Janvier
2024

Lorenzo **BARBEY**
Axel **FEVEZ**

ISIMA

Intelligence Artificielle

Projet AI SPOTTER

Étude des outils permettant l'analyse et la création d'un détecteur de textes générés par IA et utilisation des RCR et du ML pour répondre à une partie de la problématique.

Table des matières

Lorenzo **BARBEY**
Axel **FEVEZ**

1. TALN ou comment s'est mis au point les outils nécessaires à l'analyse du langage naturel.

1.1 Fondements théoriques de la linguistique et l'analyse du langage

Depuis les années 1950, les travaux de Turing et de Chomsky ont permis d'offrir de nouvelles perspectives à l'étude du langage naturel (NLP).

Ses théories ont permis de proposer une approche nouvelle sur la syntaxe et l'étude de la langue. Succédant au paradigme majoritaire d'**analyse distributionnelle** qui consistait à étudier une langue à l'aide d'un échantillon (corpus), l'idée de Chomsky était de partir du postulat que chaque langage est limité par les processus mentaux des êtres humains et que, par cette limitation, il était possible de proposer des manières de structurer tous ces langages naturels. Cette nouvelle optique, tantôt acclamée, tantôt décriée, a néanmoins permis d'offrir une nouvelle perspective à l'analyse de la langue naturelle et a permis la mise au point d'outils pour travailler sur celle-ci

De ce fait, bien que l'approche soit différente de l'analyse distributionnelle, elle est loin d'être en opposition avec la précédente et, au contraire, vient compléter l'éventail d'outils permettant l'analyse du langage.

1.2 Applications pratiques : arbres d'analyse syntaxique et parsers

Pour mettre en lien ces deux concepts avec notre problématique initiale, il faut comprendre leur rapport avec l'intelligence artificielle. La grammaire générative et transformationnelle de Chomsky est à l'origine d'un outil d'étude de grammaticalité de la langue naturelle très connue, on parle ici des **arbres d'analyse syntaxique.** Ceux-ci ont permis de permettre une représentation des phrases dites «bien formés», en effet, par définition, "*Une phrase est grammaticale si et seulement s'il est possible d'en générer une analyse à l'aide d'une grammaire donnée*", c'est donc un outil de décision qui permet de décider si une phrase est grammaticale ou non. De ces arbres sont nés les **parsers** qui permettent d'analyser la syntaxe de phrase à l'aide d'arbres. On peut citer, par exemple, le framework **Stanford Parser** réalisé en JAVA, il permet de générer des arbres syntaxiques et donc d'offrir des méthodes de visualisation du langage et des associations qui peuvent être fais entre les entités de la phrase. *(Slide 1)*

1.3 Analyse distributionnelle et plongement lexical

Ensuite, penchons-nous un peu plus sur l'analyse distributionnelle. Celle-ci, comme expliquée brièvement ci-dessus, permet de définir des liens de sens entre les mots en se basant sur un échantillon de texte plus ou moins grand. L'idée nécessaire derrière cet échantillon est qu'il doit être représentatif de l'usage de la langue. Cette méthode a été étudiée et perfectionnée pour mettre au point des outils de mise en relation de certains mots lié par leurs sens et leur proximité d'utilisation. On parle ici de **plongement lexical** (word embedding en anglais) afin de traiter une liste de mots selon un corpus donné. L'idée première est de créer une représentation des mots à l'aide de vecteurs avec la propriété suivante : "Moins un vecteur de mot est distant d'un autre, plus les contextes d'utilisations de ces deux mots sont proches. On peut citer les groupes de modèles Word2vec ou GloVe1. Sans trop entrer dans les détails, le premier utilise un réseau de neurones à deux couches alors que le deuxième se base sur une technique de factorisation de matrice. *(Slide 2)*

En somme, on voit que l'étude linguistique a permis, des décennies après, la mise au point d'applications logicielles qui ont permis d'améliorer l'analyse et le traitement du langage naturel. Pour compléter notre brève étude, travaillons sur deux concepts très importants dans le domaine de l'intelligence artificielle : «**La représentation de connaissances**» et le «**Machine Learning**».

2. La représentation de connaissances comme outil d'organisation, de structuration et de simplification de l'information.

2.1 Rôle fondamental de la représentation de connaissances

Pour prendre des décisions sur des problèmes complexes, l'informatique c'est beaucoup appuyé sur le concept de «représentation de connaissances». Celle-ci permet d'offrir une sémantique plus claire et de permettre de comprendre comment fonctionnent le raisonnement et la prise de décision d'un système. Pour analyser un texte, on va partir sur la liste de neuf bases de faits censés nous donner des réponses basées sur les caractéristiques de nos textes.

A noter qu'il est très difficile de déterminer efficacement si une IA ou si un être humain a généré un texte par représentation de connaissances. L'idée ici est de simplifier un maximum notre problématique et ainsi de partir de «postulats» qui nous permettront de déterminer des informations clés pour résoudre notre question initiale.

2.2 Graphe de connaissances pour représenter les règles

- **L'extrait a été rédigé par une IA.**
  → extrait(IA)
- **L'extrait a été rédigé par un humain.**
  → extrait(humain)
- **Si l'extrait est rédigé par un humain alors il n'est pas rédigé par une IA.**
  → extrait(humain) ⇒ ¬ extrait(IA)
- **Si un extrait comporte des mots mal-écrits alors il est rédigé par un humain.**
  → contient(extrait, superieur(faute, 0)) ⇒ extrait(humain)
- **Si un extrait comporte peu de répétitions alors il n'a pas été rédigé par un humain.**
  → contient(extrait, inferieur(repetition, 2)) ⇒ ¬ extrait(humain)
- **Si un extrait comporte peu de répétitions, pas d'erreurs et qu'une des phrases est agrammaticale alors il n'a pas été rédigé par une IA.**
  → contient(extrait, inferieur(repetition, 2)) ^ contient(extrait, egal(faute, 0)) ^ contient(extrait, superieur(agrammaticalité, 0)) ⇒ ¬ extrait(IA)
- **Si un être humain écrit un extrait alors les phrases feront moins de 20 mots.**
  → extrait(humain) ⇒ contient(phrase, inferieur(taille, 21))
- **Si une phrase est non-signifiante et ne contient pas de mots mal-écrits alors il est rédigé par un humain.**
  → contient(phrase, superieur(non-signifiante, 0)) ^ contient(extrait, inferieur(faute, 0)) ⇒extrait(humain)
- **Si un extrait contient au moins une phrase non-signifiante, agrammaticale et composée de successions d'un même caractère ou de mêmes syllabes alors l'extrait est rédigé par un humain.**
  → contient(phrase, superieur(non-signifiante, 0)) ^
  contient(extrait, superieur(agrammaticalité, 0)) ^
  (contient(mot, superieur(lettreSuccessive, 3)) v
  (contient(mot, superieur(syllabeSucessive, 4)) ⇒ extrait(humain)

Pour représenter cette base de règles, on utilise un graphe de connaissance. *(Slide 3)*

Cette manière d'utiliser l'information est donc un moyen de trouver des réponses à des problèmes délicats en simplifiant en partie la problématique et en rendant notre réponse moins précise.

Finalement, pour travailler sur des extraits de textes, on utilise de nombreux outils d'analyse afin d'être le plus complet possible. Ces outils permettent d'interagir avec les spécifiées du langage naturel et de fournir des modèles permettant de répondre le plus précisément possible à un problème donné, c'est ce qu'on appelle le Machine Learning.

3. Le Machine Learning et ses applications dans un projet d'analyse de textes.

Dans cette troisième phase de notre projet, explorons comment le Machine Learning (ML) est utilisé pour aborder notre problématique de détection de textes générés par intelligence artificielle. Le ML offre une approche puissante pour analyser des données complexes et formuler des prédictions basées sur des modèles préalablement entraînés.

Avant de commencer, le code est disponible sur git[1].

3.1 Constitution du jeu de données

Notre jeu de données a été soigneusement rassemblé à partir de Kaggle[2].

Cette base de données contient une variété de textes générés, que ce soit par des êtres humains ou par des intelligences artificielles. Les réponses humaines proviennent de sources respectables telles que "The Encyclopedia of Computer Science and Technology" et "Encyclopedia of Human-Computer Interaction". Les réponses générées par l'IA ont été obtenues en posant des questions à ChatGPT et en documentant méticuleusement les réponses obtenues.

Après cette phase de collecte, les données ont été nettoyées en éliminant les caractères indésirables, les balises de style, et autres éléments de formatage.

3.2 Configuration des paramètres du modèle

La mise en place de notre modèle d'analyse a nécessité une attention particulière à certains paramètres cruciaux, adaptés spécifiquement à notre ensemble de données :

- **Taille du Vocabulaire (*vocab_size*)**: Nous avons fixé cette valeur à 1000, limitant ainsi le modèle aux 1000 mots les plus fréquents dans nos données. Cela permet de concentrer l'attention sur les termes les plus pertinents.
- **Dimension de l'Espace d'Embedding (*embedding_dim*)**: Nous avons opté pour une dimension de 50, définissant ainsi la représentation vectorielle de chaque mot dans notre vocabulaire.
- **Longueur Maximale d'une Séquence (*max_sequence_length*)**: Fixée à 1500, cette valeur détermine la limite de mots examinés par le modèle pour chaque article. Cette valeur a été choisie pour garantir que le modèle puisse analyser une quantité significative d'informations dans chaque article

On a choisi de limiter le vocabulaire à 1000 mots et d'utiliser une représentation de 50 dimensions pour mieux repérer les répétitions importantes et améliorer la façon dont le modèle comprend les mots.

3.3 Entraînement du modèle

Les résultats de l'entraînement sur 10 epochs sont encourageants :

- ✓ **Précision sur l'Ensemble d'Entraînement**: La précision atteint environ 99.02% à la fin de l'entraînement, indiquant une excellente adaptation aux données d'entraînement.
- ✓ **Perte sur l'Ensemble d'Entraînement**: La valeur de perte diminue progressivement, atteignant environ 0.0460 à la dernière epoch, signifiant de la convergence du modèle.
- ✓ **Précision sur l'Ensemble de Validation**: La précision sur l'ensemble de validation se situe autour de 90.20%, prouvant sa capacité de généralisation.
- ✓ **Perte sur l'Ensemble de Validation**: La valeur de perte sur l'ensemble de validation se maintient à environ 0.3512.

[1] https://github.com/axelfvvv/IAProject
[2] https://www.kaggle.com/datasets/mahdimaktabdar/chatgpt-classification-dataset/

Ces résultats suggèrent une capacité du modèle à apprendre efficacement sur l'ensemble d'entraînement et à généraliser avec succès sur de nouvelles données.

3.4 Perspectives d'amélioration

3.4.1 Opérationnalité du modèle

Notre modèle actuel excelle dans l'analyse des éléments de notre base de données spécifique. Cependant, il est crucial de souligner que son optimisation se limite à cette base de données particulière. En dehors de ce contexte spécifique, sa performance pourrait présenter des variations notables. Cela souligne l'importance d'envisager une expansion de la base de données pour une amélioration plus globale de la performance du modèle. Une base de données plus diversifiée pourrait potentiellement renforcer sa capacité à généraliser et à s'adapter à une variété plus étendue de textes générés par IA.

3.4.2 Gestion des fautes

Pour garantir une détection d'erreurs plus pointue, notre modèle actuel se concentre sur l'élimination des caractères indésirables et des balises de style. Une amélioration potentielle consisterait à explorer des approches plus avancées, telles que l'intégration de techniques de correction orthographique ou l'utilisation de modèles spécialisés dans la détection d'erreurs. Cela renforcerait notre capacité à traiter de manière plus précise les fautes orthographiques et à améliorer la fiabilité du modèle.

4. Contributions des membres

Nous avons réparti les tâches de manière collaborative, chacun apportant ses compétences spécifiques au projet AI SPOTTER. Voici un aperçu des contributions individuelles :

4.1 Lorenzo

Lorenzo a initié la recherche d'un dataset adapté, puis a développé un script permettant de générer un modèle, basé sur les bibliothèques TensorFlow et scikit-learn de Python, à partir de données(dataset et paramètres) préalablement définies.

Une fois le modèle créé, Lorenzo a mis en place une interface utilisateur, utilisant des technologies web telles que HTML, CSS, et JavaScript. Pour la partie backend, il a développé une API Flask en Python permettant de tester les textes avec le modèle de machine learning qu'il a créé

4.2 Axel

Axel s'est plongé dans la partie théorique du projet en se focalisant sur la recherche fondamentale pour définir comment traiter le texte. Il a approfondi sa compréhension des algorithmes d'analyse de texte en explorant comment les phrases sont formées, cherchant à saisir les mécanismes qui les animent.

Axé sur la distinction essentielle entre les caractéristiques des textes créés par des êtres humains et ceux générés par des intelligences artificielles (IA), Axel a élaboré des règles pour identifier ces différences. Plus spécifiquement, il a conçu un modèle CSP, détaillant les différentes composantes de manière précise.