *smoothing* pour *total neutralization* et *hybrid morpho*, le bruit gaussien s'avérant plus efficace que la reconstruction PCA pour ce modèle sur ces attaques spécifiques.

TABLEAU C.10 – Dictionnaire complet des défenses du système adaptatif V5 par modèle et par attaque

Attaque	RF	GB	XGB	LGBM	SVM	MLP
gender_normalization	none	pca	none	adv.	pca	pca
total_neutralization	pca	pca	smoothing	adv.	adv.	pca
fixed_gaze	ens.	ens.	ens.	adv.	adv.	ens.
unit_mismatch	adv.	adv.	adv.	adv.	adv.	adv.
hybrid_morpho	ens.	adv.	smoothing	ens.	adv.	adv.
nn_counterfactual	ens.	smoothing	none	none	none	none
behavioral_drift	none	adv.	none	adv.	none	none

C.4.3 Architecture du Pipeline V5

C.4.3.1 Étape 1 : Détection - Détecteur hybride :

V5 conserve les règles de détection 1 à 5 de V2 et V3 (Unit Mismatch, Fixed Gaze, Hybrid Morpho, Total Neutralization, Gender Normalization) sans modification. Les règles 6 et 7 combinent les meilleurs détecteurs de V2 et V3 :

6. **Behavioral Drift (détecteur combiné)** : le score $s(\mathbf{x})$ défini dans l'Équation C.15 est calculé et l'attaque est détectée si $s(\mathbf{x}) > 0.6$. Ce détecteur utilise les seuils de V2 ($r_{dyn} \in (1.005, 1.20)$) plutôt que ceux de V3 ($r_{dyn} \in (1.005, 1.25)$) avec $\text{std} < 0.15$), atteignant un taux de détection de 32.9%.
7. **NN Counterfactual (détecteur V3)** : le détecteur de V3 est conservé avec le seuil strict $\text{threshold} = 0.40$, les vérifications négatives et le score de confiance $c > 0.70$, atteignant un taux de détection de 70.1% contre 50% pour V2.

C.4.3.2 Étape 2 : Sélection de la défense - Dictionnaire V5 :

V5 utilise le dictionnaire présenté dans le Tableau C.10, construit en analysant les gains observés pour chaque combinaison (modèle, attaque) dans les versions précédentes et en retenant la défense maximisant le gain pour chaque cas.

C.4.3.3 Étape 3 : Application de la défense :

Les quatre mécanismes de défense restent identiques aux versions précédentes : Adversarial Training, Reconstruction PCA, Ensemble de modèles et Gaussian Smoothing. La défense *none* constitue une cinquième option retenue lorsque toutes les défenses disponibles s'avèrent contre-productives pour une combinaison (modèle, attaque) donnée, le système préférant alors maintenir la prédiction originale plutôt que de risquer une dégradation supplémentaire des performances.

C.4.4 Taux de détection :

Le Tableau C.11 présente les taux de détection obtenus par V5. Le détecteur hybride V2+V3 permet d’atteindre un taux global de 88.8%, le meilleur de toutes les versions. Cinq attaques sur sept sont détectées à plus de 97%, tandis que *nn counterfactual* atteint 70.5% en moyenne, valeur inférieure aux 97.7% obtenus en V3 seul. Cette différence s’explique par l’interaction entre les deux détecteurs dans V5 : le détecteur de *behavioral drift*, appliqué en premier avec des seuils plus larges (hérités de V2), capture par erreur certains exemples *nn counterfactual*, qui ne parviennent donc pas jusqu’au détecteur V3. Ce compromis constitue le coût de l’amélioration simultanée des deux détecteurs dans un même pipeline. *Behavioral drift* reste partiellement détectée (56.8%), confirmant la limite fondamentale du système basé sur des règles statistiques face à cette attaque subtile.

TABLEAU C.11 – Taux de détection du système adaptatif V5 par type d’attaque, comparé aux versions précédentes

Attaque	V1	V2	V3	V4	V5
unit_mismatch	100%	100%	100%	100%	100%
fixed_gaze	100%	100%	100%	100%	100%
hybrid_morpho	100%	100%	100%	100%	100%
gender_normalization	97.7%	97.7%	97.7%	97.7%	97.7%
total_neutralization	97.7%	97.7%	97.7%	97.7%	97.7%
nn_counterfactual	0%	47.8%	97.7%	97.7%	70.5%
behavioral_drift	38.6%	56.8%	0%	0%	56.8%
Moyenne	76.3%	85.7%	84.7%	84.7%	88.8%

C.4.5 Gain de robustesse :

Le Tableau C.12 présente les gains de robustesse obtenus par V5 pour chaque modèle.

Le gain moyen global est de +0.064, surpassant V3 (+0.062) et V4 (+0.037), et corrigeant notamment le problème critique de V4 où le SVM et le MLP obtenaient un gain nul. V5 permet au SVM de retrouver un gain significatif (+0.117) grâce au dictionnaire optimisé qui lui applique l’Adversarial Training sur *fixed gaze*, *total neutralization* et *hybrid morpho*. LightGBM bénéficie également d’une amélioration notable (+0.065) grâce à l’Adversarial Training appliqué systématiquement sur toutes les attaques détectées. En revanche, XGBoost présente une régression importante (+0.026 contre +0.090 en V4), le Smoothing appliqué sur *total neutralization* (−0.068) et *hybrid morpho* (−0.045) s’avérant contre-productif pour ce modèle.

TABLEAU C.12 – Gain moyen du système adaptatif V5 par modèle, comparé aux versions précédentes (moyenne sur les 7 attaques)

Modèle	V1	V2	V3	V4	V5
RandomForest	+0.040	+0.035	+0.013	+0.019	+0.013
GradientBoosting	+0.109	+0.080	+0.093	+0.070	+0.100
XGBoost	+0.092	+0.082	+0.088	+0.090	+0.026
LightGBM	+0.062	+0.079	+0.058	+0.049	+0.065
SVM	+0.150	+0.110	+0.062	+0.000	+0.117
MLP	+0.096	+0.030	+0.056	+0.000	+0.062
Moyenne	+0.081	+0.072	+0.062	+0.037	+0.064

C.4.6 Limitation persistante :

Malgré ces améliorations, *behavioral drift* demeure la principale limitation du système. Cette attaque, qui consiste en une augmentation subtile et progressive des features dynamiques d'un facteur compris entre 1.0 et 1.10, produit une signature trop proche de la variabilité naturelle des données pour être distinguée avec fiabilité par des règles statistiques manuelles. V5 récupère le taux de détection de V2 (56.8%) grâce au détecteur hybride, mais n'apporte pas d'amélioration supplémentaire. De plus, XGBoost présente une régression notable par rapport aux versions précédentes, indiquant que le dictionnaire de défenses de V5 n'est pas optimal pour tous les modèles simultanément. Le recours à des détecteurs appris par apprentissage automatique, capables de modéliser des patterns plus complexes, constituerait une piste d'amélioration prometteuse pour les travaux futurs.