

Réunion 10 juillet 2026

- Expériences 5 et 6 en séparant paires positives et négatives
 - Perturbations
 - Dataset
 - Origine des NaN
 - Autres Résultats
- Passage en format CLI du script `GenericAutomatizedPerturbation.py`
- Outils pour l'expérience de contrôle
- Quantifier l'intensité de perturbation de l'espace de représentation

Dataset

Construction des données:

- 1 question, 3 pdf (limitations matérielles du drone, code de la défense - matériel de guerre, fausses infos pdt les JO/JP)
- 50% de textes pertinents
- 28 paires

Construction des perturbations:

- fréquence et TF-IDF des documents
- contenu de la question

Quelles sont les trois principales catégories de limitations techniques affectant les systèmes de drones, et comment chacune influence-t-elle l'efficacité opérationnelle de ces vecteurs ?

word	freq	TF-IDF
matériel	16	0.0627430130086598
système	11	0.0542242908052481
limitation	6	0.0495565089630139
drone	10	0.0441649020535004
défense	10	0.0440955901918619
prendre	6	0.0425658179515979
compte	5	0.0404563824330286
action	8	0.0401684995180441
article	9	0.0400092952253956
ministre	8	0.03913497111578
aérien	7	0.0366033210120314
acteur	6	0.0337084867312698
guerre	8	0.0336043799517422

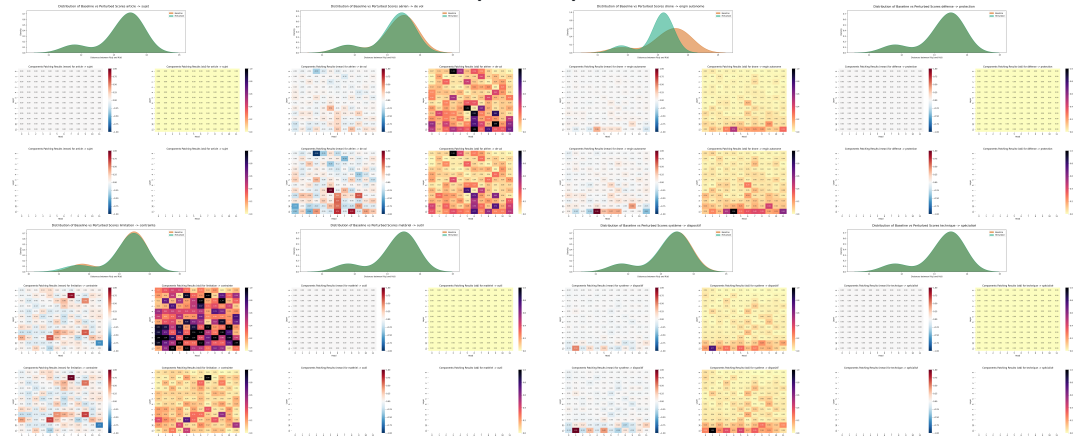
Perturbations

Mot remplacé	Niveau 1	Niveau 2	Niveau 3	Niveau 4
aérien	de vol	poétique	épicurien	acide tracté
article	sujet	déterminant	table	luge pontificale
défense	protection	corne	fromage	poire retournée
drone	vecteur	transformation	gazon	ver de verre
limitation	contrainte	bornage	animation	plastique anarchique
matériel	outil	concret	post-it	porte sans poivre
système	dispositif	ensemble	panneau	esperluette en vent
technique	spécialisé	difficile	magique	feuille entubée

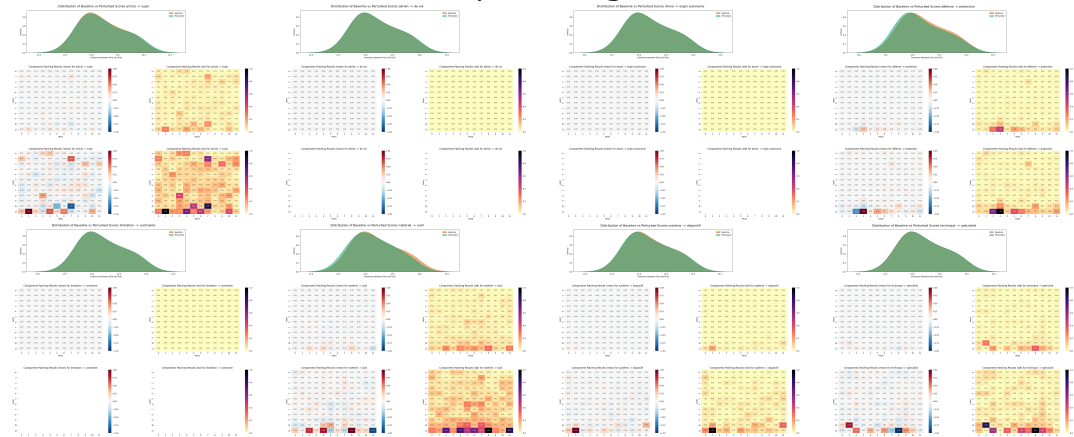
- 8 perturbations `replace` de niveau 1
- 8 perturbations `replace` de niveau 2
- 8 perturbations `replace` de niveau 3
- 8 perturbations `replace` de niveau 4
- 4 perturbations `full_replace` (une de chaque niveau, composée des 8 perturbations précédentes)

Origine des NaN

Perturbations de Niveau 1 sur les paires positives



Perturbations de Niveau 1 sur les paires négatives



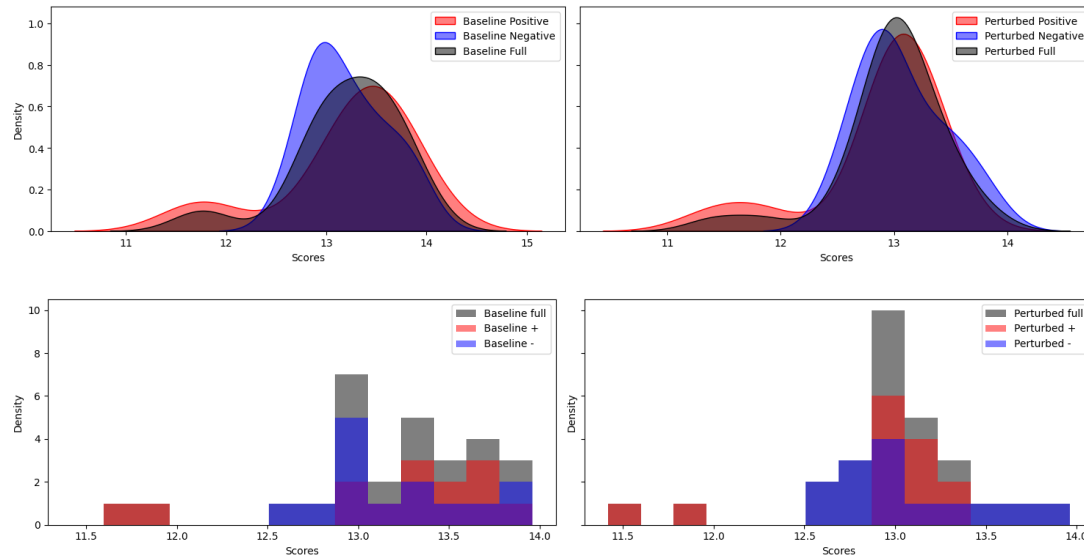
Nombre de documents contenant le mot :

Mot remplacé	Données positives	Données négatives	Dataset entier
aérien	5	0	5
article	0	3	3
défense	0	4	4
drone	7	0	7
limitation	4	0	4
matériel	0	5	5
système	5	1	6
technique	0	1	1

$$P_n = \frac{\bar{p}_n - p_{\hat{D}}}{p_{\check{D}} - p_{\hat{D}}} = \frac{p - p}{p - p} = \frac{0}{0} = NaN$$

⇒ Les NaN sont dûs aux documents *insensibles* à une perturbation

Distribution des paires dans l'espace de représentation



Il n'y a pas de distinction entre les paires positives et négatives !

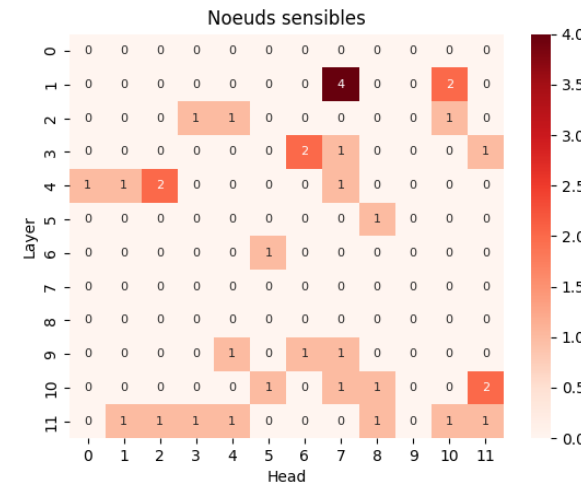
Deux documents avec un score < 12 :

Des limitations à prendre en compte

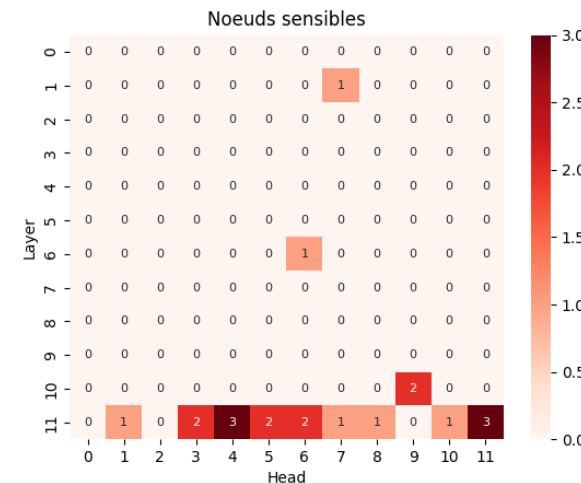
21

Noeuds sensibles

- Majoritairement dans les deux dernières couches
- Couches centrales : issu d'une seule perturbation (donc 1 seul ss-dataset)
 - aérien -> ... pour N1 et N2
 - système -> ... pour N3
- Un comportement anarchique pour les perturbations N4 issus de
 - matériel -> porte sans poivre et
 - article -> luge pontificale



ER Neg



ER Pos

Bilan

Prévention de la dilution des perturbations

- coté perturbation:
 - full_replace (avec choix stratégique)
 - append et prepend (car impactent tout)
- coté dataset
 - augmenter le vocabulaire commun
 - documents plus gros
 - faire les expés sur la partie du dataset qui est impactée
- coté scores
 - continuer à utiliser le rescaling
 - prendre en compte le nombre de documents insensible dans le critère de sensibilité

Format CLI pour `GenericAutomatizedPerturbation.py`

2 commandes:

- `values` : calcule les scores et matrices de sensibilité
- `graphs` : Génère les graphiques

```
usage: GenericAutomatizedPerturbation.py graphs [-h] [--output OUTPUT] input
```

positional arguments:

input Fichier contenant les données à mettre sous forme de graphique

options:

-h, --help show this help message and exit

--output OUTPUT, -o OUTPUT

Format CLI pour GenericAutomatizedPerturbation.py

```
usage: GenericAutomatizedPerturbation.py values [-h] [--query_field QUERY_FIELD] [--docs_field DOCS_FIELD] [--subdataset SUBDATASET] [--gen_mean] [--gen_std]
                                                [--output OUTPUT]
                                                model dataset {irdatasets,file} perturbations

positional arguments:
  model                Nom du modèle (chemin dans l'arborescence ou identifiant HuggingFace)
  dataset              Nom du dataset dans irdataset ou chemin dans l'arborescence
  {irdatasets,file}    Origine du dataset. Indique comment obtenir le dataset.
  perturbations        Chemin vers le fichier descriptif des perturbations

options:
  -h, --help            show this help message and exit
  --query_field QUERY_FIELD
                        champ du fichier des paires correspondant aux questions
  --docs_field DOCS_FIELD
                        Champ du fichier des paires correspondant aux documents
  --subdataset SUBDATASET
                        Critère de sélection d'une sous-partie du dataset
  --gen_mean            Déclenche le calcul des matrices de moyennes de sensibilité
  --gen_std            Déclenche le calcul des matrices d'écarts-types de sensibilité
  --output OUTPUT, -o OUTPUT
                        Spécifie le nom du fichier de sauvegarde des valeurs (Par défaut, le dossier des perturbations est utilisé et le fichier est nommé
                        perts_values.json)
```

Outils pour l'expérience de contrôle

- TransformerLens : Traite les Encodeurs-Décodeurs
- MechIR : Réimplémente une bonne partie pour le format Encodeur-only

=> Lecture et prise en main des **outils internes à MechIR**

- Fonction de test d'un noeud en cours de création
- Protocole à rédiger

Quantifier l'intensité de perturbation de l'espace de représentation

```
test_pos = read_json("res/mE5-MainV2/exp7/exp6/pos/perts_values.json")
test_neg = read_json("res/mE5-MainV2/exp7/exp6/neg/perts_values.json")

pert_idx = 0

perturbation_delta_neg =
    np.asarray(test_neg[pert_idx]["results"]["baseline_perf"])
    -np.asarray(test_neg[pert_idx]["results"]["perturbed_perf"])

perturbation_delta_pos =
    np.asarray(test_pos[pert_idx]["results"]["baseline_perf"])
    -np.asarray(test_pos[pert_idx]["results"]["perturbed_perf"])

X = np.linspace(0,28,28)
plt.scatter(X[:14], perturbation_delta_neg, label="Paires négatives", color="b")
plt.scatter(X[14:], perturbation_delta_pos, label="Paires positives", color="r")
plt.show()
```

Différence des scores baseline et perturbés (paires positives en rouge et paires négatives en bleu)

