UNIVERSITÉ PARIS 1
PANTHÉON SORBONNE

Détection de l'humour dans le langage naturel par deep learning

Chirine Dexposito
Hella Bouhadda

Contents

1 Introduction

2 Introduction

L'humour est une composante essentielle de la communication humaine, omniprésente dans les interactions sociales, les médias et les contenus numériques. Pourtant, il demeure l'un des aspects les plus complexes à modéliser pour les machines. Loin d'être uniquement une affaire de mots, l'humour repose sur une alchimie subtile de langage, de contexte, de références culturelles et de pragmatique implicite. Cette richesse en fait un terrain de jeu fascinant pour la linguistique computationnelle et l'intelligence artificielle.
Dans ce projet, nous proposons de concevoir un système capable de détecter automatiquement l'humour dans des textes courts en anglais. Ce défi implique bien plus que de simples traitements lexicaux : il s'agit de capturer des phénomènes comme le sarcasme, l'ironie, les jeux de mots ou encore la surprise, autant d'éléments souvent absents ou ambigus pour les modèles traditionnels de NLP.
Notre démarche s'inscrit dans une double perspective :

- **Scientifique** : mieux comprendre les mécanismes linguistiques et cognitifs sous-jacents à la perception de l'humour.
- **Technique** : expérimenter, comparer et enrichir des architectures de classification textuelle allant des modèles classiques (SVM, TF-IDF) aux modèles de pointe tels que BERT ou DistilBERT, et explorer des approches innovantes comme l'interprétabilité locale, la détection de la surprise, ou le *clustering* des types d'humour.

À travers une évaluation rigoureuse (quantitative et qualitative) et une ouverture vers des pistes de recherche avancées (zero-shot learning, prompt tuning, etc.), ce projet vise à apporter une contribution à la fois appliquée et exploratoire à la modélisation de l'humour en NLP.

3 Problématique académique

La détection automatique de l'humour constitue un défi fondamental en traitement du langage naturel (NLP) en raison de sa complexité multidimensionnelle. Contrairement à d'autres tâches de classification textuelle plus lexicalement marquées, comme l'analyse de sentiment ou la détection de propos toxiques, l'humour s'ancre dans des mécanismes cognitifs, culturels et linguistiques subtils, souvent implicites. Il ne s'agit pas simplement de repérer des mots ou expressions clés, mais de comprendre des phénomènes profonds tels que l'ambiguïté sémantique, la rupture d'attente, les références socioculturelles ou encore l'ironie.
L'ambiguïté sémantique, souvent exploitée dans les jeux de mots, nécessite une capacité à interpréter plusieurs lectures d'un même énoncé. La théorie de l'incongruité, quant à elle, postule que le rire émerge d'un contraste ou d'un retournement inattendu de situation — un élément particulièrement difficile à formaliser algorithmiquement. Par ailleurs, nombre de blagues sont enracinées dans des contextes culturels précis : sans ces connaissances implicites, même un humain pourrait passer à côté du sens comique. Enfin, l'ironie et le sarcasme exigent une lecture entre les lignes, un décodage des intentions qui dépasse largement les capacités des modèles classiques.

Les approches traditionnelles fondées sur des représentations statistiques telles que les sacs de mots (*bag-of-words*) ou les n-grammes, couplées à des classifieurs linéaires comme la régression logistique ou les SVM, peinent à saisir la profondeur contextuelle requise pour ce type de tâche. Même les modèles neuronaux récents, bien qu'entraînés sur de vastes corpus, ne sont pas naturellement adaptés à la spécificité des constructions humoristiques, souvent marginales dans les données d'entraînement générales.
Dans ce contexte, nous formulons l'hypothèse que la performance des modèles de détection de l'humour pourrait être significativement améliorée par une intégration explicite de dimensions sémantiques, pragmatiques et discursives. Cela inclut notamment la modélisation de l'intention communicative (ironie, sarcasme), la prise en compte de structures narratives spécifiques (du type *setup* $\rightarrow$ *punchline*), ou encore l'exploitation de mécanismes d'interprétabilité permettant d'identifier les marqueurs humoristiques au sein du texte.
Ce projet s'inscrit ainsi dans une volonté de dépasser les limites des approches traditionnelles en enrichissant les modèles avec des signaux cognitifs, culturels et contextuels — ouvrant la voie à une modélisation plus fine et plus réaliste de l'humour dans les systèmes de NLP.

4 Exploration du dataset

Le corpus utilisé dans ce projet provient du dépôt GitHub de Moradnejad [1], lequel contient un ensemble de textes courts annotés selon leur caractère humoristique ou non. Ces données, rédigées en anglais, sont issues de diverses plateformes en ligne, et offrent un large éventail de formulations linguistiques propices à l'analyse humoristique automatique.

Répartition des classes

Une première analyse descriptive a été menée sur la variable cible afin d'évaluer la distribution des classes au sein du corpus. Comme le montre la Figure 1, les textes sont répartis en deux catégories : *humoristiques* et *non humoristiques*.
L'analyse révèle une distribution relativement équilibrée entre les deux classes, ce qui est favorable pour l'apprentissage supervisé. En effet, un fort déséquilibre aurait pu biaiser l'entraînement du modèle en faveur de la classe majoritaire.

Longueur des textes

La longueur des textes, mesurée en nombre de mots, a été calculée pour chaque exemple, puis comparée selon l'appartenance aux deux classes. La Figure 2 présente la distribution de cette variable, représentée sous forme d'un histogramme avec lissage par noyau (KDE). On observe que les textes humoristiques sont en moyenne légèrement plus courts que les textes non humoristiques. Cette concision pourrait refléter une volonté de maximiser l'effet comique en concentrant la formulation autour d'un *punchline* rapide.

[1] `https://github.com/Moradnejad/ColBERT-Using-BERT-Sentence-Embedding-for-Humor-Detection/tree/master/Data`

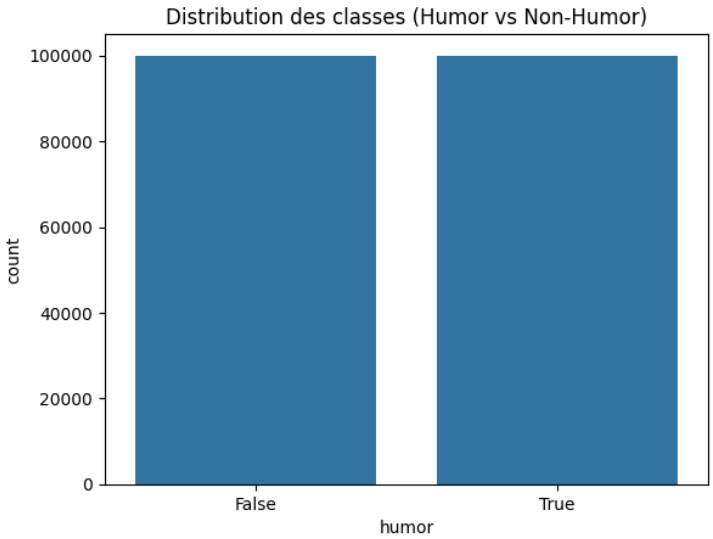

Figure 1: Distribution des classes (humoristique vs non humoristique)

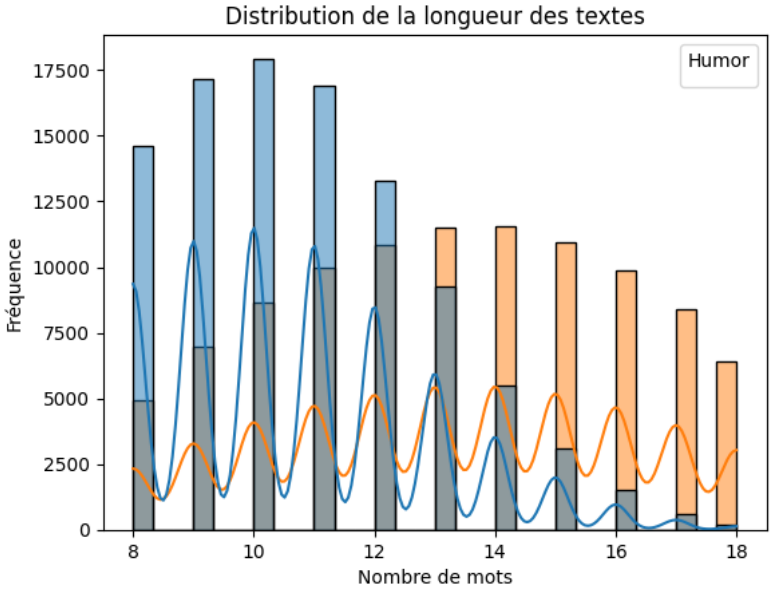

Figure 2: Distribution de la longueur des textes selon la classe

Analyse lexicale

Une analyse lexicale a été conduite à l'aide de nuages de mots, générés séparément pour les deux classes après nettoyage des textes (suppression des stopwords, mise en minuscules, etc.). Les Figures 3 et 4 présentent les mots les plus fréquents pour chaque catégorie.
Cette visualisation permet de constater des différences nettes dans les champs lexicaux utilisés. Les textes humoristiques présentent une plus grande fréquence de termes expressifs ou informels, tandis que les textes non humoristiques se caractérisent par un vocabulaire plus factuel ou descriptif.

Illustrations par échantillons

Afin d'illustrer de manière qualitative la distinction entre les deux classes, quelques extraits représentatifs ont été sélectionnés aléatoirement depuis le corpus. Les exemples humoristiques mettent souvent en œuvre des jeux de mots, de l'ironie ou des détournements de sens, tandis que les exemples non humoristiques adoptent un style plus neutre ou descriptif.
Ces illustrations permettent de visualiser concrètement la diversité des styles et des intentions qui sous-tendent les deux classes du dataset. Elles serviront également de base pour les analyses qualitatives menées ultérieurement sur les erreurs de classification.

Figure 3: Nuage de mots pour les textes humoristiques

Figure 4: Nuage de mots pour les textes non humoristiques

5 Prétraitement des données

Le prétraitement constitue une étape fondamentale dans toute tâche de classification textuelle. Il vise à transformer les textes bruts en représentations plus adaptées à l'analyse automatique, tout en réduisant le bruit linguistique.
Dans un premier temps, les textes ont été convertis en minuscules afin d'uniformiser la casse, puis nettoyés de toute ponctuation, URL et caractères spéciaux. Cette étape permet de supprimer les éléments non pertinents pour l'analyse sémantique et de produire une version standardisée des textes. Le résultat de cette transformation initiale a été enregistré dans une colonne nommée `text_clean`.
Un second niveau de traitement a ensuite été appliqué à l'aide de la bibliothèque `spaCy`. Ce traitement plus avancé comprend plusieurs opérations successives :

- la suppression des mots vides (*stopwords*), qui sont des termes fréquents n'apportant généralement pas d'information discriminante (ex. : "the", "and", "but"),
- le filtrage des tokens non alphabétiques, afin d'exclure les chiffres ou autres symboles sans signification lexicale directe,
- la lemmatisation, qui consiste à ramener chaque mot à sa forme canonique (ou "lemme"), réduisant ainsi la variabilité morphologique tout en préservant le sens.

Ce processus, stocké dans la colonne `text_processed`, a pour effet de simplifier le vocabulaire sans appauvrir le contenu informationnel, ce qui est particulièrement pertinent dans un cadre de vectorisation ultérieure.
Enfin, une série de contrôles de qualité a été appliquée à l'issue du pipeline de nettoyage. Des assertions ont permis de vérifier la présence effective de toutes les colonnes attendues, leur non-nullité, ainsi que la conformité de la variable cible `humor`, qui a été transformée en type booléen. Ces vérifications garantissent la robustesse du jeu de données traité et sa compatibilité avec les étapes suivantes de modélisation.

6 Modélisation

6.1 Modèle de référence (Baseline)

Afin d'établir une référence initiale pour la tâche de classification binaire (humoristique vs non humoristique), un premier modèle a été construit à partir de représentations vectorielles classiques. Les textes prétraités ont été vectorisés à l'aide de la méthode TF-IDF (*Term Frequency-Inverse Document Frequency*), en conservant les 5000 termes les plus informatifs du corpus. Ce choix permet de capturer les dimensions discriminantes du langage tout en atténuant l'impact des mots trop fréquents ou peu spécifiques.
Le classifieur utilisé est un `SGDClassifier`, paramétré pour simuler une régression logistique via la fonction de perte `log_loss`. Ce modèle linéaire s'entraîne de manière itérative sur des mini-lots, ce qui le rend particulièrement adapté aux matrices creuses issues de la vectorisation TF-IDF. L'entraînement a été réalisé sur 20 itérations avec `warm_start=True`, permettant une mise à jour incrémentale des poids.
À l'évaluation sur l'ensemble de test, le modèle affiche une précision globale de 86 %, avec des F1-scores équilibrés à 0.86 pour les deux classes, comme illustré dans le rapport de classification ci-dessous :

Classe	Précision	Rappel	F1-score	Support
Non humoristique	0.85	0.87	0.86	19999
Humoristique	0.86	0.85	0.86	20000
Accuracy			0.86	
Moy. macro	0.86	0.86	0.86	

Le modèle et le vectoriseur ont été sauvegardés dans les répertoires respectifs `models/` et `vectorizers/`, assurant la traçabilité et la réutilisabilité de cette base pour des approches ultérieures.

6.2 Approches classiques

Dans un second temps, plusieurs modèles classiques d'apprentissage supervisé ont été testés, en association avec deux techniques de vectorisation : *TF-IDF* et *Bag-of-Words* (CountVectorizer). L'objectif était de comparer l'impact du choix du modèle et du type de représentation textuelle sur les performances de classification.
Les modèles évalués sont les suivants :

- Régression logistique (`LogisticRegression`)
- Forêt aléatoire (`RandomForestClassifier`)
- Naive Bayes multinomial (`MultinomialNB`)
- Support Vector Machine linéaire (`LinearSVC`)

Chaque modèle a été entraîné et évalué avec les deux vectoriseurs, limités à 5000 termes. L'évaluation a été réalisée sur un jeu de test stratifié selon la variable cible. Les résultats sont résumés dans le tableau ci-dessous :
Analyse : Les résultats montrent des performances globalement élevées pour tous les modèles, avec des F1-scores macros supérieurs à 0.84, quel que soit le vectoriseur. Les meilleurs résultats sont obtenus par SVM et régression logistique, avec un F1-score maximal de 0.8756 pour le couple *TF-IDF + SVM*. Naive Bayes obtient également de bons résultats, en particulier avec la vectorisation par comptage, ce qui est cohérent avec la nature multinomiale de ce modèle.
Ces résultats confirment la robustesse des approches classiques sur ce type de tâche, en particulier lorsqu'elles sont associées à une bonne représentation des données textuelles.

6.3 Modélisation par apprentissage profond

Afin d'explorer le potentiel des modèles de type Transformer dans la détection automatique de l'humour, deux architectures pré-entraînées ont été testées : `BERT` et `DistilBERT`. Ces modèles, fondés sur des représentations contextuelles profondes, permettent de capter des nuances sémantiques que les approches classiques ne peuvent appréhender.

BERT. L'implémentation initiale s'est appuyée sur le modèle `bert-base-uncased`, appliqué à un échantillon réduit de 5000 exemples en raison des contraintes computationnelles. Le tokenizer associé a été utilisé pour encoder les textes, tronqués à 128 tokens. Un unique cycle d'entraînement a été réalisé pour limiter le temps de calcul.
L'évaluation du modèle sur l'ensemble de test complet, bien que le modèle ait été entraîné sur un sous-échantillon, montre des performances équivalentes, voire légèrement supérieures à la baseline, avec une précision globale de 86 %. Le modèle montre un com-

portement complémentaire : une meilleure capacité à détecter les textes humoristiques (*recall* = 0.91), mais une précision légèrement moindre sur la classe non-humoristique.

DistilBERT. Face aux limites de temps et de mémoire liées à BERT, un modèle `DistilBERT`, version allégée de BERT (40 % plus léger), a été retenu pour les expérimentations ultérieures. Un échantillon de 5 % du dataset total a été utilisé, avec trois époques d'apprentissage. Malgré cette réduction drastique de données et de complexité, les résultats obtenus sont particulièrement prometteurs.
Le modèle atteint un **F1-score de 0.9726** sur l'ensemble de test, ce qui dépasse nettement les performances obtenues par tous les modèles précédents, classiques ou profonds. Cette amélioration peut être attribuée à la capacité de DistilBERT à généraliser à partir d'un faible volume de données, tout en conservant l'essentiel de l'architecture Transformer.

```
F1 score on test set:  0.9726
```

L'utilisation de modèles de type Transformer, même dans des versions réduites ou avec peu de données, offre un gain significatif en performance pour la détection automatique de l'humour. Ces résultats ouvrent la voie à une exploitation plus large de l'apprentissage profond dans les tâches de classification textuelle subjective.

7 Analyses complémentaires

Afin de mieux comprendre les limites des modèles de classification développés précédemment, plusieurs analyses exploratoires supplémentaires ont été menées. Ces études, bien que ponctuelles, apportent des éclairages qualitatifs sur le comportement des modèles et la complexité du phénomène humoristique.

Analyse d'erreurs. Une première investigation a porté sur les exemples mal classés par le modèle DistilBERT. Quelques cas représentatifs ont été extraits aléatoirement. Ces exemples révèlent des types d'humour subtils, implicites ou ambigus, difficiles à détecter même pour un modèle performant. Parmi les cas notables :

"The worst part of the robot uprising will be the constant software updates"
(humoristique, prédit comme non-humoristique)
"The rotation of the earth really makes my day"
(jeu de mots, mal classé)

Ces erreurs illustrent la limite des modèles à comprendre le second degré, les références culturelles ou encore les jeux de mots, autant d'éléments clés de l'humour.

Mesure de la surprise linguistique (surprisal). Une expérimentation a été menée à l'aide du modèle de langage GPT-2, afin d'estimer le degré de surprise (ou perplexité) de différents types de textes. Deux phrases ont été comparées :

- Une blague : *"Why don't skeletons fight each other? They don't have the guts."*
- Une phrase neutre : *"The cat sat on the mat."*

Le score de perte (approximant la surprise) retourné par GPT-2 était plus faible pour la blague (3.50) que pour la phrase neutre (4.50), suggérant que, paradoxalement, certains contenus humoristiques sont plus prévisibles linguistiquement. Cette observation, bien

que limitée, alimente la réflexion sur la distinction entre complexité syntaxique et effet humoristique.

```
Surprise 1 (humoristique) :  3.5005
Surprise 2 (neutre) :  4.5025
```

Traduction automatique de contenus humoristiques. Enfin, une traduction automatique via le modèle `MarianMT` (anglais → français) a été testée sur une blague typique :

"Why did the chicken cross the road? To get to the other side!"

devient :

"Pourquoi le poulet a traversé la route ? Pour arriver de l'autre côté !"

Cette expérience a mis en évidence que, bien que la traduction soit grammaticalement correcte, l'effet comique ne se transpose pas toujours, révélant les limites des systèmes de traduction dans la gestion du contenu à charge culturelle ou émotionnelle. Cela renforce l'idée que l'humour est un phénomène non seulement linguistique, mais aussi contextuel et culturel.

8 Conclusion et perspectives

8.1 Conclusion

L'humour représente un défi singulier dans le domaine du traitement automatique du langage en raison de sa nature profondément contextuelle, implicite et culturellement ancrée. Ce mémoire a présenté une démarche expérimentale structurée visant à concevoir un système de détection automatique de l'humour dans des textes courts en anglais.

Nous avons d'abord établi une ligne de base solide à l'aide de modèles classiques (régression logistique, SVM, Naive Bayes. . .), associés à des représentations vectorielles simples telles que TF-IDF ou Bag-of-Words. Ces approches, bien qu'efficaces, se sont avérées limitées face à la complexité sémantique de l'humour.

L'introduction de modèles de type *Transformer*, notamment BERT et sa version allégée DistilBERT, a permis d'améliorer significativement les performances, atteignant un F1-score de 0.9726 avec DistilBERT sur un sous-échantillon. Ces résultats démontrent que des architectures contextuelles, même entraînées de manière contrainte, surpassent largement les méthodes classiques sur cette tâche.

En complément de ces résultats quantitatifs, des analyses qualitatives (erreurs typiques, calcul de surprisal, traduction automatique) ont mis en évidence les subtilités linguistiques que même les meilleurs modèles peinent encore à capturer : ironie, sarcasme, jeux de mots, ou connaissances culturelles implicites.

8.2 Perspectives

Les travaux menés ouvrent la voie à de nombreux prolongements possibles, tant du point de vue technique que théorique :

- **Vers une meilleure interprétabilité** : intégrer des outils tels que `LIME` ou `SHAP` permettrait de visualiser les contributions lexicales ou syntaxiques à la prédiction du modèle, et de mieux comprendre ses biais et ses forces.

- **Enrichissement linguistique** : il serait pertinent d'ajouter des annotations de types d'humour (ironie, absurde, parodie...), ou de travailler sur des structures narratives (setup → punchline) pour guider les modèles.
- **Apprentissage faiblement supervisé** : exploiter les capacités zero-shot ou few-shot des grands modèles de langage permettrait d'annoter ou généraliser sur des données peu étiquetées, sans entraînement exhaustif.
- **Exploration cross-culturelle et multilingue** : l'étude de l'humour à travers plusieurs langues ou contextes culturels poserait des questions nouvelles sur l'universalité (ou non) des ressorts comiques.
- **Modélisation multimodale** : combiner texte, image, audio ou vidéo (comme les memes ou les clips humoristiques) permettrait d'aborder l'humour dans sa complexité réelle sur les plateformes numériques.

En définitive, l'humour reste un miroir fascinant des subtilités du langage humain. Le modéliser, c'est aussi apprendre à en mieux saisir la richesse, la subjectivité et les nuances. Ce mémoire constitue un jalon vers cette ambition.

Vectorizer	**Modèle**	**F1 (macro)**	**Accuracy**
TF-IDF	LogisticRegression	0.8752	0.8752
TF-IDF	RandomForest	0.8550	0.8550
TF-IDF	NaiveBayes	0.8676	0.8676
TF-IDF	SVM	**0.8756**	**0.8756**
Count	LogisticRegression	0.8755	0.8755
Count	RandomForest	0.8440	0.8441
Count	NaiveBayes	0.8708	0.8708
Count	SVM	0.8754	0.8754

Table 1: Résultats des modèles classiques selon les vectoriseurs

Classe	Precision	Recall	F1-score	Support
0 (non humoristique)	0.90	0.81	0.85	19999
1 (humoristique)	0.83	0.91	0.87	20000
Accuracy			**0.86**	39999

Table 2: Rapport de classification BERT