Janvier
2023

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

ISIMA

Intelligence Artificielle

Université Clermont Auvergne

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.

1. **TALN ou comment c'est mis au point les outils nécessaire à l'analyse du langage naturel.**

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, de par cette limitation, il était possible de proposer des manières de structurer tout ces langages naturelles.. Cette nouvelle optique, tantôt acclamée, tantôt décriée, a néanmoins permis d'offrir une nouvelle perspective à l'analyse de la langue naturel et à permis la mise au point d'outils pour travailler sur celle-ci
De ce fait, bien que l'approche est 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.

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 connu, 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 si 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 fait entre les entités de la phrase. (slide 1)

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 cette 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 relations 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 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 simplifcation de l'information.**

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 clair et de permettre de voir clairement comment fonctionne le raisonnement et la prise de décision d'un système. Pour analyser un texte, on va partir sur la liste de 8 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 un humain a généré un texte par représentation de connaissance. L'idée ici est de simplifier un maximum notre problématique et partir de «postulats» qui nous permettront de déterminer des informations clés pour résoudre notre question initiale.

Janvier Lorenzo **BARBEY**
2023 Axel **FEVEZ**

1. L'extrait a été rédigé par une IA.

→ extrait(IA)

2. L'extrait a été rédigé par un humain.

→ extrait(humain)

3. Si un extrait comporte des mots mal-écrits alors il est rédigé par un humain.

→ contient(extrait, superieur(faute, 0)) → extrait(humain)

4. Si un extrait comporte peu de répétitions alors il n'a pas été rédigé par un humain.

→ contient(extrait, inferieur(repetition, 2)) → ¬ extrait(humain)

5. 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)

6. Si un être humain écrit un extrait alors les phrases feront moins de 20 mots.

→ extrait(humain) → contient(phrase, inferieur(taille, 21))

7. 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)

8. Si un extrait contient au moins une phrase non-signifiante, agrammaticale et composée de successions d'un même caractère ou de même 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)) ∨ (contient(mot, superieur(syllabeSucessive, 4))
→ extrait(humain)

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