



UNIVERSITE PARIS CITÉ

MASTER VISION ET MACHINE INTELLIGENTE

---

# Explicabilité des modèles d'IA multimodaux

---

Stage de Master 2

Réalisé par :

**Sammy Rabhine**

Encadré par :

Laurent BOZZI, EDF R&D  
Alice DUQUENNE, EDF R&D

Référent :

Laurent WENDLING, LIPADE, Université Paris Cité

5 juillet 2024

# Table des matières

<b>Table des matières</b>	<b>1</b>
<b>Synthèse</b>	<b>3</b>
<b>Executive Synthese</b>	<b>4</b>
<b>Mots clés</b>	<b>5</b>
<b>1 Introduction</b>	<b>6</b>
<b>2 Modèles sur données structurées tabulaires et non structurées textuelles</b>	<b>7</b>
2.1 Traitement des données structurées . . . . .	7
2.2 Traitement des données textuelles . . . . .	7
2.2.1 Feature extraction : word2vec . . . . .	8
2.2.2 BERT . . . . .	8
2.3 Traitement des données multimodales . . . . .	9
<b>3 Méthodes d'interprétabilité</b>	<b>10</b>
3.1 L'interprétabilité des algorithmes . . . . .	10
3.1.1 Définitions . . . . .	10
3.1.2 Taxonomie . . . . .	11
3.2 LIME : Local Interpretable Model-Agnostic Explanations . . . . .	11
3.3 SHAP . . . . .	12
3.4 Les différentes implémentations de SHAP . . . . .	13
3.5 Comparaison des attributions . . . . .	13
3.6 Avancées et perspectives . . . . .	14
<b>4 Revue de littérature</b>	<b>15</b>
<b>5 Natural Language Explanations</b>	<b>17</b>
5.1 Vers des explications en langage naturel ? . . . . .	17
5.2 Techniques de rationalisation . . . . .	17
5.2.1 Machine Reading Comprehension . . . . .	17
5.2.2 Commonsense Reasoning . . . . .	19
5.2.3 Natural Language Inference . . . . .	20
5.2.4 Text Classification . . . . .	20
5.3 Challenges . . . . .	22
5.3.1 Evaluations statistiques . . . . .	22
5.3.2 Métriques . . . . .	22
5.3.3 Data . . . . .	24
<b>6 Towards Explainable NLP : A Generative Explanation Framework for Text Classification</b>	<b>24</b>
6.1 Objectifs . . . . .	24
6.2 Données . . . . .	24
6.3 Méthodologie . . . . .	26
6.4 Generative Explanation Framework . . . . .	26
6.5 Explanation Factor . . . . .	27
6.6 Minimum Risk Training . . . . .	27
<b>7 Mise en pratique</b>	<b>28</b>
7.1 Jeu de données Open Data utilisé . . . . .	28
7.1.1 Prétraitements des données . . . . .	28
7.2 VAE . . . . .	28
7.2.1 Architecture . . . . .	29
7.2.2 Encoder . . . . .	29
7.2.3 Echantillonnage . . . . .	29
7.2.4 Décodeur . . . . .	30

7.2.5	Fonction de perte . . . . .	30
7.2.6	Entraînement du VAE . . . . .	30
7.2.7	Implémentation . . . . .	30
7.2.8	Résultats . . . . .	31
7.3	CVAE . . . . .	32
7.3.1	Architecture . . . . .	32
7.3.2	Encoder . . . . .	32
7.3.3	L'échantillonnage . . . . .	33
7.3.4	Décodeur . . . . .	33
7.3.5	Fonction de perte . . . . .	33
7.3.6	Entraînement du CVAE . . . . .	33
7.3.7	Implémentation . . . . .	33
7.3.8	Résultats . . . . .	35
7.4	GEF . . . . .	35
7.4.1	Architecture (rajout du pretrained classifieur) . . . . .	35
7.4.2	Implémentation du classifieur . . . . .	36
7.4.3	Résultats . . . . .	37
<b>8</b>	<b>Limites</b>	<b>38</b>
<b>9</b>	<b>Bilan et perspectives</b>	<b>39</b>
9.1	Etat de l'art . . . . .	39
9.2	Perspectives . . . . .	39
<b>10</b>	<b>Annexe</b>	<b>41</b>
10.1	Few-Shot Self-Rationalization with Natural Language Prompts . . . . .	41
	<b>Table des figures</b>	<b>42</b>
	<b>Liste des tableaux</b>	<b>42</b>

# Synthèse

Le stage s'est déroulé dans un groupe de Data Scientists de la R&D d'EDF, installé sur le site de Paris-Saclay, situé sur le plateau éponyme. Ce groupe, nommé E7C, est spécialisé en analyse de données de consommation, et en modélisation statistique dans le but de concevoir de nouvelles offres de fourniture ou des services.

Les travaux d'Explicabilité des modèles d'IA multimodaux s'inscrivent dans le projet ACACIA (Analyse Connaissance client, Algos pour Commerce & IA) qui a deux enjeux majeurs :

- Améliorer la connaissance client, méthodologique et IA pour la construction de scores pour les clients particuliers et prospects ;
- Innover et anticiper les besoins métiers par des méthodes et techniques fiables, robustes, explicables, et respectueuses des contraintes juridiques.

De nombreuses données sont générées, recueillies et traitées par EDF. Ces données peuvent avoir plusieurs formes : des tableaux de valeurs, des séries temporelles, des textes, des images ou encore des vidéos. Pour un même contexte, plusieurs modes de données (appelés modalités) peuvent être produits. Cependant, la valorisation de ces données au moyen d'algorithmes se fait souvent en ne considérant qu'un type de données à la fois. Afin de profiter de la complémentarité que peuvent avoir ces différents modes, il convient de développer des algorithmes dits multimodaux.

La première difficulté consiste à déterminer l'architecture à adopter pour ces algorithmes. Cette architecture dépend principalement du type de données, des algorithmes d'extraction de données que l'on souhaite utiliser, de la puissance et du temps de calcul disponibles. Une taxonomie de ces architectures existe dans la littérature scientifique.

La deuxième difficulté consiste en l'application des méthodes d'explicabilité. Ces méthodes sont le plus souvent développées pour s'appliquer aux algorithmes d'un mode en particulier, le plus souvent des données tabulaires. Les représentations proposées sont toujours unimodales. Très peu d'outils sont développés pour les cas multimodaux.

EDF Recherche et Développement a lancé en 2022 des travaux méthodologiques sur la modélisation multimodale (sur données tabulaires et textuelles) et son explicabilité. Un précédent stage a mis l'accent sur les méthodes de fusion entre la partie tabulaire et la partie textuelle dans la modélisation, et proposé une première méthode d'explicabilité multimodale. Cette dernière a cependant montré ses limites sur la prise en compte de l'effet des éléments de texte dans l'explicabilité.

Ce stage prend la suite des travaux passés avec un objectif de complétion de l'état de l'art sur les techniques d'explicabilité, avec un focus plus important sur la prise en compte du texte. Une démarche de recherche a été mise en œuvre et nous a menés jusqu'à la Self-Rationalization, et l'ambition de se diriger vers une explication en langage naturel des modèles d'IA multimodaux.

Une première étape de reprise de l'existant et de recherche bibliographique complémentaire a été effectuée. Une seconde phase d'analyse et lecture critique et approfondie de deux articles d'intérêt a été menée. Ces deux phases ont concentré la majeure partie du stage. Une dernière étape a été le recodage de A à Z d'un article, avec application sur un jeu de données open data.

Le cas d'application a concerné la classification de produits high-tech selon si leur notation était positive, neutre ou négative, avec des tests de génération d'une explication en langage naturel.

Il s'agissait principalement d'un stage de recherche où nous avons testé plusieurs cas de figure notamment la génération de langages avec un VAE pour commencer, puis un CVAE, et par la suite nous avons intégré un pre trained classifier pour la classification. Nous allons montrer tous les étapes effectuées et certaines limites du modèle final proposé.

# Executive Synthese

The internship took place within a group of Data Scientists at EDF's R & D department, located on the Paris-Saclay site, situated on the eponymous plateau. This group, named E7C, specializes in analyzing consumption data and statistical modeling to design new supply offers or services.

The "Explicability of Multimodal AI Models" project is part of the ACACIA project (Analysis Knowledge customer, Algos for Commerce & AI), which has two major objectives :

- Improve customer knowledge, methodologies, and AI for building scores for individual customers and prospects
- Innovate and anticipate business needs through reliable, robust, explainable, and legally compliant methods and techniques

EDF generates, collects, and processes a large amount of data. This data can take several forms : tables of values, time series, texts, images, or videos. For a given context, several data modes (called modalities) can be produced. However, the valorization of this data through algorithms often only considers one type of data at a time. To take advantage of the complementarity that these different modes can have, it is necessary to develop so-called multimodal algorithms.

The first challenge is to determine the architecture to adopt for these algorithms. This architecture mainly depends on the type of data, the data extraction algorithms that we wish to use, the computing power and time available. A taxonomy of these architectures exists in the scientific literature.

The second challenge lies in the application of explainability methods. These methods are most often developed to apply to algorithms of a particular mode, most often tabular data. The proposed representations are always unimodal. Very few tools are developed for multimodal cases.

EDF Research and Development launched methodological work in 2022 on multimodal modelisation (on tabular and textual data) and its explainability. A previous internship focused on the fusion methods between the tabular part and the textual part in the modeling, and proposed a first method of multimodal explainability. However, this method showed its limits on taking into account the effect of text elements in the explainability.

This internship follows on from past work with the objective of completing the state of the art on explainability techniques, with a greater focus on taking into account the text. A research approach was implemented and led us to Self-Rationalization, with the ambition of moving towards a natural language explanation of multimodal AI models.

A first step was to familiarize myself with the existing work. An additional bibliographic research was carried out then. A second phase of analysis and in-depth critical reading of two articles of interest was conducted. These two phases concentrated the majority of the internship. A final step was the recoding from A to Z of an article, with application on an open data dataset.

The application case concerned the classification of high-tech products according to whether their rating was positive, neutral or negative, with tests for generating an explanation in natural language.

It was mainly a research internship where we tested several scenarios, notably language generation with a VAE to start with, then a CVAE, and subsequently, we integrated a pre-trained classifier for classification. We will show all the steps taken and certain limitations of the final proposed model.

## Mots clés

Explicabilité

XAI

Interprétabilité

Multimodal

Données structurées

Données non-structurées

Deep Learning

VAE : Variational Auto Encoder

CVAE : Conditional Variational Auto Encoder

CNN : Convolutional Neural Network

MRC : Machine Reading Comprehension

NLI : Natural Language Inference

Self-rationalization

Natural language Explanation : NLE

Generative Explanation Framework : GEF

Large language Model : LLM

# 1 Introduction

Le stage s'est déroulé dans un groupe de Data Scientists de la R&D d'EDF, installé sur le site de Paris-Saclay, situé sur le plateau éponyme. Ce groupe, nommé E7C, est spécialisé en analyse de données de consommation, et en modélisation statistique dans le but de concevoir de nouvelles offres de fourniture ou des services.

Le projet d'Explicabilité des modèles d'IA multimodaux s'inscrit dans le projet ACACIA (Analyse Connaissance client, Algos pour Commerce & IA) qui a deux enjeux majeurs :

- Améliorer la connaissance client, méthodologique et IA pour la construction de scores pour les clients particuliers et prospects ;
- Innover et anticiper les besoins métiers par des méthodes et techniques fiables, robustes, explicables, et respectueuses des contraintes juridiques.

De nombreuses données sont générées, recueillies et traitées par EDF. Ces données peuvent être de plusieurs formes telles que les séries temporelles, textes, images, audios, vidéos. . . Or ces données sont généralement entraînées de manière unimodale c'est-à-dire qu'on utilise qu'une seule entrée. Ces dernières années la littérature a mis l'accent notamment sur les données de type vision-langage model tel que le VQA (Vision Question Answering) par exemple. Nous avons pu constater des progrès phénoménaux dans ces domaines en combinant plusieurs types de données. Bien que de nombreux travaux aient été réalisés sur l'explicabilité des modèles d'IA, peu d'entre eux se sont concentrés sur le cas multimodal impliquant des données tabulaires et textuelles. Cette lacune est problématique car la compréhension de ces modèles est essentielle pour garantir la transparence et la conformité avec les réglementations telles que le RGPD et l'AI Act prévu pour 2026.

EDF R&D a donc lancé des travaux de recherche sur l'explicabilité des modèles d'IA multimodaux. Cependant, cette tâche s'avère complexe en raison des défis supplémentaires liés à l'utilisation de données multimodales. Il est donc nécessaire de développer des méthodes d'explication adaptées à ce type de modèles pour répondre aux exigences réglementaires et garantir une utilisation éthique et transparente de l'IA.

Trois problématiques, qui guident le présent travail, se sont posées lors de ces modélisations :

1. Quels sont les influences de chaque modalité sur les modèles ?
2. Comment développer des algorithmes capables de prédire et de justifier leurs décisions à partir de différents types de données ?
3. Comment expliquer les résultats obtenus ?

Le stage s'est tout d'abord concentré sur une revue de la littérature scientifique concernant trois sujets spécifiques, à savoir l'utilisation de données textuelles et tabulaires pour la classification et la justification de celle-ci. Notre objectif principal était de compléter l'état de l'art en suivant une démarche de recherche rigoureuse.

Dans les trois premières parties du rapport, nous présentons une synthèse des notions et méthodes rencontrées lors de notre revue de la littérature scientifique (parties 2 à 4).

Cette dernière nous a amenés à explorer la génération d'explications en langage naturel (partie 5). Nous nous sommes alors concentrés sur un article en particulier pour approfondir notre compréhension de la méthode proposée (partie 6).

Nous avons ensuite recodé cette méthode de A à Z pour la mettre en application, en utilisant des jeux de données libres d'accès (partie 7). L'une des principales difficultés rencontrées a été de développer un modèle suffisamment performant pour réaliser une classification multi-labels et justifier cette classification par la génération de texte.

Certaines méthodes étudiées n'ont pas été mises en pratique, mais nous les avons incluses en bibliographie et annexe pour référence future.

## 2 Modèles sur données structurées tabulaires et non structurées textuelles

Le terme de modalité est habituellement utilisé pour désigner les différentes valeurs que prennent les variables qualitatives. Ce terme est aussi utilisé pour définir les différents modes de données. Parmi les modes de données traités par ordinateur, il est possible de distinguer deux catégories :

- a) **Les données structurées** : ce sont des données telles que l'on peut les faire figurer dans une base de données. Ce type de données est le plus adapté aux méthodes d'apprentissage automatique. Les données tabulaires sont un exemple de données structurées.
- b) **Les données non structurées** : ce sont toutes les autres données, que l'on ne peut pas aisément mettre dans une base de données. Ces données sont plutôt destinées à être consultées par des humains que des algorithmes. Les images et les textes en sont des exemples.



FIGURE 1 – Illustration des principales modalités.

### 2.1 Traitement des données structurées

Les techniques d'apprentissage automatique ont d'abord été développées pour le traitement des données structurées. Pour le traitement de ces données, les méthodes les plus performantes sont généralement des méthodes basées sur des arbres de décision (forêts aléatoires, gradient boosting, ...) [13]

Si l'écart reste significatif entre les algorithmes basés sur des arbres de décision et les algorithmes de *deep-learning*, dans l'objectif de construire un modèle multimodal explicable, il est plus confortable de travailler en utilisant exclusivement les algorithmes de *deep-learning*.

### 2.2 Traitement des données textuelles

Lorsque l'on utilise des données non structurées, des méthodes propres à la modalité employée permettent une transformation en données structurées. Dans le cas étudié, seules des données textuelles seront utilisées.

Le traitement du langage se heurte à plusieurs difficultés, parmi lesquelles :

- a) l'ambiguïté linguistique : les mots et groupes de mots peuvent avoir des sens différents en fonction de leur contexte ;
- b) la taille des données : les données d'entrée sont de tailles variables ;
- c) le traitement des mots inconnus.

La chaîne de traitement de langage peut être schématisée comme suit :

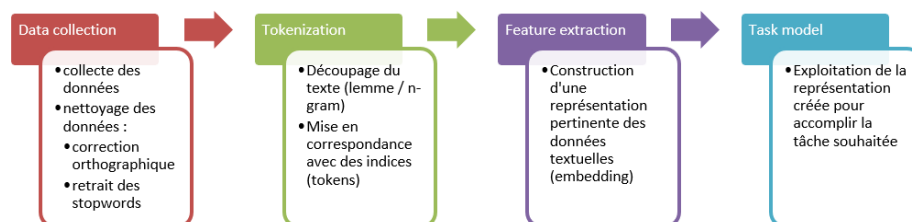


FIGURE 2 – Schéma des étapes de traitement automatisé du langage.



Cette chaîne propose une étape dite de *feature extraction*. Cette étape consiste à parvenir à construire un vecteur représentatif du texte étudié. Ce vecteur permet par la suite de réaliser la tâche voulue en utilisant les algorithmes d'apprentissage automatique existant.

Dans le processus de traitement de langage, deux moments semblent compatibles pour une fusion avec un tableau de valeur : une fois le vecteur (*embedding*) produit par l'étape de *feature extraction*, ou la sortie du *task model*.

### 2.2.1 Feature extraction : word2vec

Les algorithmes « word2vec » [26] sont des algorithmes de traitement du langage dont l'objectif est de représenter les mots sous la forme de vecteurs numériques (*word embedding*). Il s'agit d'une façon particulièrement efficace de réaliser l'étape de *feature extraction*.

Deux principales approches sont couramment utilisées pour entraîner ces algorithmes :

- a) « *bag of words* » : l'algorithme est entraîné à prédire un mot à partir du contexte (les mots qui l'entourent). L'objectif est de faire apprendre à l'algorithme à associer les mots d'un contexte à un mot cible ;
- b) « *skip-gram* » [25] : la tâche est inversée, l'algorithme prend en entrée un mot, il doit prédire le contexte.

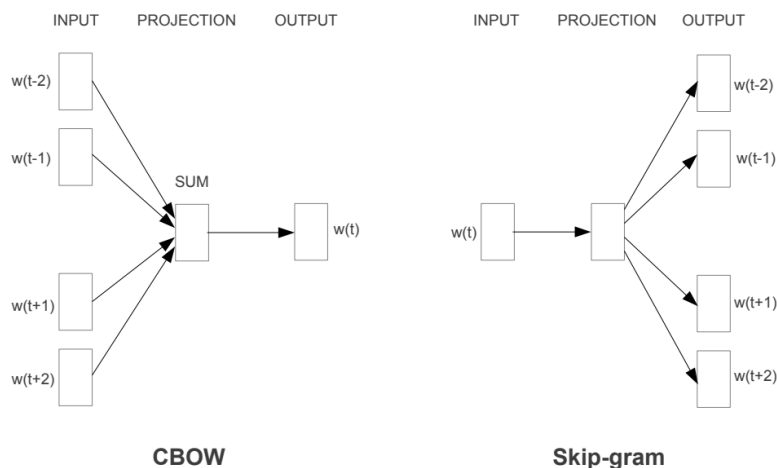


FIGURE 3 – Architecture de l'entraînement des modèles « continuous bag of words » (CBOW) et « skip-gram ».

Dans les deux cas, l'algorithme word2vec consiste en un réseau de neurones. Lors de l'entraînement, l'algorithme parcourt chaque fenêtre de mots, un mot central et ses voisins, et optimise ses paramètres pour maximiser la probabilité de prédire les mots correctement. Une fois que l'algorithme est suffisamment entraîné, il est possible de récupérer un vecteur de nombres réels (parmi les états intermédiaires du réseau de neurones) associé à chaque mot. Ce vecteur est appelé « *embedding* ».

Ces deux méthodes permettent d'associer un mot à un *embedding*. Cet *embedding* peut être utilisé directement pour des tâches telles que : mesurer la similarité entre les mots ou établir des clusters de mots.

Pour des tâches plus complexes, telles que l'analyse de sentiments (tâche de classification), un réseau de neurone (souvent récurrent) est employé pour prendre en compte les interactions des mots entre eux.

### 2.2.2 BERT

L'introduction des *Transformers* [42] a permis une avancée significative dans le domaine du traitement automatique du langage. En particulier, ils permettent d'éviter l'utilisation de réseaux récurrents, et améliorent grandement l'efficacité des modèles (meilleure parallélisation). Ils sont basés sur le mécanisme d'attention, ce qui permet de prendre en compte l'intégralité des interactions entre les mots.

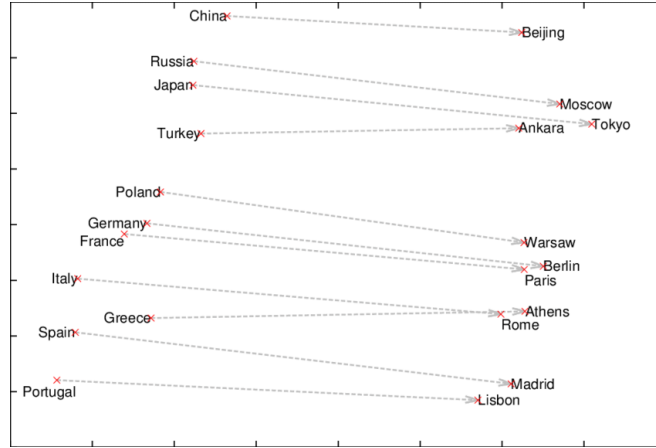


FIGURE 4 – Exemple de projection d’une représentation word2vec.

Les modèles de traitement du langage choisis sont basés sur le modèle BERT (*Bidirectional Encoder Representations from Transformers*) [10].

L’architecture de ce type de modèle peut être découpé en 3 parties :

- un *tokenizer* : sa fonction est de découper le texte d’entrée en lemmes (groupes de lettres de petite taille) qui sont identifiés à une clé numérique unique. Le tokenizer est un dictionnaire, il n’a pas besoin d’être entraîné ;
- le modèle BERT, qui contient les couches de Transformers et renvoie les *embeddings* de chaque *token*. Il s’agit de la partie du modèle utilisée pré-entraînée ;
- le modèle spécifique, qui exploite les *embeddings* pour réaliser la tâche voulue.

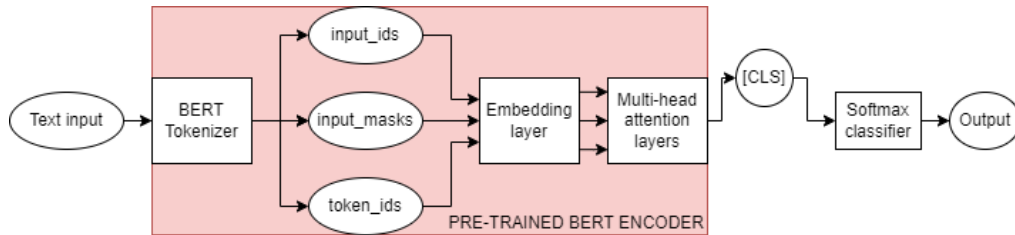


FIGURE 5 – Graphique synthétique d’un modèle d’analyse de sentiment basé sur BERT.

Le modèle est construit de façon à découper son entraînement en deux étapes : le pré-entraînement et la spécialisation (*fine-tuning*). Le pré-entraînement consiste à faire prédire au modèle des mots masqués aléatoirement dans une phrase. Cette tâche permet un apprentissage non-supervisé, sur de grands jeux de données. Le modèle obtenu à ce stade correspond au sous réseau encadré en rouge dans la Figure 5.

Dans le cas de l’analyse de sentiments, la spécialisation consiste à utiliser le modèle pré-entraîné pour récupérer le *token* de classification ([CLS]). Ce token est placé au début de chaque phrase par le *tokenizer*. Son état final sert de vecteur d’entrée pour le classifieur.

Cette architecture de modèle permet d’utiliser des modèles pré-entraînés sur de grands volumes de données, diffusés sur internet, puis de ne réaliser que l’étape de spécialisation sur un jeu de données plus restreint. Le modèle *bert-base-uncased* [10] a été choisi. Il présente l’avantage d’avoir un modèle pré-entraîné en anglais, et plus simple à finetuner (plus petit vocabulaire que la version *cased*).

La version *bert-base-cased* n’a pas été retenue du fait qu’elle utilise une taille de vocabulaire beaucoup plus grande, nous étions limités sur les ressources computationnelles.

## 2.3 Traitement des données multimodales

Le traitement des données multimodales consiste à combiner des données de différentes modalités, telles que des données structurées (tabulaires) et des données non structurées (textuelles), pour améliorer

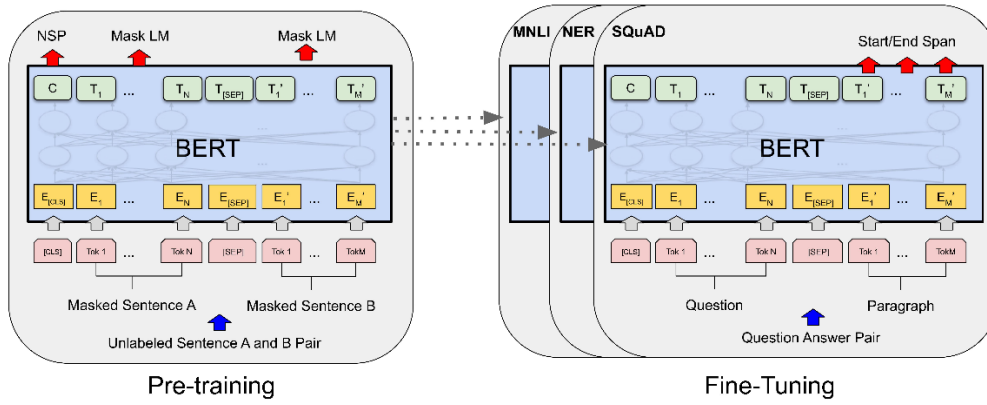


FIGURE 6 – Pré-entraînement et entraînement du modèle BERT.

les performances d'un modèle d'apprentissage automatique. Dans cette approche, les informations complémentaires fournies par les différentes modalités peuvent aider à mieux comprendre les données et à prendre de meilleures décisions.

Pour traiter des données multimodales, il est nécessaire de fusionner les informations provenant des différentes modalités. Il existe plusieurs méthodes pour réaliser cette fusion :

- Fusion précoce** : Cette méthode consiste à concaténer les caractéristiques (features) des différentes modalités en une seule représentation avant de l'utiliser comme entrée d'un modèle d'apprentissage automatique. Par exemple, pour combiner des données tabulaires et textuelles, on peut concaténer les vecteurs d'embedding des mots (obtenus à partir d'un modèle word2vec) avec les caractéristiques tabulaires.
- Fusion tardive** : Dans cette approche, les différentes modalités sont traitées séparément par des modèles d'apprentissage automatique distincts, et les résultats de ces modèles sont combinés pour prendre une décision finale. Par exemple, on peut utiliser un perceptron multicouche (MLP) pour traiter les données tabulaires et un réseau de neurones récurrent (RNN) pour traiter les données textuelles. Les sorties de ces deux modèles sont ensuite combinées à l'aide d'une fonction de fusion (par exemple, une moyenne pondérée) pour obtenir la prédiction finale.
- Fusion intermédiaire** : Cette méthode consiste à combiner les représentations intermédiaires des différentes modalités à un certain niveau dans le modèle d'apprentissage automatique. Par exemple, on peut utiliser un modèle de deep learning qui traite conjointement les données tabulaires et textuelles en combinant les couches intermédiaires des deux sous-modèles correspondant à chaque modalité.

### 3 Méthodes d'interprétabilité

Cette partie introduit d'abord des définitions, puis présente les méthodes qui ont semblé les plus pertinentes à approfondir dans le cadre d'algorithmes multimodaux.

#### 3.1 L'interprétabilité des algorithmes

##### 3.1.1 Définitions

Certaines notions du domaine de l'explicabilité des IA (XAI) restent subjectives. Par exemple la définition de ce qu'est une « bonne explication ». Il n'existe pas de consensus dans les publications du domaine sur la définition, en particulier les définitions ou distinctions entre les notions d'interprétabilité et d'explicabilité. Les définitions retenues sont celles données par Tim Miller (Miller, 2019) [27] et rappelées dans le livre « *Interpretable Machine Learning* » de Christoph Molnar (Molnar, 2020)[29].

L'**interprétabilité** d'un modèle est définie comme « le degré auquel un observateur peut comprendre la cause d'une décision ». On peut ainsi ordonner le niveau d'interprétabilité des modèles en fonction de la facilité de compréhension par un humain des décisions prises par ce modèle. Le terme d'**explicabilité** est ici confondu avec celui d'interprétabilité. Le terme « **explication** » est utilisé pour désigner l'étude de la prédiction pour un individu en particulier.

### 3.1.2 Taxonomie

Les méthodes d'interprétabilité peuvent être distinguées de plusieurs façons. Les termes d'intrinsèque et de post-hoc distinguent les méthodes qui consistent à restreindre la complexité d'un modèle pendant sa phase d'entraînement (**intrinsèques**) des méthodes appliquées après l'entraînement du modèle (**post-hoc**).

Les méthodes intrinsèques consistent à contraindre la complexité d'un modèle pour en maintenir un niveau d'explicabilité acceptable. Par exemple, dans le cas d'un arbre de décision, la limitation de la profondeur est une méthode intrinsèque d'explicabilité. Ces méthodes sont propres à certains types de modèles, que l'on qualifiera souvent de « intrinsèquement interprétable ». C'est le cas de la régression linéaire, de la régression logistique, des arbres de décision (de faible profondeur) ou encore des k-plus proches voisins.

Les méthodes post-hoc sont applicables à un modèle déjà entraîné. Elles peuvent également s'appliquer à des modèles intrinsèquement interprétables. Ces méthodes considèrent le plus souvent le modèle comme une boîte noire. Il est alors question de méthodes dites « **agnostiques** au modèle ».

Enfin, les méthodes d'interprétabilité globales et locales sont distinguées. Une méthode fournissant des explications pour un individu particulier, ou un groupe d'individus, est dite « **locale** ». Les méthodes fournissant des éléments d'explicabilité indépendants des individus est une méthode **globale**. Il est souvent compliqué d'obtenir l'explicabilité globale d'un modèle.

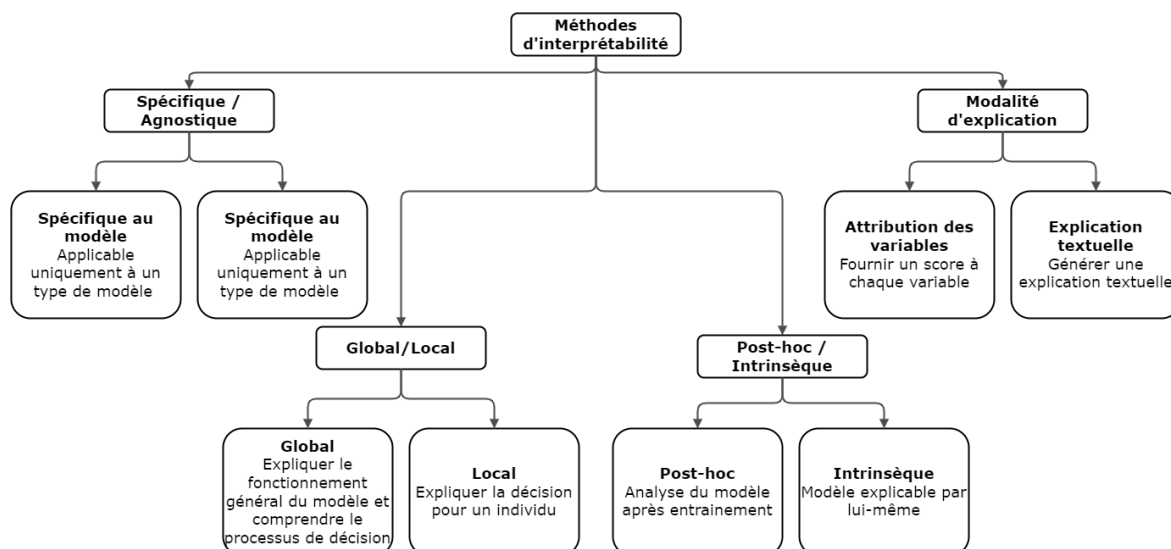


FIGURE 7 – Exemples de taxonomies des méthodes d'interprétabilité

Parmi les méthodes d'explicabilité les plus populaires, on retrouve LIME (Local Interpretable Model-agnostic Explanations) et SHAP (SHapley Additive exPlanations). Ces méthodes permettent de fournir des explications locales pour les prédictions d'un modèle, en se basant sur les contributions des différentes variables d'entrée. Il existe par ailleurs différentes façons de représenter ces explications, selon la forme des données utilisées pour l'interprétabilité. Voici quelques exemples :

- les paramètres du modèle dans le cas des modèles intrinsèquement interprétables ;
- un individu particulier (existant ou synthétique) dans le cas des méthodes de *counterfactual explanation* ou d'*adversarial examples* ;
- des données intermédiaires du modèle. Par exemple dans le traitement d'images avec des réseaux de neurones, la visualisation de sorties intermédiaires des réseaux de convolutions ;
- des statistiques sur les variables d'entrées : l'importance qu'elles ont ou encore la mesure de l'interaction entre les variables par exemple. Ces statistiques peuvent éventuellement être présentées sous la forme de graphiques.

## 3.2 LIME : Local Interpretable Model-Agnostic Explanations

LIME (Local Interpretable Model-Agnostic Explanations) [36] consiste à créer un modèle local interprétable qui approxime le modèle à expliquer en autour d'un individu spécifique. Le modèle local est créé en

échantillonnant des perturbations autour du point d'entrée, puis en ajustant un modèle linéaire à la relation entre les perturbations et les prédictions du modèle. Les poids du modèle linéaire sont utilisés pour expliquer la prédiction du modèle pour l'individu initial.

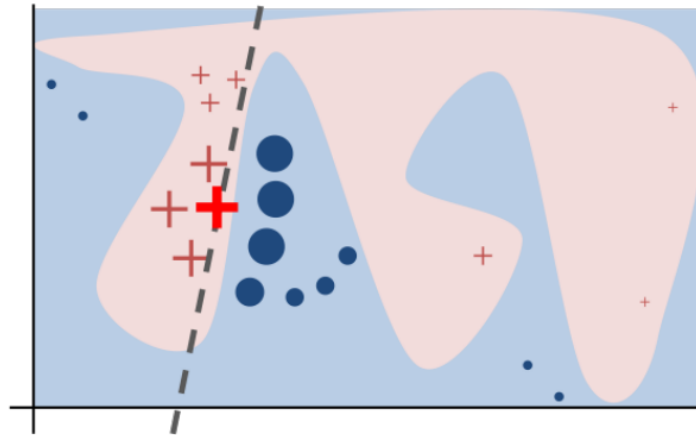


FIGURE 8 – Illustration de LIME.

LIME peut être utilisé sur toutes les modalités. En effet, cette méthode repose sur la capacité à générer des perturbations autour d'un point à étudier. Dans le cas de données tabulaires, il suffit de modifier les valeurs du vecteur représentant le point à étudier pour obtenir les points perturbés.

Pour des données textuelles, de nouveaux textes sont générés en retirant des mots aléatoirement de l'entrée initiale. Une entrée est alors représentée comme un vecteur composé de 0 et de 1, dont les variables représentent la présence ou non du mot dans l'entrée. Cette représentation permet de revenir à un cas tabulaire. Les poids obtenus représentent alors l'importance d'un mot dans la prédiction réalisée.

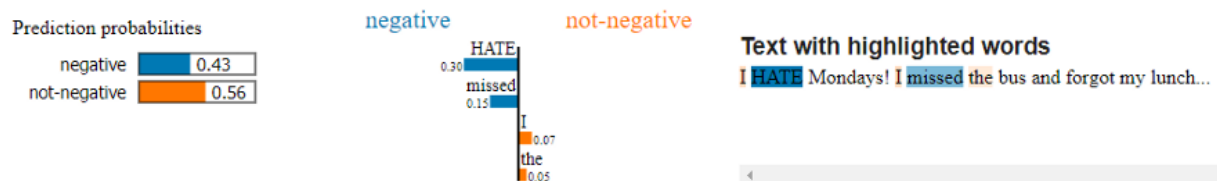


FIGURE 9 – Exemple de l'utilisation de LIME pour de l'analyse de sentiments.

LIME possède les avantages suivants :

- a) c'est une méthode agnostique ;
- b) c'est une méthode locale, elle est pertinente pour expliquer la décision d'un algorithme sur un individu spécifique ;
- c) la méthode s'applique à toutes les modalités.

Elle possède les inconvénients suivants :

- a) la génération du voisinage du point à étudier dans le cas tabulaire n'est pas clairement définie, ce qui peut mener à des instabilités dans les explications qui en découlent ;
- b) dans certains cas, l'explication fournie pour deux individus proches peut être très différente ;
- c) la génération d'explications pour de grands ensembles de données peut s'avérer coûteuse.

La librairie *lime* est utilisable, mais n'est plus maintenue depuis 2021. A l'exception de *Captum*, l'ensemble des librairies trouvées proposant *LIME* dépendent de cette librairie.

### 3.3 SHAP

Les valeurs de Shapley (Shapley, 1953) [39] proviennent de la théorie des jeux et visent à distribuer équitablement le gain d'une prédiction entre les variables d'un modèle. SHAP (SHapley Additive exPlanations) utilise ces valeurs pour expliquer les prédictions des modèles en prenant en compte les interactions

entre les variables (Lundberg & Lee, 2017) [21]. Bien que SHAP offre une base théorique solide et de nombreuses représentations graphiques utiles, son calcul est souvent coûteux, surtout avec KernelSHAP, qui nécessite une limitation du nombre d'instances expliquées.

### 3.4 Les différentes implémentations de SHAP

- KernelSHAP est une approximation basée sur le modèle linéaire ajusté pour relier les coalitions de variables aux prédictions. Ce modèle est utilisé pour estimer les valeurs SHAP en considérant uniquement deux affectations pour chaque variable : celle de l'individu à expliquer et la valeur nulle.
- TreeSHAP, adapté aux modèles basés sur des arbres de décision, permet un calcul plus efficace des valeurs SHAP en exploitant la structure des arbres pour réduire le coût computationnel.

#### Limitations de SHAP :

- KernelSHAP : Nécessite de limiter le nombre d'instances à expliquer et ignore les dépendances entre variables, menant à des résultats biaisés.
- TreeSHAP : Peut fournir des coefficients non intuitifs et des résultats biaisés en raison des dépendances non prises en compte.
- DeepExplainer : Adapté aux réseaux de neurones mais peu efficace avec les modèles PyTorch, en particulier avec des données de type *torch.Tensor*, et avec les modèles nécessitant un GPU.

### 3.5 Comparaison des attributions

La comparaison des algorithmes d'attribution est complexe. Il n'existe pas de méthode standardisée pour comparer les vecteurs d'attributions, rendant difficile l'évaluation de l'erreur entre l'algorithme d'attribution et l'algorithme lui-même. La distance de Wasserstein est une méthode utilisée pour mesurer cette distance, implémentée dans la librairie *scipy*.

#### Représentations graphiques avec SHAP :

1. Graphique en barres *waterfall* : Représente les importances globales des variables en calculant la moyenne des valeurs absolues des valeurs SHAP, permettant de visualiser l'ordre d'importance des variables sans indiquer le sens de l'influence.

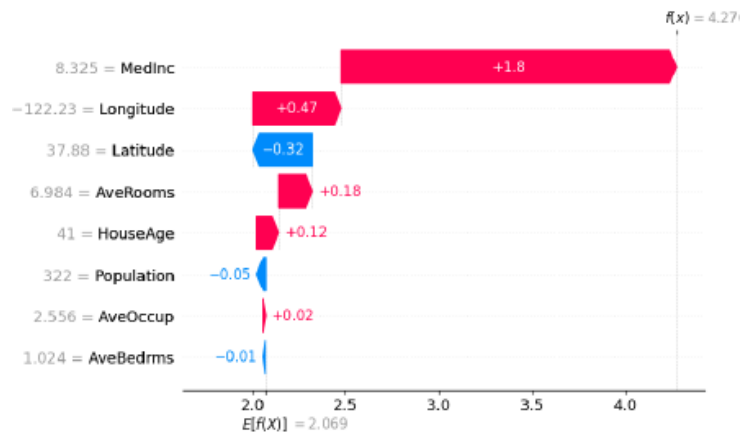


FIGURE 10 – Illustration du graphique en barres.

2. *Beeswarm Plot* : Ce graphique montre la distribution des valeurs SHAP pour chaque variable, permettant de visualiser les corrélations potentielles entre une variable et ses valeurs SHAP.

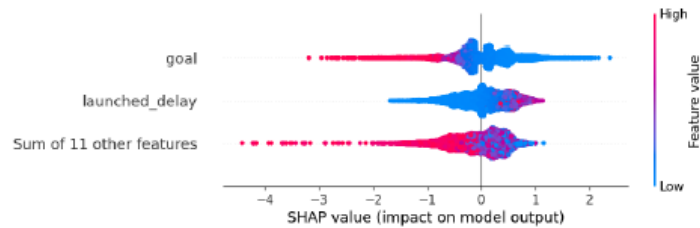


FIGURE 11 – Illustration du Beeswarm Plot.

3. *Graphique de Dépendance (Partial Dependence Plot)* : Trace les valeurs SHAP en fonction des valeurs des variables, souvent enrichi par une échelle de couleur pour une autre variable, utile uniquement pour les données tabulaires.

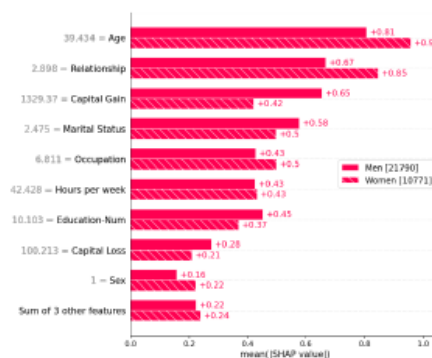


Figure : moyenne des valeurs absolues des valeurs SHAP séparées en deux échantillons.

FIGURE 12 – Illustration du graphique de dépendance.

### 3.6 Avancées et perspectives

Le domaine de l'explicabilité des modèles d'IA est encore en développement, avec des outils actuellement performants, principalement pour les données tabulaires. L'adaptation des méthodes d'explicabilité à d'autres modalités, comme les données textuelles et les images, reste rudimentaire. Les progrès réalisés en traitement automatique du langage (NLP) peuvent potentiellement améliorer l'explicabilité pour les données textuelles. Des stratégies d'agrégation multimodales et la combinaison d'algorithmes de deep learning avec des techniques de boosting de gradient représentent des perspectives prometteuses.

Bien que SHAP soit un outil puissant pour l'explicabilité des modèles, son application est souvent limitée par des coûts computationnels élevés et des adaptations incomplètes pour les données non tabulaires. Les efforts futurs devraient se concentrer sur l'amélioration de ces méthodes pour les scénarios multimodaux, en intégrant de nouvelles techniques pour une explicabilité plus robuste et généralisable.

En complément des méthodes d'attribution des caractéristiques, une nouvelle orientation prometteuse dans le domaine de l'explicabilité est l'utilisation des explications en langage naturel. Ces techniques, connues sous le terme de Natural Language Explanations (NLE), offrent des explications compréhensibles par les humains, surpassant souvent les méthodes traditionnelles basées sur les caractéristiques. En combinant le raisonnement sémantique et les interactions complexes des modèles, les NLE peuvent fournir des explications plus intuitives et détaillées des décisions des modèles. Cet aspect sera approfondi dans la suite du rapport.

## 4 Revue de littérature

L'exploration des techniques d'explicabilité en intelligence artificielle (IA) a été largement dominée par des méthodes basées sur les caractéristiques (feature-based methods) et les méthodes basées sur la perturbation (perturbation-based methods). Ces approches visent à fournir des explications compréhensibles pour les décisions des modèles en identifiant les éléments des données d'entrée les plus influents. Les travaux de Danielvsky et al. (2020) [8] et Ross et al. (2017) [37] illustrent bien ces techniques en se concentrant respectivement sur l'explicabilité des données textuelles et sur la contrainte des explications pour des modèles différentiables. De même, les études de Ramon et al. (2020) [34] et de Shi et al. (2021) [40] mettent en évidence l'importance de l'explicabilité dans des contextes multimodaux, en intégrant diverses formes de données comme le texte et les images. Les trois tableaux suivants présentent la synthèse des articles les plus importants qui ont été étudiés pendant la revue de littérature.







Papier	Résumé	Type de données
Danielvsky, M., Qian, K., Aharonov, R., Katsis, Y., Kawas, B., & Sen, P. (2020). A survey of the state of explainable AI for natural language processing. arXiv preprint arXiv:2010.00711.	L'article se concentre sur les techniques d'explicabilité et d'évaluations sur les données de type texte. Ce qui est intéressant dans leur article est la partie Predictive Process Coverage où l'explication peut se faire étape par étape, comme une démonstration mathématique à creuser.	 Text
Ross, A. S., Hughes, M. C., & Doshi-Velez, F. (2017). Right for the right reasons: Training differentiable models by constraining their explanations. arXiv preprint arXiv:1703.03717.	Les auteurs ont démontré qu'entraîner des modèles avec des inputs gradients pénalisés peuvent rendre possible d'apprendre des décisions généralisables logiques. Leur méthode est plus rapide que LIME et parfois plus précise (surtout avec des variables continues). Pour les variables catégorielles, la durée computationnelle a l'air d'être similaire entre les deux méthodes, mais cela reste à tester.	 Structured data  Images  Text
Ramon, Y., Martens, D., Provost, F., & Evgeniou, T. (2020). A comparison of instance-level counterfactual explanation algorithms for behavioral and textual data: SEDC, LIME-C and SHAP-C. Advances in Data Analysis and Classification, 14, 801-819.	Ramon utilise une nouvelle méthode permettant de générer des counterfactuals explanations (EDC). Elle compare à SHAP-C et LIME-C et obtient des performances très intéressantes.	 Structured data  Text

TABLE 1 – Listes des papiers pertinents sur l'explicabilité




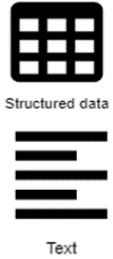
Papier	Résumé	Type de données
Ben-Younes, H., Cadene, R., Thome, N., & Cord, M. (2019, July). Block: Bilinear superdiagonal fusion for visual question answering and visual relationship detection. In Proceedings of the AAAI conference on artificial intelligence (Vol. 33, No. 01, pp. 8102-8109).	Les auteurs ont créé une nouvelle architecture permettant une fusion des modalités optimale tout en surpassant toutes les autres architectures de fusion de l'état de l'art.	
Shi, X., Mueller, J., Erickson, N., Li, M., & Smola, A. J. (2021). Benchmarking multimodal automl for tabular data with text fields. arXiv preprint arXiv:2111.02705.	Papier très complet regroupant différents datasets alliant les données textuelles et tabulaires. Plusieurs méthodes de fusion sont évoquées : early et late fusion (la late fusion étant la méthode finalement retenue par les auteurs). Ces derniers utilisent la librairie autogluon (développée par Amazon).	

TABLE 2 – Liste des papiers pertinents pour la modélisation sur données multimodales


Papier	Résumé	Type de données
Joshi, G., Walambe, R., & Kotecha, K. (2021). A review on explainability in multimodal deep neural nets. IEEE Access, 9, 59800-59821.	<p>Article revenant sur de nombreuses définitions et faisant la taxonomie des modèles multimodaux et des méthodes d'explicabilité. Les données structurées ne sont pas considérées dans cet article. Méthodes d'explicabilité multimodales :</p> <ul style="list-style-type: none"> <li>— Attention based approaches</li> <li>— Counterfactual explanations (comment trouver le changement minimal pour modifier la décision)</li> <li>— Interactive approaches</li> <li>— Graph based approaches</li> <li>— Attribute based methods</li> <li>— Natural Language Explanation</li> </ul>	

TABLE 3 – Papier sur l'explicabilité multimodale, ayant dirigé les travaux vers le NLE

Malgré les avancées significatives apportées par ces méthodes d'explicabilité multimodales (Table 3), elles présentent certaines limitations, notamment en termes de lisibilité et de pertinence des explications pour les utilisateurs finaux. Les méthodes d'explicabilité traditionnelles peuvent parfois manquer de clarté, rendant difficile la compréhension des décisions des modèles pour les non-experts. Les techniques de "Natural Language Explanation" (NLE) promettent de surmonter certaines des limitations des méthodes traditionnelles en offrant des explications en langage naturel plus intuitives et compréhensibles. Par conséquent, mes tuteurs et moi avons décidé de nous concentrer par la suite sur ces techniques innovantes.

## 5 Natural Language Explanations

### 5.1 Vers des explications en langage naturel ?

La majorité des méthodes d'explicabilité se focalisent sur la fourniture d'explications basées sur les caractéristiques d'entrée, en évaluant l'importance ou la contribution de chaque caractéristique après que les modèles aient été entraînés et fixés.

Ces dernières années, des méthodes basées sur l'attribution de caractéristiques se sont développées (feature based methods). Par exemple pour l'analyse d'images, celles-ci mettent en évidence les régions de l'image qui contribuent de manière significative à la prise de décision (GRAD-CAM).

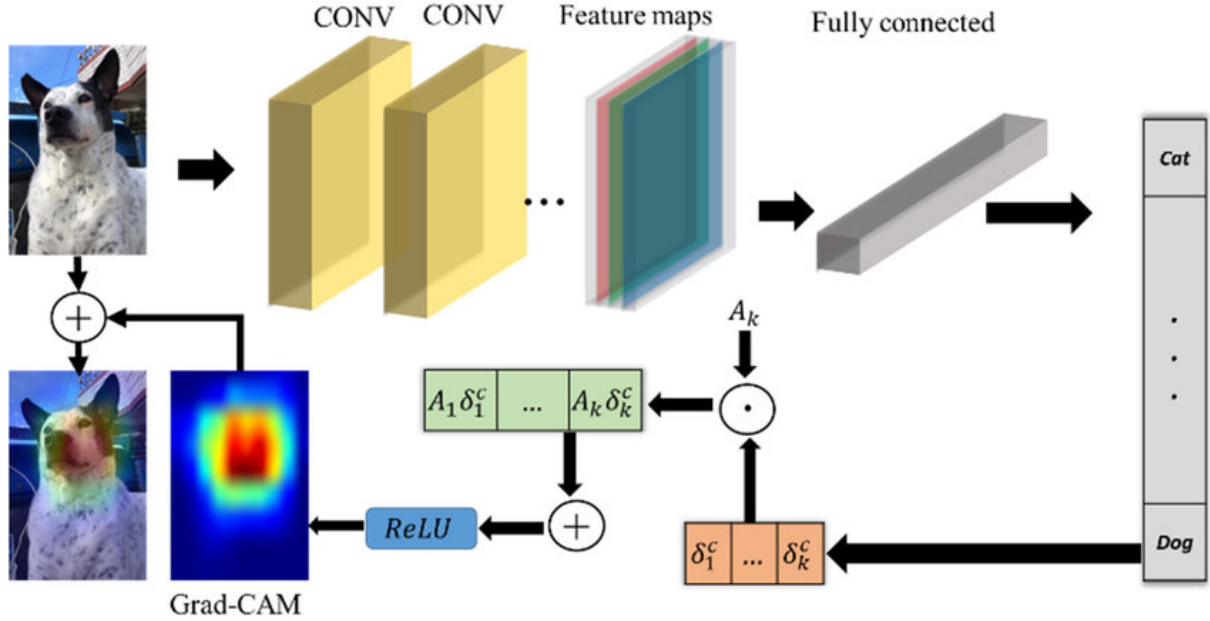


FIGURE 13 – GradCAM architecture

Cependant, le raisonnement sémantique et les interactions ne sont pas pris en compte ici.

Ces méthodes feature-based peuvent notamment :

- Echouer à fournir des explications lisibles par l'homme car les caractéristiques sous-jacentes utilisées par les modèles d'IA peuvent être difficiles à comprendre même pour les utilisateurs experts (par exemple, les tokens pour le texte et les pixels pour les images) ;
- Détecter des cas où le modèle prend ses décisions pour les mauvaises raisons, sans fournir pour autant de solution générale pour leur correction.

En combinant le raisonnement sémantique et les interactions complexes des modèles, les approches NLE peuvent fournir des explications plus détaillées et intuitives. Cette nouvelle orientation représente une avancée significative vers une IA plus explicable et fiable, capable non seulement de prédire des résultats mais aussi de justifier ses prédictions de manière compréhensible et transparente. Nous nous pencherons plus en détail sur les approches de rationalisation automatique, qui intègrent des explications en langage naturel et permettent une meilleure compréhension des décisions prises par les modèles d'IA.

### 5.2 Techniques de rationalisation

#### 5.2.1 Machine Reading Comprehension

Le Machine Reading Comprehension (MRC) permet à un modèle de répondre à des questions concernant un contexte donné (Baradaran et al. 2022) [2]. Pour cette raison, il est également fréquemment appelé systèmes de réponse aux questions (AQ).

Plusieurs avancées ont été faites dans ce domaine, dont trois travaux importants que nous allons présenter ici.

- 1) Mihaylov et al. 2018, [24] ont introduit le **dataset OpenBook QA** qui consiste en des questions à choix multiples avec les réponses justifiées sur des données scientifiques.
- 2) Le **WorldTree V2 (Xie et al., 2020)** [44], a comme but principal de générer une explication du domaine scientifique avec une base de connaissances semi-structurée.

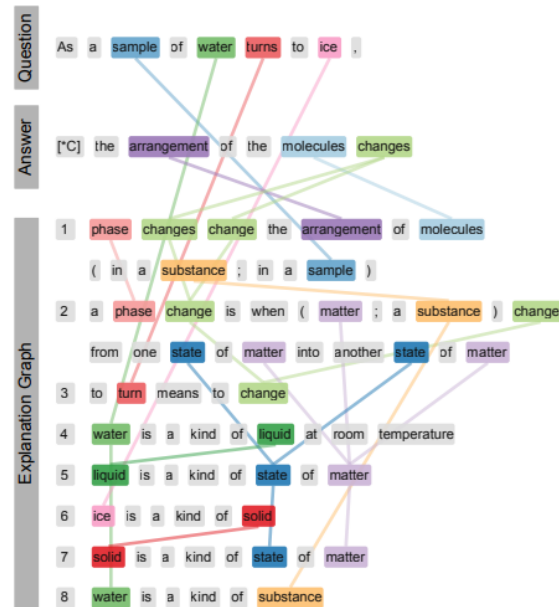


FIGURE 14 – Un exemple de question à choix multiples inspiré du WorldTree V2.

L'explication forme un "graphe d'explication" interconnecté avec les faits comme nœuds et les arêtes basées sur les mots partagés entre les faits et le texte de la question ou de la réponse.

Xie et al. (2020) [44] ont constaté que la plupart des modèles d'inférence multi-sauts, c'est-à-dire des modèles qui effectuent des inférences en combinant plusieurs faits, étaient limités à l'utilisation de deux ou trois faits au maximum. Cependant, dans leur étude, les auteurs ont réussi à fusionner en moyenne six faits à partir d'une base de connaissances semi-structurée de 9216 faits, ce qui a abouti à la création du corpus WorldTree V2 pour les questions scientifiques standardisées. Ce corpus comprend 5100 explications détaillées qui peuvent être utilisées pour former et instrumenter des systèmes de questions-réponses d'inférence multi-sauts.

3) Lakhotia et al. (2021) [15], ont proposé le **FiD-Ex (Fusion-in-Decoder Extractive)**. Les modèles seq2seq (séquence à séquence) sont des modèles de traitement automatique du langage naturel qui ont démontré leur efficacité pour générer des explications et des prédictions ensemble. Cependant, ces modèles nécessitent un grand ensemble de données labélisées pour l'entraînement et présentent des défis tels que la fabrication d'explications pour des prédictions incorrectes et la difficulté d'adaptation aux longs documents d'entrée.

Pour relever ces défis, le cadre FiD-Ex propose une nouvelle approche qui combine les avantages des modèles extractifs et génératifs. Les modèles extractifs sélectionnent des phrases ou des passages pertinents dans le document d'entrée pour répondre à une question, tandis que les modèles génératifs génèrent une réponse en utilisant des informations provenant de plusieurs sources. Dans FiD-Ex, les marqueurs de phrases sont utilisés pour encourager les explications extractives, ce qui permet de sélectionner les phrases les plus pertinentes dans le document d'entrée pour répondre à une question. Ensuite, ces phrases sont transmises à un décodeur pour générer une réponse cohérente et complète.

FiD-Ex utilise un affinage intermédiaire pour améliorer les performances en quelques coups sur les ensembles de données de QA (Questions-Réponses) à domaine ouvert. L'affinage intermédiaire consiste à pré-entraîner le modèle sur un ensemble de données connexe avant de l'affiner sur l'ensemble de données cible. Cela permet d'améliorer les performances du modèle lorsque les données d'entraînement sont limitées.

FiD-Ex est un nouveau cadre du Machine Reading Comprehension qui combine les avantages des modèles extractifs et génératifs pour fournir des explications et des prédictions précises. Les marqueurs de phrases et l’affinage intermédiaire sont utilisés pour améliorer les performances du modèle sur les ensembles de données de QA à domaine ouvert.

Ce nouvel outil a été testé sur les ensembles de données ERASER (Evaluating Rationales And Simple English Reasoning) et leurs références pour les évaluations (DeYoung et al. 2020) [11]. Cette expérience conclut que FiD-Ex améliore de manière significative les travaux précédents sur les métriques d’explication et de précision de tâches sur des modèles supervisés et few shot.

Ling et al. (2017) [18] ont présenté un ensemble de données et une approche qui fournit des justifications de réponse, des séquences de langage naturel et des expressions mathématiques lisibles par l’homme pour résoudre des problèmes algébriques écrits en langage naturel. Les auteurs ont proposé un modèle seq2seq qui génère une séquence d’instructions et fournit les justifications après avoir sélectionné la réponse.

Les auteurs ont pour cela construit un ensemble de données contenant 100 000 problèmes, dans lequel chaque question est décomposée en quatre parties - deux entrées et deux sorties.

L’ensemble de données est utilisé pour générer des justifications pour les problèmes de mathématiques et pour comprendre la qualité de ces justifications ainsi que la capacité à obtenir une réponse correcte. De plus, les auteurs ont utilisé un modèle seq2seq basé sur l’attention comme référence et ont comparé les résultats en fonction de la perplexité moyenne au niveau de la phrase et du BLEU-4 (Bilingual Evaluation Understudy). La perplexité est une mesure couramment utilisée pour évaluer la qualité d’un modèle de langage, tandis que le BLEU-4 est une métrique de qualité de la traduction automatique qui évalue la similarité entre la sortie générée par le modèle et une référence humaine. Les auteurs ont noté que cette nouvelle approche pouvait surpasser les modèles neuronaux existants en termes de capacité à résoudre des problèmes et de fluidité des justifications générées.

**Problem 1:**  
**Question:** Two trains running in opposite directions cross a man standing on the platform in 27 seconds and 17 seconds respectively and they cross each other in 23 seconds. The ratio of their speeds is:  
**Options:** A) 3/7 B) 3/2 C) 3/88 D) 3/8 E) 2/2  
**Rationale:** Let the speeds of the two trains be x m/sec and y m/sec respectively. Then, length of the first train = 27x meters, and length of the second train = 17 y meters.  $(27x + 17y) / (x + y) = 23 \rightarrow 27x + 17y = 23x + 23y \rightarrow 4x = 6y \rightarrow x/y = 3/2$ .  
**Correct Option:** B

FIGURE 15 – Un exemple de problème résolu avec la justification de la réponse ("rationale").

### 5.2.2 Commonsense Reasoning

La connaissance du sens commun aide les humains à naviguer dans les situations de la vie quotidienne. De même, le raisonnement de sens commun en TAL (traitement automatique du langage) est la capacité d’un modèle à aller au-delà de la reconnaissance de motifs et à utiliser des connaissances sur le monde pour faire des inférences (Apperly 2011 [7] ; Sap et al. 2020 [38]).

Sap et al. 2020 [38] démontrent une solution pour la connaissance du sens commun en utilisant un LSTM (Long Short Term Memory) encoder decoder. Le principal objectif était de convertir les actions d’un agent autonome en langage naturel à l’aide de la traduction automatique neuronale. Dans ce but, les auteurs ont construit un corpus de pensées de personnes alors qu’elles accomplissent des tâches dans le jeu Frogger, qui sont ensuite stockées sous forme d’états et d’actions. Dans l’étape suivante, des encodeurs et décodeurs LSTM ont été utilisés pour traduire les actions ainsi que les états en langage naturel. Enfin, les auteurs ont utilisé le score BLEU pour calculer la similarité des phrases et évaluer la précision de la sélection de la meilleure rationale.

L’expérience de Frogger se conclut par le fait que le cadre Encoder-Decoder a surpassé les modèles de référence et démontre que l’utilisation d’approches de théorie des jeux pour générer des justifications est

une technique prometteuse. Des travaux similaires de (Chang et al. 2020 [6] , 2019 [5] ; Li et al. 2022 [17] ; Yu et al. 2019 [45]) ont encore fait progresser les avancées dans ce domaine.

Rajani et al. (2019) [33] ont développé le cadre Commonsense Auto-Generated Explanations (CAGE) pour générer des explications pour la compréhension de questions de bon sens (CQA). Les auteurs ont également créé un nouveau jeu de données - Common Sense Explanations (CoSE) - en collectant des explications humaines pour le raisonnement de bon sens et en mettant en évidence des annotations. Dans cet article, les auteurs ont conclu que CAGE pouvait être efficacement utilisé avec des modèles de langage pré-entraînés pour améliorer les performances de raisonnement de bon sens.

### 5.2.3 Natural Language Inference

La tâche d'inférence de langage naturel (NLI) aide à identifier une hypothèse en langage naturel à partir d'une prémisse en langage naturel [22].

e-SNLI peut être utilisé pour divers objectifs mentionnés ci-dessus et peut également être utilisé pour améliorer les modèles ainsi que pour renforcer leur fiabilité.

Un autre problème avec l'inférence de langage naturel (NLI) est la fidélité des explications générées, abordée par Kumar et Talukdar 2020 [14] et Wiegrefe et al. 2020 [43]

Kumar et Talukdar 2020 [14] ont mentionné que les méthodes existantes ne fournissent pas de solution pour comprendre les corrélations des explications avec la prise de décision du modèle, ce qui peut affecter la fidélité des explications générées. En considérant ce problème, les auteurs ont proposé et présenté un nouveau cadre - NILE (Natural Language Inference over Label-specific Explanations). Le cadre NILE peut générer des explications en langage naturel pour chaque décision possible et traiter ces explications pour produire une décision finale pour les problèmes de classification.

NILE est une approche efficace pour fournir avec précision à la fois des labels et des explications. De plus, Kumar et Talukdar 2020 [14] se sont également concentrés sur la nécessité de la fidélité pour indiquer le processus de prise de décision du modèle en étudiant les justifications abstraites. L'auteur a proposé deux mesures : l'équivalence de robustesse et l'accord sur l'importance des caractéristiques.

Les auteurs ont réalisé une étude sur les ensembles de données CommonsenseQA (Talmor et al. 2019) [41] et SNLI en utilisant des modèles basés sur T5 (Narang et al. 2020) [30].

Le modèle T5 (Text-To-Text Transfer Transformer) a été introduit par (Narang et al. 2020) [30]. Il s'agit d'un modèle de transformation de texte à texte conçu pour unifier toutes les tâches de traitement du langage naturel (NLP) sous un même format : convertir toutes les tâches en une tâche de traduction où l'entrée et la sortie sont des textes. T5 repose sur l'architecture Transformer et a été pré-entraîné sur un vaste corpus de données textuelles, puis affiné pour des tâches spécifiques telles que la traduction, le résumé de texte, la classification de texte.

Les résultats montrent que ces modèles, basés sur l'architecture T5, présentent des propriétés intéressantes pour la génération d'explications fidèles et abstraites. Les modèles T5 se sont révélés efficaces pour produire des justifications en langage naturel, offrant un potentiel prometteur pour améliorer la compréhension et la transparence des décisions des modèles d'IA.

### 5.2.4 Text Classification

La classification de texte, également connue sous le nom de catégorisation de texte, est le processus d'attribution de labels ou d'étiquettes à des données textuelles telles que des phrases, des requêtes, des paragraphes et des documents (Minaee et al. 2021) [28]. Classifier le texte et extraire des informations peut conduire à une compréhension plus riche des données, mais en raison de leur nature non structurée, cela est difficile et fastidieux. Les techniques de traitement du langage naturel (NLP) dans la classification de texte permettent l'annotation et la labélisation automatique des données afin de faciliter l'obtention de ces informations plus approfondies.

Zhang et al. (2016) [47] proposent un nouveau modèle CNN pour la classification de texte qui exploite les justifications associées aux documents.

Les Convolutional Neural Networks (CNN), traditionnellement utilisés en vision par ordinateur pour extraire des caractéristiques spatiales des images, ont également trouvé des applications efficaces en traitement du langage naturel (NLP). Leur capacité à détecter des motifs locaux et à les combiner pour

former une compréhension globale s'avère utile pour diverses tâches de NLP, telles que la classification de texte et l'analyse de sentiments.

### **Comment les CNN s'appliquent-ils au texte ?**

#### — Représentation des mots :

Les mots ou les phrases sont d'abord convertis en vecteurs numériques à l'aide de techniques d'embedding comme Word2Vec ou GloVe. Chaque mot est représenté par un vecteur de dimension fixe.

#### — Convolution et Pooling :

Les vecteurs de mots sont ensuite traités par des filtres de convolution. Un filtre de convolution glisse sur les vecteurs de mots pour détecter des motifs locaux, tels que des combinaisons de mots ou des n-grammes, similaires à la manière dont ils détectent des motifs dans les images. Après la convolution, une opération de pooling (souvent max-pooling) est appliquée pour réduire la dimensionnalité tout en conservant les informations les plus importantes détectées par les filtres.

#### — Combinaison des caractéristiques locales :

Les caractéristiques locales détectées par les filtres de convolution et réduites par le pooling sont combinées pour former une représentation globale du texte. Cette représentation est ensuite utilisée pour des tâches de classification.

Dans leur article, les auteurs affirment être les premiers à incorporer des justifications dans les modèles neuronaux pour la classification de texte. Les auteurs utilisent un réseau de neurones convolutionnel (CNN) qui analyse chaque phrase du document de manière indépendante pour estimer la probabilité que cette phrase serve de justification. Ils démontrent que leur technique surpasse les méthodes standards et les différentes variantes de CNN sur cinq ensembles de données de classification.

Intuitivement, utiliser plus de données peut conduire à une meilleure prise de décision par les réseaux de neurones. Zaidan et al. (2007) [46] proposent un nouveau cadre pour améliorer les performances de l'apprentissage automatique supervisé en utilisant des "types" de données plus riches. Leur approche, appelée "annotator rationales", consiste à exploiter un ensemble de données d'entraînement avec des justifications annotées. Les justifications mettent en évidence les preuves soutenant la prédiction. Zaidan et al. (2007) [46] testent leur approche sur des tâches de catégorisation de texte, spécifiquement la classification de sentiments de critiques de films. Ils affirment que ces justifications permettent à la machine de comprendre pourquoi la prédiction est faite de cette manière. Les justifications aident le modèle à distinguer le signal du bruit. Les algorithmes d'apprentissage automatique font face au "problème d'attribution de crédit" qui signifie que de nombreuses caractéristiques dans les données (X) pourraient avoir affecté le résultat prédit (Y).

Liu et al. (2019b) [20] ont proposé un nouveau cadre génératif d'explication (GEF) pour les problèmes de classification qui peut générer des explications détaillées. La motivation derrière ce cadre d'explication est de fournir des explications lisibles par l'homme sans ignorer les informations détaillées telles que les explications textuelles pour le label.

Pour comprendre la précision des explications, les auteurs ont mené des expériences sur deux ensembles de données - PCMag et Skytrax User Reviews. Ces ensembles de données ont été traités à l'aide du tokenizer Stanford. De plus, les auteurs ont utilisé une architecture Encoder-Predictor dans laquelle ils ont utilisé un auto-encodeur variationnel conditionnel (CVAE) comme modèle de base pour les explications textuelles et une mémoire à long terme courte (LSTM) pour les explications numériques.

Les résultats expérimentaux ont indiqué qu'après avoir combiné les modèles de base avec GEF, les performances du modèle de base ont été améliorées avec une meilleure qualité des explications. En outre, les auteurs ont utilisé une évaluation humaine pour évaluer l'explicabilité des explications textuelles générées.

Cet article semble très intéressant pour les problématiques EDF : pour qu'un modèle d'IA soit compréhensible par un non-expert, que ce soit un client, ou encore un conseiller clientèle, il est utile qu'il soit expliqué comme l'aurait fait un humain. Nous avons alors décidé d'implémenter ce papier pour pouvoir l'appliquer par la suite à un cas d'usage proche de ceux d'EDF (cf. partie 7).

## 5.3 Challenges

### 5.3.1 Evaluations statistiques

Il n'existe actuellement aucune évaluation statistique standard pour la rationalisation. Il existe une grande variété de mesures utilisées, telles que l'erreur quadratique moyenne (Lei et al. 2016) [16], la précision (Du et al. 2019 [12]; Rajani et al. 2019 [33]; Zaidan et al. 2007 [46]), le score F1 (Alhindi et al. 2018 [1]; Rana et al. 2022 [35]), l'ANOVA (analyse de la variance) (Das et Chernova 2020) [9] et précision (Plyler et al. 2022) [32]. La métrique statistique la plus utilisée est la précision.

Cependant, cette métrique concerne uniquement les tâches de prédiction, et est insuffisante pour une tâche d'explicabilité, où l'on doit à la fois réaliser une première tâche de prédiction (ou classification) avec une seconde tâche de justification. Si la prédiction est très précise mais que la NLE (Natural Language Explanation) est peu claire alors cela perd en utilité.

Pour évaluer la qualité du texte généré, il existe plusieurs métriques telles que le BLEU (BiLingual Evaluation Understudy) score by Papineri et al. (2002) [31] and the ROUGE-N (Recall-Oriented Understudy for Gisting Evaluation) score by Lin (2004) [19]. Ces métriques servent à comparer le texte généré avec l'ensemble de textes de références (set of ground truth reference texts). Les textes de référence sont rédigés par des humains. Ces évaluations ont cependant une limitation car elles se font au niveau des tokens (similarités entre les n-grammes qui sont des séquences de n mots).

Voici un exemple :

**Phrase de référence** : "Le chat mange la souris." | **Phrase générée** : "La souris mange le chat."

Bien que les deux phrases aient des significations complètement différentes, elles partagent de nombreux n-grammes (par exemple, "le chat", "chat mange", "mange la", "la souris" pour les bigrammes). Par conséquent, la phrase générée pourrait obtenir un score élevé selon les métriques ROUGE-N et BLEU, malgré le fait qu'elle soit sémantiquement incorrecte par rapport à la phrase de référence.

### 5.3.2 Métriques

#### ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

Pour chaque texte à résumer, on dispose de références, qui peuvent être un ou plusieurs textes résumés du texte initial. Le candidat (ou "candidate" dans les formules ci-après) est alors la proposition de résumé faite par le LLM.

$$\text{Recall} = \frac{\text{Overlapping n-grams}}{\text{Number of n-grams in reference}} \quad (1)$$

$$\text{Precision} = \frac{\text{Overlapping n-grams}}{\text{Number of n-grams in candidate}} \quad (2)$$

À partir de là, le calcul est ensuite un simple  $F_1$ -score.

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

Il existe plusieurs variantes de ROUGE, chacune utilisant des n-grammes de différentes longueurs. Voyons quelques exemples de ces variantes.

#### ROUGE-1

La métrique ROUGE-1 utilise uniquement des uni-grammes, c'est-à-dire n'utilise que la présence ou non de chaque mot obtenu dans la référence.

**Reference** : (Hello), (everyone), (on), (Blent)  
**Candidate** : (Welcome), (everyone), (on), (Blent), (website)

En comparant la référence avec le candidat, il y a 3 mots sur 4 du candidat présents dans la référence : le rappel est de 0.75 . En revanche, seuls 3 mots de la référence sont présents parmi les 5 du candidats, donc la précision est de 0.6 . Ainsi, le  $F_1$ -score pour ce texte est d'environ 0.67.

## ROUGE-2

Dans le cas de la métrique ROUGE-2, on n'utilise plus cette fois-ci les uni-grammes mais les bi-grammes : ce sont dorénavant des séquences de couples de mots qui sont utilisés.

**Reference** : (Hello, everyone), (everyone, on), (on, Blent)

**Candidate** : (Welcome, everyone), (everyone, on), (on, Blent), (Blent, website)

Nous voyons que 2 séquences du candidat sont présentes parmi les 3 séquences de la référence, ce qui donne un rappel de 0.67 (arrondi supérieur). À l'inverse, il y a aussi 2 séquences de la référence dans les 5 séquences du candidat, soit une précision de 0.4. Le F1-score est donc de 0.5.

## ROUGE-N

La métrique ROUGE-n permet d'effectuer les mêmes opérations mais en considérant des séquences de n-grammes. Évidemment, plus la valeur de n-grammes est élevée, plus les séquences qui doivent être exactement retrouvées entre la référence et le candidat sont grandes.

## BLEU (Bilingual Evaluation Understudy)

BLEU est une métrique principalement utilisée pour évaluer la qualité de la traduction entre deux langues. Tout comme ROUGE, cette métrique utilise également le concept de n-grammes ainsi que le calcul de la précision, mais présente une modification.

**Reference** : (Hello), (world), (on), (this), (website)

**Candidate** : (Hello), (Hello), (Hello), (Hello), (Hello)

Ici, nous avons 5 fois le même mot qui apparaît dans le candidat. Puisque les 5 mots (identiques) apparaissent parmi les 5 mots de la référence, la précision est de 1... ce qui n'est pas vraiment réaliste ici puisque le générateur de texte ne fait que répéter le même mot à chaque fois !

Pour pallier ce problème, BLEU calcule la notion de Modified Precision  $p_n$ . La solution pour contourner ce problème, c'est de tronquer le nombre d'occurrences d'un mot dans le candidat par rapport à la référence, noté ici MaxRefCount.

$$\text{Count}_{\text{clip}} = \min(\text{Count}, \text{MaxRefCount}) \quad (4)$$

Dans notre exemple, le mot **Hello** n'apparaît qu'une seule fois dans la référence, ce qui donne :

$$\text{Count}_{\text{clip}} = \min(5, 1) = 1.$$

Ainsi, la valeur de  $p_n$  ici est égale à  $\frac{1}{5} = 0.2$ .

Un autre problème qui peut survenir est lorsque les phrases du candidat sont plus petites que celles de la référence. Dans cette situation, BLEU calcule également le *Brevity Penalty* pour **pénaliser les prédictions trop courtes par rapport aux références**. Il s'agit simplement d'un calcul qui compare la longueur (nombre de mots) de la référence  $r$  avec celle du candidat  $c$ .

$$\text{BP} = \begin{cases} 1 & \text{si } c > r \\ \exp\left(1 - \frac{r}{c}\right) & \text{si } c \leq r \end{cases}$$

À partir d'ici, la métrique BLEU se calcule comme une moyenne géométrique, en calculant la Modified Precision  $p_n$  pour tous les n-grammes de 1 à  $N$  (étant ici le  $n$  maximum de n-grammes que l'on peut former) et  $w_1, \dots, w_N$  des pondérations dont la somme vaut 1.

$$\text{BLEU} = \text{BP} \exp\left(\sum_{i=1}^N w_n \log p_n\right)$$

L'intérêt des pondérations est de **ne pas considérer certains n-grammes qui seraient trop grands ou trop petits** par exemple. Ainsi, on pourrait privilégier les bi-grammes ou tri-gramme au détriment des n-grammes d'ordres supérieurs. Dans la pratique, le calcul de la métrique BLEU se limite souvent jusqu'aux tétragrammes (4 mots).



### 5.3.3 Data

Disponibilité : La collecte de données est une tâche coûteuse et chronophage. Il est possible de réutiliser des ensembles de données existants, mais leur modification nécessite un travail humain manuel. C'est pourquoi les chercheurs construisent souvent leurs ensembles de données pour une tâche spécifique sur laquelle ils travaillent. Camburu et al. (2018) [4] ont développé l'ensemble de données e-SNLI en modifiant l'ensemble de données SNLI de Bowman et al. (2015b) [3]. Camburu et al. (2018) [4] ont obtenu des résultats prometteurs dans leur tâche, démontrant comment leur jeu de données peut permettre un large éventail de nouvelles directions de recherche en modifiant et en réaffectant des jeux de données existants.

Diversité : Sans suffisamment d'ensembles de données, les nouvelles recherches en rationalisation seront limitées. Les chercheurs seront limités aux ensembles de données existants pour réaliser de nouveaux progrès. Il existe une relation directe entre la disponibilité des données et les progrès réalisés. Plus de travail dans la création de nouveaux ensembles de données pour la rationalisation peut aider à améliorer la diversité et les progrès de certains domaines nécessitant encore des avancées, comme le NMT (Neural Machine Translation). De nouveaux ensembles de données dans tous les domaines, en général, augmenteront l'intérêt et le travail de rationalisation car les chercheurs auront plus de flexibilité pour concevoir de nouvelles techniques et expérimenter avec un large éventail de données.

## 6 Towards Explainable NLP : A Generative Explanation Framework for Text Classification

### 6.1 Objectifs

Comme évoqué dans la partie 5.2.4, nous allons maintenant présenter plus en détail le cadre génératif d'explication (GEF) de Liu et al. [20].

Les auteurs se sont penchés sur ce sujet après avoir constaté que les approches existantes pour les systèmes d'apprentissage automatique explicables se concentrent principalement sur l'interprétation des résultats ou des relations entre les entrées et les sorties.

Cependant, les informations détaillées (informations textuelles pour les labels) sont souvent ignorées.

La problématique de recherche à laquelle les auteurs ont essayé de répondre est la suivante :

*Comment pouvons-nous générer des explications détaillées pour les décisions que notre modèle de classification prend ?*

Pour s'attaquer à ce problème les auteurs ont fait deux contributions :

- Ils ont proposé un nouveau cadre génératif d'explication qui apprend à prendre des décisions de classification et à générer des explications détaillées en même temps.
- Ils ont introduit l'Explanation Factor et l'approche du Minimum Risk Training qui apprend à générer des explications plus raisonnables.

### 6.2 Données

Les données utilisées sont des données qui comprennent 6 colonnes :

- **product\_name** : Nom du produit associé à un appareil.
- **overall\_rating** : Note de 0 à 5 (on peut les considérer comme des étoiles sur les sites e-commerces) qui note le produit, 0 est la pire note et 5 la meilleure note.
- **review\_text** : Cette colonne fournit une description complète de l'expérience de l'auteur de l'avis avec le produit.
- **positive\_comment** : Cette colonne contient le texte intégral de l'évaluation. Elle fournit une description complète de l'expérience positive de l'auteur de l'avis avec le produit.
- **negative\_comment** : Cette colonne contient le texte intégral de l'évaluation. Elle fournit une description complète de l'expérience négative de l'auteur de l'avis avec le produit.
- **neutral\_comment** : Cette colonne contient le texte intégral de l'évaluation. Elle fournit une description complète de l'expérience neutre de l'auteur de l'avis avec le produit.

	product_name	overall_rating	review_text	positive_comment	negative_comment	neural_comment
0	Nikon D5500	4.0	Nikon has a habit of refreshing its midrange c...	Omits optical low-pass filter \nSharp vari-an...	Smaller body means controls are somewhat cramp...	The Nikon D5500 D-SLR delivers images that are...
1	Trend Micro InterScan Gateway Security Applian...	4.0	You 've already got your small or medium busin...	High-performance malware scanner \n10/100/100...	Desktop cleanup is Windows and IE only \nDoes...	This appliance gives Microsoft shops an easy-t...
2	Motorola Krave ZN4	3.5	The sexy Motorola Krave ZN4 for Verizon looks ...	Beautiful design with high-resolution touch sc...	Other than the mobile TV , the software is a b...	Verizon 's Motorola Krave ZN4 is a gorgeous li...
3	HelloTalk	3.5	HelloTalk is an app that helps you practice la...	Encourages free-form interactions \nMatches L...	No structured learning content \nValue of the...	When you 're ready to experiment with a foreig...
4	Symantec Norton Security Deluxe (for Mac)	4.0	Antivirus protection under the Norton name has...	Certified by one independent testing lab \nFa...	Expensive \nSo-so scores against phishing sit...	Symantec 's Norton Security Deluxe delivers ex...

FIGURE 16 – Aperçu du dataset PCMAG.

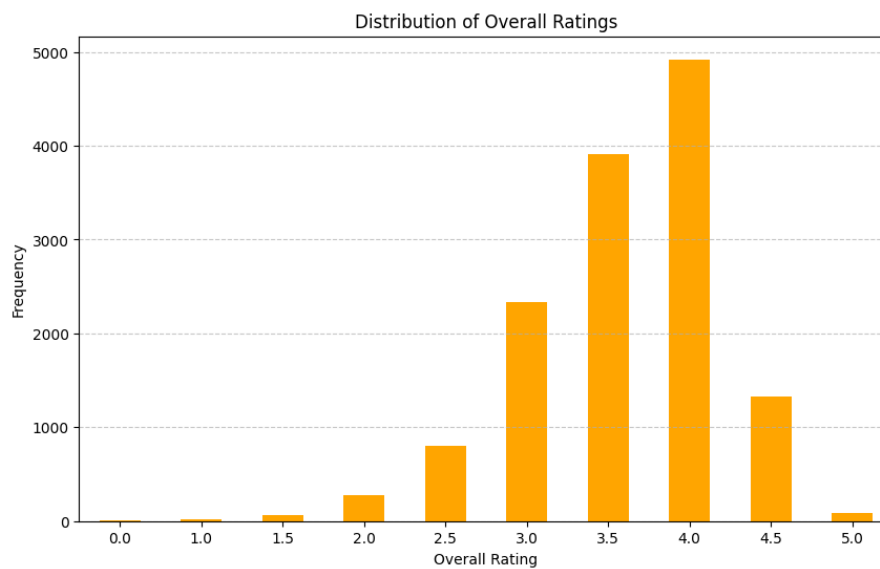


FIGURE 17 – Fréquence des notes globales ("overall ratings").

### 6.3 Méthodologie

Les informations détaillées sont dans notre cas le texte *review\_text*, *positive\_comment*, *negative\_comment*, *neutral\_comment* et les notes d'évaluation *overall\_ratings*.

Le but est ici de prédire la note d'évaluation de chaque produit du jeu de données test (tâche de classification) et de générer des explications textuelles (tâche de génération de texte).

### 6.4 Generative Explanation Framework

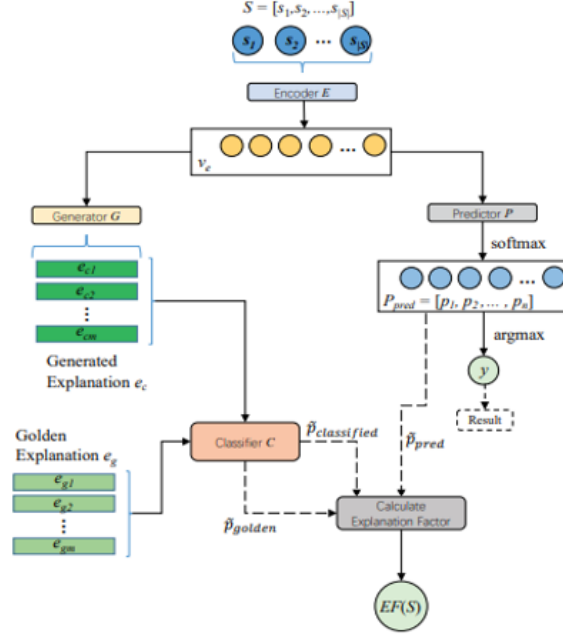


FIGURE 18 – Architecture du Generative Explanation Framework

Il y a 2 parties importantes dans le GEF, le générateur  $G$  et le prédicteur  $P$ .

$E$  encode  $S$  en un vecteur de représentation  $V_e$ .

Le générateur  $G$  prend  $V_e$  en entrée et génère des explications  $e_c$ .

Les Golden Explanations sont ici considérées comme des explications de références (ce sont les *positive\_comment*, *negative\_comment* et *neutral\_comment* dans notre cas).

Le Classifieur  $C$  est un classifieur pré-entraîné qui permettra de pouvoir classer le texte généré en 3 labels (positif, négatif et neutre) donnant ainsi  $Y$  comme indiqué dans le schéma en figure 18.

Ce classifieur apprend à prédire la catégorie  $Y$  en prenant directement les explications comme entrée. Son objectif est d'imiter le comportement humain. Il devrait prédire les évaluations globales plus précisément que le modèle de base avec le texte original en entrée.

Les auteurs l'utilisent pour fournir une orientation solide à l'encodeur de texte  $E$ , le rendant capable de générer une représentation plus informative du vecteur  $V_e$  (vecteur d'entrée de texte).

Voici les formules de la probabilité de distribution des Golden Explanations  $P_{gold}$  et de la probabilité de distribution des résultats prédits  $P_{classified}$ .

$$P_{classified} = \text{softmax}(f_C(W_C \cdot e_C + b_C)) \quad (5)$$

$$P_{gold} = \text{softmax}(f_C(W_C \cdot e_G + b_C)) \quad (6)$$

Pour mesurer la distance entre les résultats prédits, les explications générées et les explications de référence, nous extrayons les probabilités de référence  $\tilde{p}_{classified}$ ,  $\tilde{p}_{pred}$  et  $\tilde{p}_{gold}$  à partir de  $P_{classified}$ ,  $P_{pred}$  et  $P_{gold}$

respectivement. Elles seront utilisées pour mesurer la divergence entre le résultat prédit et le résultat de référence dans l'entraînement à risque minimal.

## 6.5 Explanation Factor

$$EF(S) = |\tilde{p}_{\text{classified}} - \tilde{p}_{\text{gold}}| + |\tilde{p}_{\text{classified}} - \tilde{p}_{\text{pred}}| \quad (7)$$

$$|\tilde{p}_{\text{classified}} - \tilde{p}_{\text{gold}}| \quad (8)$$

Pour mesurer la distance entre les explications générées *ec* et les explications de référence *eg*, nous considérons que si des explications similaires sont fournies au classifieur C, des prédictions similaires devraient être générées. Par exemple, si nous fournissons une explication de référence "Great performance" au classifieur C et qu'il indique que cela signifie "un bon produit", alors nous fournissons une autre explication "Excellent performance" à C, il devrait également indiquer que l'explication signifie "un bon produit".

$$|\tilde{p}_{\text{classified}} - \tilde{p}_{\text{pred}}| \quad (9)$$

Les explications générées doivent être en mesure d'interpréter le résultat global. Par exemple, si le modèle de base prédit que S est "un bon produit", mais que le classifieur a tendance à classer *ec* comme étant les explications pour "un mauvais produit", alors *ec* ne peut pas expliquer correctement la raison pour laquelle le modèle de base donne de telles prédictions.

## 6.6 Minimum Risk Training

Le but du Minimum Risk Training est de renforcer les connexions entre les informations détaillées (fine-grained information) et le texte d'entrée.

$$\mathcal{L}_{MRT}(e_g, S, \theta) = \sum_{(e_g, S) \in D} \mathbb{E}_{y(e_g, S, \theta)} \Delta(y, \tilde{y}) \quad (10)$$

Où :

$$y(e_g, S, \theta) \quad : \text{Ensemble des résultats prédits} \quad (11)$$

$$\Delta(y, \tilde{y}) \quad : \text{Distance sémantique entre les résultats prédits et la vérité de référence} \quad (12)$$

$$\mathcal{L}_{MRT}(e_g, S, \theta) = \sum_{(e_g, S) \in D} \mathcal{L}(e_g, S, \theta) EF(S) \quad (13)$$

On sait que :

$$\mathbb{E}_{y(e_g, S, \theta)} \quad \text{est l'espérance sur l'ensemble } \mathcal{Y}(e_g, S, \theta), \quad \text{qui représente la perte globale} \quad (14)$$

Afin d'éviter la dégradation totale de la perte, ils définissent la fonction de perte finale comme la somme de la perte de génération d'explications et la somme de la perte du MRT.

$$\mathcal{L}_{\text{final}} = \sum_{(e_g, S) \in D} \mathcal{L} + \mathcal{L}_{MRT} \quad (15)$$

## 7 Mise en pratique

### 7.1 Jeu de données Open Data utilisé

L'utilisation des données client étant très contrainte chez EDF à cause des réglementations RGPD, il est très fréquent d'utiliser des données open data dans les benchmarks méthodologiques. Procéder ainsi facilite notamment le partage des résultats et la publication d'articles scientifique. Pour la mise en pratique de la méthode présentée dans la partie précédente, nous avons donc conservé le jeu de données open data mis à disposition par les auteurs. Il s'agit des données présentées brièvement dans la partie 6.2.

#### 7.1.1 Prétraitements des données

Après exploration des données brutes utilisées dans l'article, nous nous sommes aperçus que leur exploitation directe était impossible sans re-traitement, pour la génération de texte et la classification. Nous avons alors décidé de reformater le dataframe en 5 colonnes :

- **product\_name** : Nom du produit associé
- **overall\_rating** : Notes globales
- **review\_text** : Description complète de l'expérience de l'auteur de l'avis avec le produit
- **comment** : Avis positif négatif ou neutre du produit
- **label** : Positive, Negative ou Neutre selon l'avis

Voici un exemple du dataframe après avoir effectué le preprocessing :

	product_name	overall_rating	review_text	comment	label
0	Nikon D5500	4.0	Nikon has a habit of refreshing its midrange c...	Omits optical low-pass filter .\nSharp vari-an...	Pos
1	Nikon D5500	4.0	Nikon has a habit of refreshing its midrange c...	Smaller body means controls are somewhat cramp...	Neg
2	Nikon D5500	4.0	Nikon has a habit of refreshing its midrange c...	The Nikon D5500 D-SLR delivers images that are...	Neu
3	Trend Micro InterScan Gateway Security Applian...	4.0	You 've already got your small or medium busin...	High-performance malware scanner .\n10/100/100...	Pos
4	Trend Micro InterScan Gateway Security Applian...	4.0	You 've already got your small or medium busin...	Desktop cleanup is Windows and IE only .\nDoes...	Neg
5	Trend Micro InterScan Gateway Security Applian...	4.0	You 've already got your small or medium busin...	This appliance gives Microsoft shops an easy-t...	Neu
6	Motorola Krave ZN4	3.5	The sexy Motorola Krave ZN4 for Verizon looks ...	Beautiful design with high-resolution touch sc...	Pos
7	Motorola Krave ZN4	3.5	The sexy Motorola Krave ZN4 for Verizon looks ...	Other than the mobile TV , the software is a b...	Neg
8	Motorola Krave ZN4	3.5	The sexy Motorola Krave ZN4 for Verizon looks ...	Verizon 's Motorola Krave ZN4 is a gorgeous li...	Neu
9	HelloTalk	3.5	HelloTalk is an app that helps you practice la...	Encourages free-form interactions .\nMatches I...	Pos

FIGURE 19 – Aperçu du dataset PCMAG preprocessed.

### 7.2 VAE

Avant de me plonger dans l'architecture proposée par le papier, j'ai entrepris une réflexion méthodique en explorant les différentes architectures de génération de données. Mon objectif était de comprendre les principes fondamentaux et les approches existantes pour pouvoir les appliquer de manière efficace.

Tout d'abord, j'ai commencé par me renseigner sur les diverses architectures de génération de données, ce qui m'a conduit naturellement à étudier en profondeur les Auto-Encodeurs Variationnels (VAE, pour Variational Auto Encoders). Les VAE sont des modèles génératifs puissants qui permettent de représenter des données complexes de manière probabiliste. Ils se distinguent par leur capacité à apprendre une distribution latente des données, ce qui facilite la génération de nouvelles instances similaires aux données d'origine.

### 7.2.1 Architecture

Les VAE fonctionnent en deux étapes principales : l'encodage et le décodage. Pendant l'encodage, les données d'entrée sont transformées en une distribution latente. Ensuite, pendant le décodage, des échantillons de cette distribution latente sont utilisés pour générer de nouvelles données. Cette approche permet non seulement de compresser les informations essentielles des données d'entrée, mais aussi de générer des données nouvelles et plausibles en échantillonnant la distribution latente.

Après avoir acquis une compréhension solide des VAE, j'ai examiné comment cette architecture pouvait être appliquée pour générer des explications détaillées et compréhensibles pour les problèmes de classification. Ce cadre théorique m'a fourni une base solide pour aborder l'architecture spécifique proposée dans le papier, que je vais détailler dans les sections suivantes.

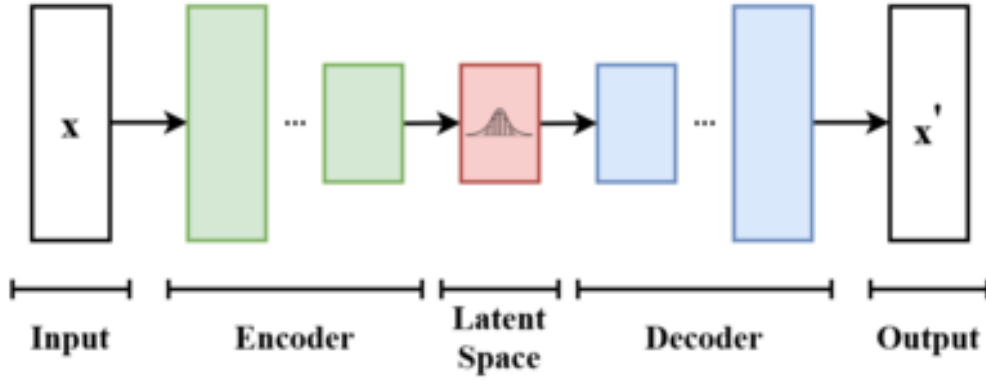


FIGURE 20 – Architecture d'un VAE.

### 7.2.2 Encoder

L'encodeur est un réseau de neurones qui prend les données d'entrée  $x$  et les transforme en une distribution latente  $q(z | x)$ . Contrairement aux auto-encodeurs classiques qui compressent les données en un vecteur de caractéristiques déterministe, les VAE apprennent les paramètres d'une distribution probabiliste (souvent une distribution gaussienne) dans l'espace latent.

L'encodeur produit deux sorties :

- $\mu(x)$  : le vecteur des moyennes.
- $\sigma(x)$  : le vecteur des écarts types (souvent log-variances pour la stabilité numérique).

Les variables latentes  $z$  sont ensuite échantillonnées à partir de cette distribution  $\mathcal{N}(\mu(x), \sigma(x)^2)$ .

Formellement, l'encodeur peut être représenté comme suit :

$$q(z | x) = \mathcal{N}(z; \mu(x), \sigma(x)^2)$$

### 7.2.3 Échantillonnage

L'étape suivante consiste à échantillonner les variables latentes  $z$  à partir de la distribution  $q(z | x)$ . Cependant, l'échantillonnage direct n'est pas différentiable, ce qui pose un problème pour l'optimisation par gradient. Pour surmonter cette difficulté, les VAE utilisent le trick de reparamétrisation.

Avec le trick de reparamétrisation, nous exprimons  $z$  comme :

$$z = \mu(x) + \sigma(x) \odot \epsilon$$

où  $\epsilon$  est une variable aléatoire échantillonnée à partir d'une distribution normale standard  $\mathcal{N}(0, I)$ , et  $\odot$  représente la multiplication élément par élément.

### 7.2.4 Décodeur

Le décodeur est un autre réseau de neurones qui prend les variables latentes  $z$  et les utilise pour reconstruire les données d'origine  $x$ . Le décodeur apprend la distribution conditionnelle  $p(x | z)$ , souvent modélisée comme une distribution gaussienne lorsque les données sont continues, ou une distribution de Bernoulli pour des données binaires.

Formellement, le décodeur peut être représenté comme suit :

$$p(x | z) = \mathcal{N}(x; \hat{\mu}(z), \hat{\sigma}(z)^2)$$

où  $\hat{\mu}(z)$  et  $\hat{\sigma}(z)$  sont les paramètres de la distribution reconstruite, prédits par le décodeur.

### 7.2.5 Fonction de perte

L'apprentissage d'un VAE repose sur la minimisation d'une fonction de perte composée de deux termes principaux :

- **Erreur de reconstruction** : mesure la divergence entre les données d'entrée  $x$  et les données reconstruites  $\hat{x}$ . Pour des données continues, il s'agit souvent de l'erreur quadratique moyenne (MSE), et pour des données binaires, de l'entropie croisée.

$$\mathcal{L}_{\text{reconstruction}} = -\mathbb{E}_{q(z|x)}[\log p(x | z)]$$

- **Divergence de Kullback-Leibler (KL)** : mesure la divergence entre la distribution latente  $q(z | x)$  et une distribution prior  $p(z)$ , généralement une distribution gaussienne standard  $\mathcal{N}(0, I)$ .

$$\mathcal{L}_{\text{KL}} = D_{\text{KL}}(q(z | x) \parallel p(z))$$

La fonction de perte totale à minimiser est alors :

$$\mathcal{L}_{\text{VAE}} = \mathcal{L}_{\text{reconstruction}} + \mathcal{L}_{\text{KL}}$$

### 7.2.6 Entraînement du VAE

L'entraînement d'un VAE consiste à optimiser cette fonction de perte totale en utilisant des méthodes de gradient, typiquement avec des algorithmes comme Adam. Grâce à la structure probabiliste et à la technique de reparamétrisation, le VAE apprend à générer des données réalistes à partir de l'espace latent, tout en assurant que cet espace latent capture des caractéristiques significatives des données d'origine.

### 7.2.7 Implémentation

#### Préparation des données

##### 1) Chargement des données :

- Charger les données d'entraînement, de validation et de test à partir de fichiers CSV contenant des critiques avec des commentaires positifs, négatifs et neutres.
- Les données sont concaténées en une seule colonne pour former un texte combiné.

Ici le but était de générer du texte d'où la concaténation en une seule colonne.

##### 2) Tokenization et padding :

- Utilisation du tokenizer BERT pour convertir les textes en séquences d'IDs de tokens.
- Les séquences sont ensuite complétées (padding) pour avoir toutes la même longueur.

##### 3) Préparation des embeddings GloVe :

- Chargement des embeddings GloVe pré-entraînés pour initialiser une matrice d'embedding.
- Construction d'une matrice d'embedding GloVe, en utilisant les embeddings disponibles pour les mots du tokenizer.

### Modèle VAE

##### 1) Encodeur VAE :

- Utilisation d'un réseau LSTM bidirectionnel pour encoder les séquences tokenisées en une représentation latente.
- Calcul des paramètres de distribution ( $\mu$ ,  $\log\sigma$ ) pour le modèle VAE à partir de l'encodeur LSTM.

## 2) Décodeur VAE :

- Décodeur LSTM qui reçoit la représentation latente et génère une séquence de tokens prédite.
- Utilisation d'une couche linéaire pour prédire les probabilités des tokens dans le vocabulaire.

## 3) Fonction de perte VAE :

- Définition d'une fonction de perte combinant une perte de reconstruction (`CrossEntropyLoss`) et une perte KL (perte de divergence de Kullback-Leibler) pour régulariser la distribution latente.

## Entraînement du modèle

### 1) Initialisation et optimisation :

- Initialisation du modèle VAE, des optimiseurs (Adam) et des paramètres d'entraînement.
- Gestion des batches et de la vérification des indices de tokens pour éviter les dépassements de mémoire.

### 2) Boucle d'entraînement :

- Boucle d'entraînement sur plusieurs époques, où chaque batch est utilisé pour calculer la perte, effectuer la rétropropagation et mettre à jour les poids du modèle.
- Affichage périodique de la perte moyenne pour chaque époque.

## Evaluation et génération de texte

### 1) Conversion des tokens en texte :

- Fonction pour convertir les séquences d'IDs de tokens en texte lisible, en filtrant les tokens spéciaux comme `<UNK>` et `<PAD>`.

### 2) Evaluation du modèle :

- Passage en mode évaluation pour générer des prédictions sur un jeu de données de validation.
- Utilisation du modèle pour reconstruire des séquences de texte et comparer les résultats avec le texte original.

## 7.2.8 Résultats

### Exemple de résultat :

##### — Hdzhkdzk the drink is cnz#-#—

Le résultat obtenu dans cet exemple, bien que composé de caractères et de mots en langage naturel, n'est pas encore parfait et laisse entrevoir des possibilités d'amélioration. En effet, on reconnaît bien des mots qui existent et une structure de phrase (sujet-verbe) mais qui est couplée avec des caractères indésirables et une absence de sens. La présence de ces éléments et le manque de cohérence dans certaines phrases générées nous ont conduit à rechercher des pistes d'amélioration pour notre modèle.

Nous avons identifié des méthodes d'encodage/décodage et des choix d'embedding plus performants pour notre modèle, tels que l'utilisation d'un transformer et de l'embedding BERT.

Voici les 2 suppositions :

- **Méthode d'encodage/décodage avec le LSTM :** Les performances actuelles peuvent être améliorées en remplaçant l'encodage/décodage LSTM par une méthode utilisant un transformer. Les transformers sont connus pour leur capacité à capturer des dépendances à longue portée dans les données textuelles, ce qui pourrait potentiellement produire des résultats de meilleure qualité et plus cohérents.

- **Choix de l'embedding :** Actuellement, les embeddings GloVe sont utilisés plutôt que ceux de BERT. BERT, étant un modèle de langage pré-entraîné basé sur des transformers, pourrait offrir une représentation



plus riche et contextuellement appropriée des mots, ce qui pourrait conduire à des générations de texte plus précises et naturelles.

Nous avons décidé de suivre ces pistes dans l'implémentation du CVAE.

### 7.3 CVAE

Après avoir exploré et compris les avantages ainsi que les limites du Variational Autoencoder (VAE), nous avons choisi de nous orienter vers une approche plus avancée : le Conditional Variational Autoencoder (CVAE). Cette transition s'inscrit dans notre volonté d'améliorer la capacité du modèle à générer des données tout en contrôlant de manière conditionnelle leur distribution latente. Avant de plonger dans l'architecture détaillée du CVAE, examinons de plus près pourquoi cette évolution est pertinente et les bénéfices attendus par rapport à notre précédent modèle.

#### 7.3.1 Architecture

Les CVAE sont une extension des VAE qui incluent des informations conditionnelles lors de la génération de données. Contrairement aux VAE classiques qui génèrent des données indépendamment de toute condition, les CVAE permettent de générer des données conditionnellement à une variable latente et à des conditions spécifiques. Cela les rend particulièrement adaptés à la génération de données contrôlées par des facteurs spécifiques.

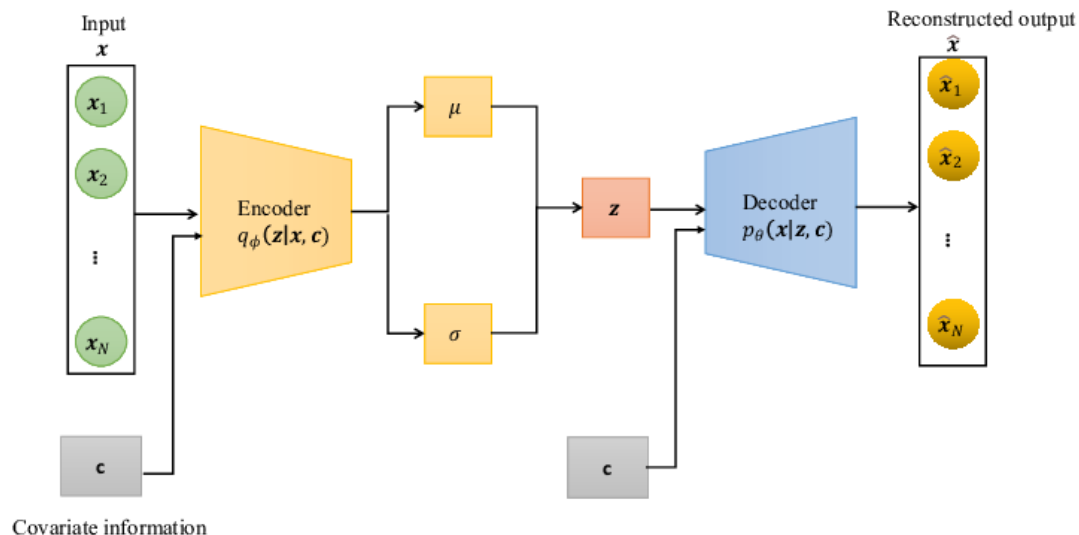


FIGURE 21 – Architecture d'un CVAE.

#### 7.3.2 Encoder

L'encodeur dans un CVAE prend à la fois les données d'entrée  $x$  et les conditions  $c$  comme input, et les transforme en une distribution latente conditionnelle  $q(z | x, c)$ . L'objectif est d'apprendre non seulement la distribution latente des données, mais aussi la manière dont elle varie en fonction des conditions spécifiques.

L'encodeur produit deux sorties :

$\mu(x, c)$  : le vecteur des moyennes conditionnelles,

$\sigma(x, c)$  : le vecteur des écarts types conditionnels (souvent log-variances pour la stabilité numérique).

Les variables latentes  $z$  sont échantillonnées à partir de cette distribution conditionnelle :

$$q(z | x, c) = \mathcal{N}(z; \mu(x, c), \sigma(x, c)^2).$$

### 7.3.3 L'échantillonnage

Comme dans les VAE, l'échantillonnage des variables latentes  $z$  à partir de la distribution conditionnelle  $q(z | x, c)$  utilise le trick de reparamétrisation pour rendre l'échantillonnage différentiable par rapport aux paramètres du modèle.

### 7.3.4 Décodeur

Le décodeur dans un CVAE prend les variables latentes  $z$  ainsi que les conditions  $c$  et les utilise pour reconstruire les données d'origine  $x$ . Le décodeur apprend la distribution conditionnelle  $p(x | z, c)$ , modélisée comme une distribution gaussienne ou de Bernoulli selon la nature des données.

Formellement, le décodeur peut être représenté comme :

$$p(x | z, c) = \mathcal{N}(x; \hat{\mu}(z, c), \hat{\sigma}(z, c)^2),$$

où  $\hat{\mu}(z, c)$  et  $\hat{\sigma}(z, c)$  sont les paramètres de la distribution reconstruite, prédits par le décodeur.

### 7.3.5 Fonction de perte

La fonction de perte pour un CVAE est similaire à celle d'un VAE, mais inclut également les conditions  $c$  lors du calcul de la divergence KL et de la reconstruction.

Erreur de reconstruction :

$$L_{reconstruction} = -\mathbb{E}_{q(z|x,c)}[\log p(x | z, c)],$$

Divergence de Kullback-Leibler (KL) :

$$L_{KL} = D_{KL}(q(z | x, c) || p(z)),$$

La fonction de perte totale est :

$$L_{CVAE} = L_{reconstruction} + L_{KL}.$$

### 7.3.6 Entraînement du CVAE

L'entraînement d'un CVAE implique l'optimisation de cette fonction de perte totale en utilisant des méthodes de gradient, souvent avec des algorithmes comme Adam. Les CVAE permettent de générer des données conditionnelles tout en conservant les avantages des VAE en termes de représentation latente et de génération de données réalistes.

### 7.3.7 Implémentation

#### Nettoyage des données et tokenization

##### 1) Nettoyage des données :

Le nettoyage des données textuelles implique plusieurs étapes pour préparer les données à l'analyse et à la modélisation. Cela comprend généralement :

- Suppression de la ponctuation et des caractères spéciaux : Élimination des signes de ponctuation tels que les points, virgules, guillemets, etc., ainsi que des caractères spéciaux comme les emojis ou les symboles non pertinents pour l'analyse.
- Normalisation des mots : Conversion des mots en minuscules pour assurer la cohérence et réduire la variabilité due à la casse.
- Gestion des espaces et des tokens : Gestion des espaces supplémentaires ou des tokens spéciaux qui pourraient perturber l'analyse.

##### 2) Tokenization :

La tokenization divise le texte en unités plus petites appelées tokens, qui peuvent être des mots, des sous-mots ou des caractères, selon le tokenizer utilisé. Pour les modèles basés sur BERT, qui est utilisé

dans notre cas, la tokenization se fait souvent de manière sub-word, ce qui signifie que certains mots peuvent être divisés en sous-parties si nécessaire pour une meilleure représentation linguistique.

## Tokenizer BERT

### 1) Utilisation du tokenizer BERT :

Le tokenizer BERT (Bidirectional Encoder Representations from Transformers) est une composante essentielle lorsqu'on travaille avec des modèles pré-entraînés comme BERT. Il assure que le texte d'entrée est converti en une séquence de tokens conformément à la configuration spécifiée dans le fichier `vocab.txt`. Cela garantit que les tokens générés sont cohérents avec ceux attendus par le modèle BERT pré-entraîné, assurant ainsi une interprétation précise du texte.

## Encodage des données

### 1) Sauvegarde et chargement des données tokenisées :

Après la tokenization, les données textuelles sont converties en tenseurs PyTorch pour faciliter le traitement et l'entraînement sur de grandes quantités de données. Les tenseurs sont sauvegardés dans des fichiers comme `train_input_ids.pt` et `train_attention_masks.pt`, qui stockent respectivement les IDs des tokens et les masques d'attention nécessaires pour le modèle BERT. Cette approche permet une gestion efficace des données et garantit la reproductibilité des expériences, car les mêmes données peuvent être chargées et utilisées à plusieurs reprises sans nécessiter de nouvelle tokenization.

## Modèle CVAE

Le modèle CVAE (Conditional Variational AutoEncoder) est une architecture complexe qui combine plusieurs composantes clés :

- **Encodeur (BertModel) :** Utilisé pour encoder le texte d'entrée et capturer ses représentations sémantiques. Le modèle BERT est particulièrement adapté pour cette tâche en raison de son architecture bidirectionnelle et de sa capacité à capturer les relations contextuelles entre les mots.
- **Réseau de reconnaissance :** Ce réseau apprend les paramètres statistiques de la distribution latente des variables, souvent représentées par  $\mu$  (moyenne) et  $\log\sigma$  (logarithme de la variance). Ces paramètres permettent de générer des échantillons de variables latentes, qui sont cruciaux pour la génération d'explications.
- **Générateur d'explications (G) :** Utilise une architecture basée sur les transformers pour générer des explications textuelles en réponse aux représentations latentes obtenues à partir de l'encodeur. Cette partie du modèle est essentielle pour produire des explications pertinentes et cohérentes.
- **Méthodes spécifiques :** Le modèle CVAE comprend également des méthodes pour encoder les données, reparamétriser les variables latentes pour l'échantillonnage et générer des explications basées sur les variables latentes produites.
- **Hyperparamètres :**

hidden_dim	latent_dim	num_labels	vocab_size	rating_classes	learning_rate
768	64	3	30522	9	1e-3

TABLE 4 – Liste des hyperparamètres utilisés pour l'implémentation de mon CVAE

Si nous avions eu plus de temps, nous aurions aimé utiliser la librairie Optuna pour optimiser les hyperparamètres, notamment le taux d'apprentissage et l'espace latent.

## Fonctions de perte

Plusieurs fonctions de perte sont utilisées pour optimiser et guider l'apprentissage du modèle CVAE :

- **explanation\_factor\_loss :** Cette fonction mesure la différence entre les étiquettes prédites pour les explications générées et les étiquettes réelles attendues. Elle permet d'évaluer la qualité et la pertinence des explications produites par le modèle.
- **minimum\_risk\_training\_loss :** Intègre la distance sémantique et les pertes spécifiques au CVAE pour optimiser l'entraînement du modèle. Elle garantit que les explications générées non seulement correspondent aux données d'entrée mais aussi respectent les contraintes et les objectifs définis par l'utilisateur.

- **combined\_loss\_function** : Cette fonction agrège différentes composantes de perte, telles que la génération d'explications, la divergence KL (Kullback-Leibler) et la perte MRT (Minimum Risk Training), pendant le processus d'entraînement. Elle est conçue pour maximiser la performance globale du modèle en prenant en compte plusieurs aspects de l'optimisation et de l'évaluation.

### 7.3.8 Résultats

Exemple d'un résultat après l'implémentation du CVAE :

Positive : The goal tree danced with polka-dotted elephants, singing jingle bells in the roaster ocean.  
Kitchen take top hat made of spaghetti and laughed at the moon's purple sneezes.

Les résultats obtenus révèlent que le modèle transformeur, agissant en tant qu'encodeur-décodeur, a réussi à générer un texte composé uniquement de mots valides du dictionnaire, démontrant ainsi sa capacité à produire du contenu linguistiquement cohérent.

En parallèle, pour optimiser le temps d'entraînement, j'ai réduit la taille du dataset de 60%, ce qui a considérablement accéléré le processus d'entraînement tout en maintenant des performances acceptables.

J'ai du diviser la taille de 60% pour optimiser le temps d'entrainement sur GPU. J'utilisais durant mon stage le NVIDIA A40 avec 46 GB de mémoire.

NVIDIA-SMI 535.104.05			Driver Version: 535.104.05			CUDA Version: 12.2		
GPU	Name	Persistence-M	Bus-Id	Disp.A	Volatile	Uncorr.	ECC	
Fan	Temp	Perf	Pwr:Usage/Cap	Memory-Usage	GPU-Util	Compute	M.	
0	NVIDIA A40	Off	00000000:37:00.0	Off	0			
0%	57C	P0	87W / 300W	43839MiB / 46068MiB	4%	Default	N/A	
1	NVIDIA A40	Off	00000000:86:00.0	Off	0			
0%	52C	P0	84W / 300W	16399MiB / 46068MiB	0%	Default	N/A	

FIGURE 22 – Extrait de code pour l'initialisation du classifieur pré-entraîné

À la lumière de ces observations, plusieurs hypothèses initiales peuvent être formulées :

- Augmenter les données d'entraînement permettrait au modèle d'apprendre plus efficacement les contextes et les structures linguistiques, conduisant potentiellement à une meilleure génération de texte.
- L'intégration d'un classifieur pré-entraîné serait envisageable pour améliorer la qualité de classification des phrases générées, assurant ainsi une meilleure pertinence et adéquation aux intentions de l'utilisateur.

## 7.4 GEF

Comme mentionné précédemment, nous avons gardé la même architecture de modèle, et l'avons entraîné sur l'intégralité du jeu de données, tout en intégrant un classifieur pré-entraîné.

### 7.4.1 Architecture (rajout du pretrained classifieur)

- Classifieur pré-entraîné :

Intégration du classifieur BERT : le modèle BERT utilisé, *BertForSequenceClassification*, est pré-entraîné pour la tâche spécifique de classification de séquences. Ce modèle est chargé à partir des poids pré-entraînés disponibles dans la bibliothèque transformers. Il est utilisé pour prédire les labels de sentiment (négatif, neutre, positif) en fonction des représentations encodées des données d'entrée. Cette capacité à intégrer un modèle pré-entraîné permet de bénéficier des connaissances et des capacités de généralisation acquises lors de l'entraînement sur de vastes ensembles de données.

### 7.4.2 Implémentation du classifieur

— Initialisation du classifieur pré-entraîné :

Tout d'abord, le modèle BertForSequenceClassification est chargé à partir des poids pré-entraînés disponibles dans la bibliothèque transformers. Ce modèle est spécifiquement conçu pour la classification de séquences, ce qui le rend approprié pour prédire les étiquettes de sentiment comme négatif, neutre ou positif à partir des représentations encodées des données d'entrée.

```
from transformers import BertTokenizer, BertModel, BertForSequenceClassification, BertConfig

model_name = 'bert-base-uncased'
config = BertConfig.from_pretrained(model_name, hidden_dropout_prob=0.5, num_labels=9)
model = BertForSequenceClassification.from_pretrained(model_name, config=config).to(device)

model.load_state_dict(torch.load('best_bert_classifier.pth'))

#Load the pretrained classifier weights
pretrained_classifier = BertForSequenceClassification.from_pretrained('bert-base-uncased', num_labels=9).to(device)
pretrained_classifier.load_state_dict(torch.load('best_bert_classifier.pth'))
pretrained_classifier.eval()
```

FIGURE 23 – Extrait de code pour l'initialisation du classifieur pré-entraîné.

— Utilisation dans le modèle CVAE :

Une fois chargé, le classifieur pré-entraîné BertForSequenceClassification est intégré dans le modèle CVAE de plusieurs manières :

Extraction de représentations :

Lors de l'encodage des données textuelles par l'encodeur BertModel dans le CVAE, les représentations intermédiaires générées sont utilisées comme entrée pour le classifieur. Cela permet d'obtenir des prédictions de classification basées sur les mêmes embeddings utilisés pour la génération d'explications.

```
def classify_explanation(self, explanation_ids, attention_mask):
    explanation_ids = explanation_ids.to(self.pretrained_classifier.device)
    attention_mask = attention_mask.to(self.pretrained_classifier.device)
    outputs = self.pretrained_classifier(explanation_ids, attention_mask=attention_mask)
    return F.softmax(outputs.logits, dim=1)
```

FIGURE 24 – Extrait du code pour la classification explications.

### Fonction de perte :

La fonction de perte combinée (`combined_loss_function`) intègre à la fois la génération de texte et la classification. Cela est crucial dans les tâches où la pertinence des explications générées par rapport à un label de sentiment prédit est importante. On utilise les prédictions du classifieur (`class_logits`) pour calculer la perte de classification (`classification_loss`). Cela assure que les explications générées sont cohérentes avec les prédictions de sentiment faites par le modèle.

```
def combined_loss_function(P_classified, P_gold, P_pred, P_true, mu, logvar, generative_logits, input_ids, train_loader, device):
    # Explanation generation loss (CrossEntropyLoss)
    explanation_loss = nn.CrossEntropyLoss()(generative_logits.view(-1, generative_logits.size(-1)), input_ids.view(-1))

    # Minimum risk training loss
    mrt_loss = minimum_risk_training_loss(P_classified, P_gold, train_loader, semantic_distance, cvae, device)

    # KL divergence loss
    kld_loss = -0.5 * torch.sum(1 + logvar - mu.pow(2) - logvar.exp()) / mu.size(0)

    # CVAE loss
    cvae_loss = explanation_loss + kld_loss

    # Final loss
    final_loss = cvae_loss + mrt_loss

    return final_loss
```

FIGURE 25 – Extrait du code pour la loss combinée.

### Entraînement conjoint :

Pendant l’entraînement du CVAE, les poids du classifieur (`BertForSequenceClassification`) sont ajustés simultanément avec les autres parties du modèle. Cela permet au modèle d’apprendre à générer des explications tout en améliorant la classification, ce qui est bénéfique dans les applications où les deux tâches sont interdépendantes.

### Évaluation et interprétation :

Une fois le modèle CVAE entraîné, le classifieur pré-entraîné peut être utilisé pour évaluer les performances du modèle global sur des données de test indépendantes.

### 7.4.3 Résultats

Le fait d’avoir été contraint d’utiliser l’ensemble de données complet et d’intégrer un classifieur pré-entraîné a entraîné une augmentation significative du temps d’entraînement nécessaire.

Training Epoch 1: 5%	139/2577	[14:00:47<245:56:27, 363.16s/
Training Epoch 1: 5%	140/2577	[14:06:51<245:59:11, 363.38s/
Training Epoch 1: 5%	141/2577	[14:12:55<245:56:13, 363.45s/
Training Epoch 1: 6%	144/2577	[14:31:04<245:26:32, 363.17s/
Training Epoch 1: 6%	145/2577	[14:37:08<245:25:30, 363.29s/
Training Epoch 1: 6%	146/2577	[14:43:12<245:25:56, 363.45s/
Training Epoch 1: 6%	147/2577	[14:49:15<245:22:12, 363.51s/
Training Epoch 1: 6%	148/2577	[14:55:19<245:23:54, 363.70s/

FIGURE 26 – Aperçu du temps d’entraînement pour 1 epoch.

Durant notre étude, nous avons constaté que les délais d’exécution élevés ont considérablement entravé l’obtention de résultats dans les délais impartis. Cette observation met en lumière plusieurs limitations, tant dans le cadre de notre implémentation que dans les travaux précédemment publiés sur le sujet. Ces contraintes soulèvent des questions importantes qui nécessitent une attention particulière dans les recherches futures.

## 8 Limites

L'article implementé a présenté plusieurs limites :

### — Absence de détails pratiques

L'article se concentre principalement sur les aspects théoriques, en négligeant les étapes pratiques telles que le prétraitement des données et la mise en œuvre du code. Cette absence de détails pratiques rend la reproduction des résultats difficile et limite la compréhension complète de l'approche proposée. Des informations plus détaillées sur les étapes de prétraitement et d'implémentation auraient permis une évaluation plus approfondie de l'efficacité de la méthode proposée.

### — Manque de disponibilité du code source

L'absence de code source disponible sur des plateformes comme GitHub restreint la transparence et la reproductibilité des résultats présentés dans l'article. Sans accès au code source, il est difficile d'évaluer les choix de conception et d'optimisation effectués lors de la mise en œuvre de l'approche proposée.

Le manque de ces détails complique l'évaluation de l'efficacité de la méthode et sa comparaison avec d'autres approches similaires dans le domaine.

Malgré mes tentatives pour contacter les auteurs afin d'obtenir plus d'informations, je n'ai reçu aucune réponse.

## 9 Bilan et perspectives

### 9.1 Etat de l’art

Le domaine de l’explicabilité des algorithmes est un domaine encore jeune. Si des outils performants ont été développés pour les algorithmes de données tabulaires, les autres modalités disposent de peu d’outils spécifiques, et l’adaptation des méthodes tabulaires restent rudimentaires.

De nombreuses bibliothèques implémentent des outils d’explicabilité, mais ces outils sont difficilement interprétable (Beeswarm plot...). De plus, lorsqu’ils sont mis à disposition, les codes associés aux méthodes introduites dans les articles sont le plus souvent stockés dans des répertoires plus ou moins structurés par les auteurs. Les bibliothèques proposant une API normalisées pour l’utilisation de plusieurs méthodes dépendent généralement de ces répertoires non maintenus (cas de LIME notamment).

La question de l’explicabilité pour des algorithmes de données multimodales est à ce jour peu étudiée, et principalement circonscrite au couple de modalités texte et image (algorithmes de *Visual Question Answering* notamment). Très peu de travaux traitant de l’utilisation de données structurées ont été trouvés. Dans ce domaine, peu de sources traitant de l’utilisation de données textuelles et tabulaires ont été trouvés.

Par ailleurs, pour EDF, il y a un réel enjeu à disposer d’explications claires et compréhensibles des modèles d’IA multimodaux pour les utilisateurs non-expert. L’intérêt d’approfondir les techniques permettant de générer des explications en langage naturel est alors non négligeable.

Après avoir complété l’état de l’art, nous avons constaté que les méthodes basées sur les caractéristiques (features based methods) présentaient certaines limites que le NLE (Natural Language Explanation) pouvait compenser, telles que le manque de lisibilité des explications pour les utilisateurs finaux. Liu et al. [20] ont proposé une méthode qui permet non seulement de classer les données, mais aussi de générer des justifications grâce à l’utilisation d’un CVAE (Conditional Variational Autoencoder).

Nous avons ensuite implémenté cette méthode. De par l’ancienneté de l’article (2019), nous avons dû remplacer certaines techniques utilisées par les auteurs dans le CVAE par des méthodes plus récentes et plus performantes. Nous avons ainsi remplacé le LSTM par une méthode basée sur un transformer, et opté pour un embedding par BERT plutôt que par GloVe.

Enfin, nous avons identifié des perspectives qui pourraient permettre d’améliorer le modèle et d’envisager d’autres cas d’utilisation.

### 9.2 Perspectives

#### Utilisation potentielle d’un LLM à la place d’un CVAE

Bien que nous ayons choisi d’utiliser un CVAE dans notre étude pour rester fidèles à l’article original, nous pourrions envisager d’explorer l’utilisation d’un modèle de langage de type LLM à l’avenir. Les LLM ont démontré leur efficacité et leur flexibilité dans la génération et la compréhension de texte, ce qui pourrait rendre notre approche plus efficace. En particulier, l’utilisation d’un LLM pourrait permettre une meilleure modélisation des dépendances à long terme dans les données textuelles, ce qui pourrait se traduire par une amélioration de la qualité des résultats générés.

#### Utilisation de LLM pour interroger les modèles

En plus d’utiliser des LLM pour améliorer nos performances, nous prévoyons également d’explorer l’utilisation de LLM pour interroger les modèles et comprendre leur fonctionnement interne. Cette approche pourrait nous permettre de mieux comprendre les décisions prises par les modèles et d’identifier les éventuels biais ou erreurs dans les résultats générés. En particulier, l’utilisation de LLM pour interroger les modèles pourrait aider à améliorer la transparence et l’interprétabilité de l’approche, ce qui est essentiel pour une utilisation responsable et éthique de l’IA.

#### Évaluation sur des données EDF et d’autres jeux de données

Afin d’évaluer la généralisation de notre approche, il sera opportun de tester notre méthode sur d’autres jeux de données Open Data tels que les commentaires de produits Amazon, mais surtout sur cas d’usage EDF avec des données EDF. L’utilisation de différents jeux de données permettra d’évaluer la robustesse de notre approche dans différents contextes et de comparer ses performances à celles d’autres méthodes



similaires. En particulier, l'utilisation de données EDF permettra d'évaluer l'efficacité de notre approche dans un contexte industriel réel, où les données peuvent être bruyantes et incomplètes.

## 10 Annexe

### 10.1 Few-Shot Self-Rationalization with Natural Language Prompts

Les auteurs ont cherché à résoudre deux problématiques liées aux modèles de "self rationalization" qui génèrent des prédictions et des justifications en langage naturel [23]. La première problématique est la nécessité d'un grand nombre de données annotées pour entraîner ces modèles, ce qui peut être coûteux en temps et en ressources. La deuxième problématique est de savoir s'il existe une corrélation entre le nombre de paramètres du modèle et la qualité des explications générées.

Ce qui nous intéresse tout particulièrement ici est la première problématique.

Pour aborder le "few shot self rationalization", les auteurs ont utilisé une approche de fine-tuning basée sur des prompts en langage naturel (NL prompts) avec le modèle T5. Des éléments sur le modèle T5 sont présentés en partie 5.2.3. Les NL prompts sont des phrases ou des questions en langage naturel qui guident le modèle dans la génération de réponses appropriées. Les auteurs ont démontré que l'utilisation de NL prompts permettait de généraliser les modèles de "self rationalization" à des tâches similaires avec peu de données d'entraînement.

Par exemple, dans la figure ci-dessous, les NL prompts sont utilisés pour guider le modèle dans la génération de justifications pour une tâche de classification de sentiments.

<p><b>Sentence1:</b> The stove was cleaned with a cleaner. <b>Sentence2:</b> The stove was cleaned with a mop. <b>Nonsensical Sentence:</b> Sentence2 <b>Explanation:</b> A mop is too large to clean the stove.</p> <hr/> <p><b>Prompt:</b> INFILLING x BASIC <b>Input:</b> explain sensemaking choice1: The stove was cleaned with a cleaner. choice2: The stove was cleaned with a mop. &lt;extra_id_0&gt; because &lt;extra_id_1&gt; <b>Output:</b> &lt;extra_id_0&gt; choice2 &lt;extra_id_1&gt; A mop is too large to clean the stove. &lt;extra_id_2&gt;</p> <hr/> <p><b>Prompt:</b> INFILLING x NATURAL SOUNDING <b>Input:</b> explain sensemaking choice1: The stove was cleaned with a cleaner. choice2: The stove was cleaned with a mop. It is &lt;extra_id_0&gt; that choice2 is less common because &lt;extra_id_1&gt; <b>Output:</b> &lt;extra_id_0&gt; True &lt;extra_id_1&gt; A mop is too large to clean the stove. &lt;extra_id_2&gt;</p> <hr/> <p><b>Prompt:</b> ≈T5 x COPA <b>Input:</b> explain sensemaking choice1: The stove was cleaned with a cleaner. choice2: The stove was cleaned with a mop. Less common is choice2 <b>Output:</b> True because a mop is too large to clean the stove.</p> <hr/> <p><b>Prompt:</b> SQUADIS x YES/NO + TAGS <b>Input:</b> explain sensemaking question: Is choice2 more nonsensical? context: choice1: The stove was cleaned with a cleaner. choice2: The stove was cleaned with a mop. <b>Output:</b> Yes because a mop is too large to clean the stove.</p> <hr/> <p><b>Prompt:</b> SQUADIS x WHAT IS...? + TAGS <b>Input:</b> explain sensemaking question: What is more nonsensical? context: choice1: The stove was cleaned with a cleaner. choice2: The stove was cleaned with a mop. <b>Output:</b> choice2 because a mop is too large to clean the stove.</p> <hr/> <p><b>Prompt:</b> QASIMPLE x YES/NO <b>Input:</b> explain is choice2 more nonsensical? \n The stove was cleaned with a cleaner. The stove was cleaned with a mop.&lt;/&gt; <b>Output:</b> yes because a mop is too large to clean the stove.</p>	<p><b>1.INFILLING x BASIC:</b></p> <ol style="list-style-type: none"><li>1. Input asks the model to choose which sentence makes more sense.</li><li>2. The output is a choice along with an explanation that "A mop is too large to clean the stove," suggesting the second sentence is nonsensical.</li></ol> <p><b>2.INFILLING x NATURAL SOUNDING:</b></p> <ol style="list-style-type: none"><li>1. The model is prompted in a similar way but is likely prompted to choose the more natural-sounding sentence.</li><li>2. The response indicates that the first sentence is natural because the second sentence doesn't make sense, repeating the explanation from the basic prompt.</li></ol> <p><b>3.t5 x COPA:</b></p> <ol style="list-style-type: none"><li>1. This refers to the model being prompted in the style of the Choice of Plausible Alternatives (COPA) format, a task for assessing causal understanding.</li><li>2. The output is simply "True" with an explanation similar to the previous ones, implying that it's true that the second option is less sensible.</li></ol> <p><b>4.SQUAD x YES/NO + TAGS:</b></p> <ol style="list-style-type: none"><li>1. Here the model is using a format inspired by the Stanford Question Answering Dataset, but the task is to answer yes or no to a question about the nonsensical nature of one of the choices.</li><li>2. The response is "Yes" and reiterates that a mop is too large to clean the stove.</li></ol> <p><b>5.SQUAD x WHAT IS...? + TAGS:</b></p> <ol style="list-style-type: none"><li>1. This prompt appears to be asking the model to evaluate which sentence is more nonsensical, framed in a "What is" question.</li><li>2. The model's output is again pointing out the nonsensical nature of using a mop to clean a stove.</li></ol> <p><b>6.QASimple x YES/NO:</b></p> <ol style="list-style-type: none"><li>1. Similar to the SQUAD style but with a different pretraining model</li><li>2. The input is a yes/no question regarding which sentence is more nonsensical, and the output is "yes," followed by the familiar explanation.</li></ol>
--	---

FIGURE 27 – Exemple de NL prompts pour la génération de justifications

Les auteurs ont proposé une approche efficace pour le "few shot self rationalization" en utilisant des NL prompts avec le modèle T5, ce qui permet de généraliser les modèles à des tâches similaires avec peu de données d'entraînement.

## Table des figures

1	Illustration des principales modalités. . . . .	7
2	Schéma des étapes de traitement automatisé du langage. . . . .	7
3	Architecture de l'entraînement des modèles « continuous bag of words » (CBOW) et « skip-gram ». . . . .	8
4	Exemple de projection d'une représentation word2vec. . . . .	9
5	Graphique synthétique d'un modèle d'analyse de sentiment basé sur BERT. . . . .	9
6	Pré-entraînement et entraînement du modèle BERT. . . . .	10
7	Exemples de taxonomies des méthodes d'interprétabilité . . . . .	11
8	Illustration de LIME. . . . .	12
9	Exemple de l'utilisation de LIME pour de l'analyse de sentiments. . . . .	12
10	Illustration du graphique en barres. . . . .	13
11	Illustration du Beeswarm Plot. . . . .	14
12	Illustration du graphique de dépendance. . . . .	14
13	GradCAM architecture . . . . .	17
14	Un exemple de question à choix multiples inspiré du WorldTree V2. . . . .	18
15	Un exemple de problème résolu avec la justification de la réponse ("rationale"). . . . .	19
16	Aperçu du dataset PCMAG. . . . .	25
17	Fréquence des notes globales ("overall ratings"). . . . .	25
18	Architecture du Generative Explanation Framework . . . . .	26
19	Aperçu du dataset PCMAG preprocessed. . . . .	28
20	Architecture d'un VAE. . . . .	29
21	Architecture d'un CVAE. . . . .	32
22	Extrait de code pour l'initialisation du classifieur pré-entraîné . . . . .	35
23	Extrait de code pour l'initialisation du classifieur pré-entraîné. . . . .	36
24	Extrait du code pour la classification explications. . . . .	36
25	Extrait du code pour la loss combinée. . . . .	37
26	Aperçu du temps d'entraînement pour 1 epoch. . . . .	37
27	Exemple de NL prompts pour la génération de justifications . . . . .	41

## Liste des tableaux

1	Listes des papiers pertinents sur l'explicabilité . . . . .	15
2	Liste des papiers pertinents pour la modélisation sur données multimodales . . . . .	16
3	Papier sur l'explicabilité multimodale, ayant dirigé les travaux vers le NLE . . . . .	16
4	Liste des hyperparamètres utilisés pour l'implémentation de mon CVAE . . . . .	34

## Références

- [1] Tariq ALHINDI, Savvas PETRIDIS et Smaranda MURESAN. “Where is your evidence : Improving fact-checking by justification modeling”. In : *Proceedings of the first workshop on fact extraction and verification (FEVER)*. 2018, p. 85-90.
- [2] Razieh BARADARAN, Razieh GHIASI et Hossein AMIRKHANI. “A Survey on Machine Reading Comprehension Systems”. In : *Natural Language Engineering* 28.6 (2022), p. 683-732. DOI : 10.1017/S1351324921000395.
- [3] Samuel R BOWMAN et al. “A large annotated corpus for learning natural language inference”. In : *arXiv preprint arXiv :1508.05326* (2015).
- [4] Oana-Maria CAMBURU et al. “e-snli : Natural language inference with natural language explanations”. In : *Advances in Neural Information Processing Systems* 31 (2018).
- [5] Shiyu CHANG et al. “A Game Theoretic Approach to Class-wise Selective Rationalization”. In : *Advances in Neural Information Processing Systems*. Sous la dir. de H. WALLACH et al. T. 32. Curran Associates, Inc., 2019. URL : [https://proceedings.neurips.cc/paper\\_files/paper/2019/file/5ad742cd15633b26fdce1b80f7b39f7c-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2019/file/5ad742cd15633b26fdce1b80f7b39f7c-Paper.pdf).
- [6] Shiyu CHANG et al. “Invariant Rationalization”. In : *Proceedings of the 37th International Conference on Machine Learning*. Sous la dir. d’Hal Daumé III et Aarti SINGH. T. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, p. 1448-1458. URL : <https://proceedings.mlr.press/v119/chang20c.html>.
- [7] Wayne CHRISTENSEN et John MICHAEL. “Ian Apperly, Mindreaders : the cognitive basis of theory of mind”. In : *Phenomenology and the Cognitive Sciences* 12 (déc. 2013). DOI : 10.1007/s11097-012-9292-9.
- [8] Marina DANILEVSKY et al. “A Survey of the State of Explainable AI for Natural Language Processing”. In : *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*. Sous la dir. de Kam-Fai WONG, Kevin KNIGHT et Hua WU. Suzhou, China : Association for Computational Linguistics, déc. 2020, p. 447-459. URL : <https://aclanthology.org/2020.aacl-mai-n.46>.
- [9] Devleena DAS et Sonia CHERNOVA. “Leveraging rationales to improve human task performance”. In : *Proceedings of the 25th International Conference on Intelligent User Interfaces*. 2020, p. 510-518.
- [10] Jacob DEVLIN et al. “BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding”. In : (juin 2019). Sous la dir. de Jill BURSTEIN, Christy DORAN et Thamar SOLORIO, p. 4171-4186. DOI : 10.18653/v1/N19-1423. URL : <https://aclanthology.org/N19-1423>.
- [11] Jay DEYOUNG et al. “ERASER : A Benchmark to Evaluate Rationalized NLP Models”. In : *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Sous la dir. de Dan JURAFSKY et al. Online : Association for Computational Linguistics, juill. 2020, p. 4443-4458. DOI : 10.18653/v1/2020.acl-main.408. URL : <https://aclanthology.org/2020.acl-main.408>.
- [12] Mengnan DU et al. “Learning credible deep neural networks with rationale regularization”. In : *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2019, p. 150-159.
- [13] Leo GRINSZTAJN, Edouard OYALLON et Gael VAROQUAUX. “Why do tree-based models still outperform deep learning on typical tabular data?” In : 35 (2021). Sous la dir. de S. KOYEJO et al., p. 507-520. URL : [https://proceedings.neurips.cc/paper\\_files/paper/2022/file/0378c7692da36807bd ec87ab043cdadc-Paper-Datasets\\_and\\_Benchmarks.pdf](https://proceedings.neurips.cc/paper_files/paper/2022/file/0378c7692da36807bd ec87ab043cdadc-Paper-Datasets_and_Benchmarks.pdf).
- [14] Sawan KUMAR et Partha TALUKDAR. “NILE : Natural language inference with faithful natural language explanations”. In : *arXiv preprint arXiv :2005.12116* (2020).
- [15] Kushal LAKHOTIA et al. “FiD-Ex : Improving Sequence-to-Sequence Models for Extractive Rationale Generation”. In : *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Sous la dir. de Marie-Francine MOENS et al. Online et Punta Cana, Dominican Republic : Association for Computational Linguistics, nov. 2021, p. 3712-3727. DOI : 10.18653/v1/2021.emnlp-main.301. URL : <https://aclanthology.org/2021.emnlp-main.301>.
- [16] Tao LEI, Regina BARZILAY et Tommi JAAKKOLA. “Rationalizing neural predictions”. In : *arXiv preprint arXiv :1606.04155* (2016).

- [17] Shuyang LI, Bodhisattwa Prasad MAJUMDER et Julian MCAULEY. “Self-supervised bot play for conversational recommendation with justifications”. In : *arXiv preprint arXiv :2112.05197* (2021).
- [18] Wang LING et al. “Program Induction by Rationale Generation : Learning to Solve and Explain Algebraic Word Problems”. In : *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*. Sous la dir. de Regina BARZILAY et Min-Yen KAN. Vancouver, Canada : Association for Computational Linguistics, juill. 2017, p. 158-167. DOI : 10.18653/v1/P17-1015. URL : <https://aclanthology.org/P17-1015>.
- [19] Wang LING et al. “Program induction by rationale generation : Learning to solve and explain algebraic word problems”. In : *arXiv preprint arXiv :1705.04146* (2017).
- [20] Hui LIU, Qingyu YIN et William Yang WANG. “Towards explainable NLP : A generative explanation framework for text classification”. In : *arXiv preprint arXiv :1811.00196* (2019).
- [21] Scott M LUNDBERG et Su-In LEE. “A unified approach to interpreting model predictions”. In : *Advances in neural information processing systems* 30 (2017).
- [22] Bill MACCARTNEY et Christopher D MANNING. “Natural logic and natural language inference”. In : *Computing Meaning : Volume 4*. Springer, 2014, p. 129-147.
- [23] Ana MARASOVIĆ et al. “Few-shot self-rationalization with natural language prompts”. In : *arXiv preprint arXiv :2111.08284* (2021).
- [24] Todor MIHAYLOV et al. “Can a Suit of Armor Conduct Electricity ? A New Dataset for Open Book Question Answering”. In : *CoRR* abs/1809.02789 (2018). arXiv : 1809.02789. URL : <http://arxiv.org/abs/1809.02789>.
- [25] Tomas MIKOLOV et al. “Distributed representations of words and phrases and their compositionality”. In : *Advances in neural information processing systems* 26 (2013).
- [26] Tomas MIKOLOV et al. “Efficient estimation of word representations in vector space”. In : *arXiv preprint arXiv :1301.3781* (2013).
- [27] Tim MILLER. “Explanation in artificial intelligence : Insights from the social sciences”. In : *Artificial intelligence* 267 (2019), p. 1-38.
- [28] Shervin MINAEI et al. “Deep learning-based text classification : a comprehensive review”. In : *ACM computing surveys (CSUR)* 54.3 (2021), p. 1-40.
- [29] Christoph MOLNAR. *Interpretable machine learning*. Lulu. com, 2020.
- [30] Sharan NARANG et al. “Wt5 ?! training text-to-text models to explain their predictions”. In : *arXiv preprint arXiv :2004.14546* (2020).
- [31] Kishore PAPINENI et al. “Bleu : a method for automatic evaluation of machine translation”. In : *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002, p. 311-318.
- [32] Mitchell PLYLER, Michael GREEN et Min CHI. “Making a (counterfactual) difference one rationale at a time”. In : *Advances in Neural Information Processing Systems* 34 (2021), p. 28701-28713.
- [33] Nazneen Fatema RAJANI et al. “Explain Yourself! Leveraging Language Models for Commonsense Reasoning”. In : *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Sous la dir. d’Anna KORHONEN, David TRAUM et Lluís MÀRQUEZ. Florence, Italy : Association for Computational Linguistics, juill. 2019, p. 4932-4942. DOI : 10.18653/v1/P19-1487. URL : <https://aclanthology.org/P19-1487>.
- [34] Yanou RAMON et al. “A comparison of instance-level counterfactual explanation algorithms for behavioral and textual data : SEDC, LIME-C and SHAP-C”. In : *Advances in Data Analysis and Classification* (sept. 2020). DOI : 10.1007/s11634-020-00418-3.
- [35] Ashish RANA et al. “RerrFact : Reduced Evidence Retrieval Representations for Scientific Claim Verification”. In : *arXiv preprint arXiv :2202.02646* (2022).
- [36] Marco Tulio RIBEIRO, Sameer SINGH et Carlos GUESTRIN. “" Why should i trust you ?" Explaining the predictions of any classifier”. In : *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, p. 1135-1144.
- [37] Andrew Slavin ROSS, Michael C. HUGHES et Finale DOSHI-VELEZ. “Right for the Right Reasons : Training Differentiable Models by Constraining their Explanations”. In : *CoRR* abs/1703.03717 (2017). arXiv : 1703.03717. URL : <http://arxiv.org/abs/1703.03717>.

- [38] Maarten SAP et al. “Commonsense Reasoning for Natural Language Processing”. In : *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics : Tutorial Abstracts*. Sous la dir. d’Agata SAVARY et Yue ZHANG. Online : Association for Computational Linguistics, juill. 2020, p. 27-33. DOI : 10.18653/v1/2020.acl-tutorials.7. URL : <https://aclanthology.org/2020.acl-tutorials.7>.
- [39] Lloyd S SHAPLEY et al. “A value for n-person games”. In : (1953).
- [40] Xingjian SHI et al. “Benchmarking Multimodal AutoML for Tabular Data with Text Fields”. In : *CoRR* abs/2111.02705 (2021). arXiv : 2111.02705. URL : <https://arxiv.org/abs/2111.02705>.
- [41] Alon TALMOR et al. “Commonsenseqa : A question answering challenge targeting commonsense knowledge”. In : *arXiv preprint arXiv :1811.00937* (2018).
- [42] Ashish VASWANI et al. “Attention is all you need”. In : *Advances in neural information processing systems* 30 (2017).
- [43] Sarah WIEGREFFE, Ana MARASOVIĆ et Noah A SMITH. “Measuring association between labels and free-text rationales”. In : *arXiv preprint arXiv :2010.12762* (2020).
- [44] Zhengnan XIE et al. “WorldTree V2 : A Corpus of Science-Domain Structured Explanations and Inference Patterns supporting Multi-Hop Inference”. In : *Proceedings of the Twelfth Language Resources and Evaluation Conference*. Sous la dir. de Nicoletta CALZOLARI et al. Marseille, France : European Language Resources Association, mai 2020, p. 5456-5473. URL : <https://aclanthology.org/2020.lrec-1.671>.
- [45] Mo YU et al. “Rethinking Cooperative Rationalization : Introspective Extraction and Complement Control”. In : *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Sous la dir. de Kentaro INUI et al. Hong Kong, China : Association for Computational Linguistics, nov. 2019, p. 4094-4103. DOI : 10.18653/v1/D19-1420. URL : <https://aclanthology.org/D19-1420>.
- [46] Omar ZAIDAN, Jason EISNER et Christine PIATKO. “Using “annotator rationales” to improve machine learning for text categorization”. In : *Human language technologies 2007 : The conference of the North American chapter of the association for computational linguistics ; proceedings of the main conference*. 2007, p. 260-267.
- [47] Ye ZHANG, Iain MARSHALL et Byron C WALLACE. “Rationale-augmented convolutional neural networks for text classification”. In : *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*. T. 2016. NIH Public Access. 2016, p. 795.